CentOS 7 操作命令-基础篇 1.2

说明：

1.本文档没有目录，本文档在发布时为 pdf 文档，有章节书签，可以下载到本地来查看，点击书签进入相应的章节。

2.蓝色的字为配置命令，绿色的字为命令的注释，有时命令太密集时，就不用蓝色标出了。

3.本文档仅为基础操作教程，不涉及网络服务的配置（比如 web 服务器等）

4.注意：本文档的所有操作请先在在虚拟机里进行实践，请不要直接在真实的服务器中操作！

作者：李茂福

日期：2019 年 12 月 27 日

0、下载并安装 CentOS 7 系统

系统安装镜像下载地址 1：http://mirrors.163.com/centos/7.7.1908/isos/x86_64/

下载第 2 个，文件大小 4G 的那个

Index of /centos/7.7.1908/isos/x86_64/

../		
0_README.txt	17-Sep-2019 02:44	2495
CentOS-7-x86_64-DVD-1908.iso	12-Sep-2019 02:51	4G
CentOS-7-x86_64-DVD-1908.torrent	17-Sep-2019 20:39	87K
CentOS-7-x86_64-Everything-1908.iso	10-Sep-2019 03:09	10G
CentOS-7-x86_64-Everything-1908.torrent	17-Sep-2019 20:38	103K
CentOS-7-x86_64-LiveGNOME-1908.iso	17-Sep-2019 02:57	1G
CentOS-7-x86_64-LiveGNOME-1908.torrent	17-Sep-2019 20:39	29K
CentOS-7-x86_64-LiveKDE-1908.iso	17-Sep-2019 03:27	2G

系统安装镜像下载地址 2：http://mirrors.aliyun.com/centos/7/isos/x86_64/

下载第 2 个，4664066048 字节（4G 多）的那个

Index of /centos/7/isos/x86_64/

../		
0_README.txt	16-Sep-2019 18:44	2495
CentOS-7-x86_64-DVD-1908.iso	11-Sep-2019 18:51	4664066048
CentOS-7-x86_64-DVD-1908.torrent	17-Sep-2019 12:39	89467
CentOS-7-x86_64-Everything-1908.iso	09-Sep-2019 19:09	11026825216

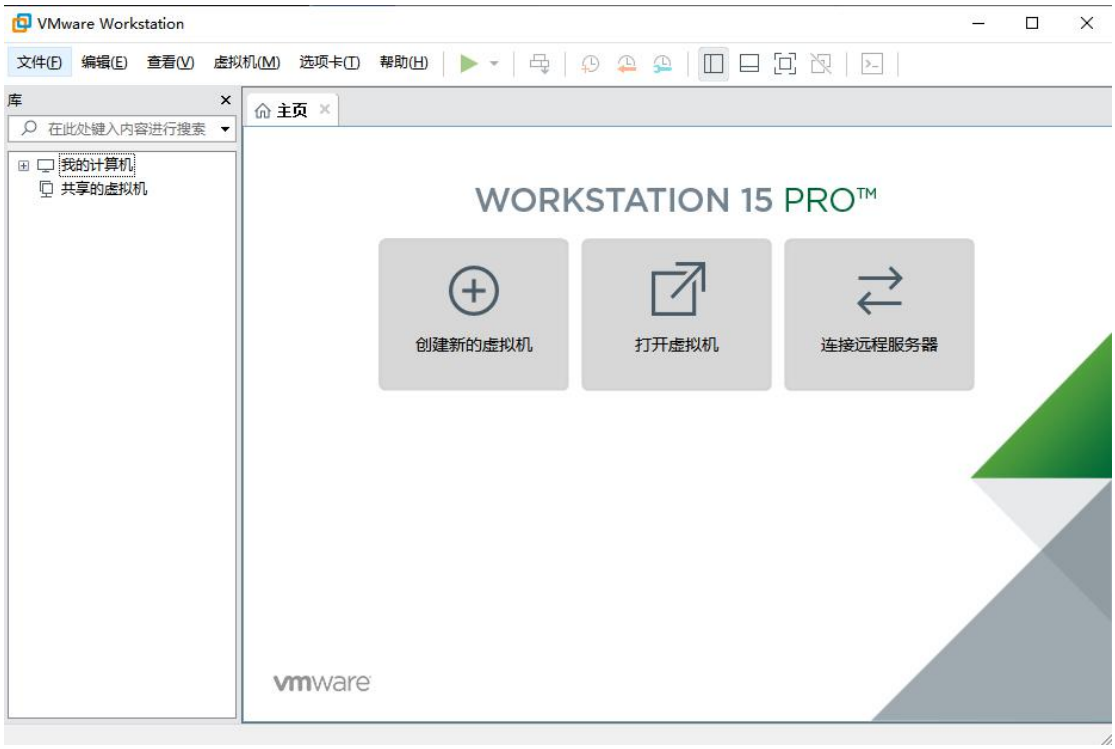
下载到本地磁盘后，初学者建议安装到虚拟机里，虚拟机软件可以用以下 2 个（任选一个）

VMware Workstation

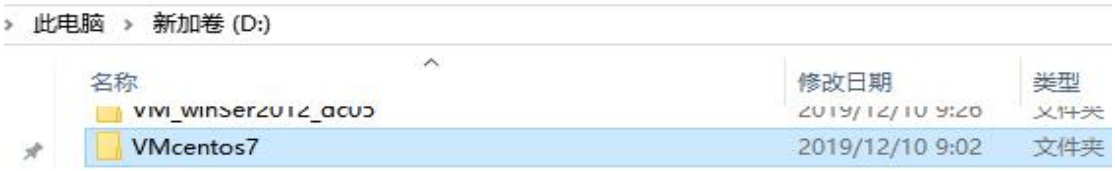
Oracle VM VirtualBox

本教程使用 VMware Workstation 虚拟机软件，下载并安装，试用一个月。

双击桌面的图标，进入主界面：



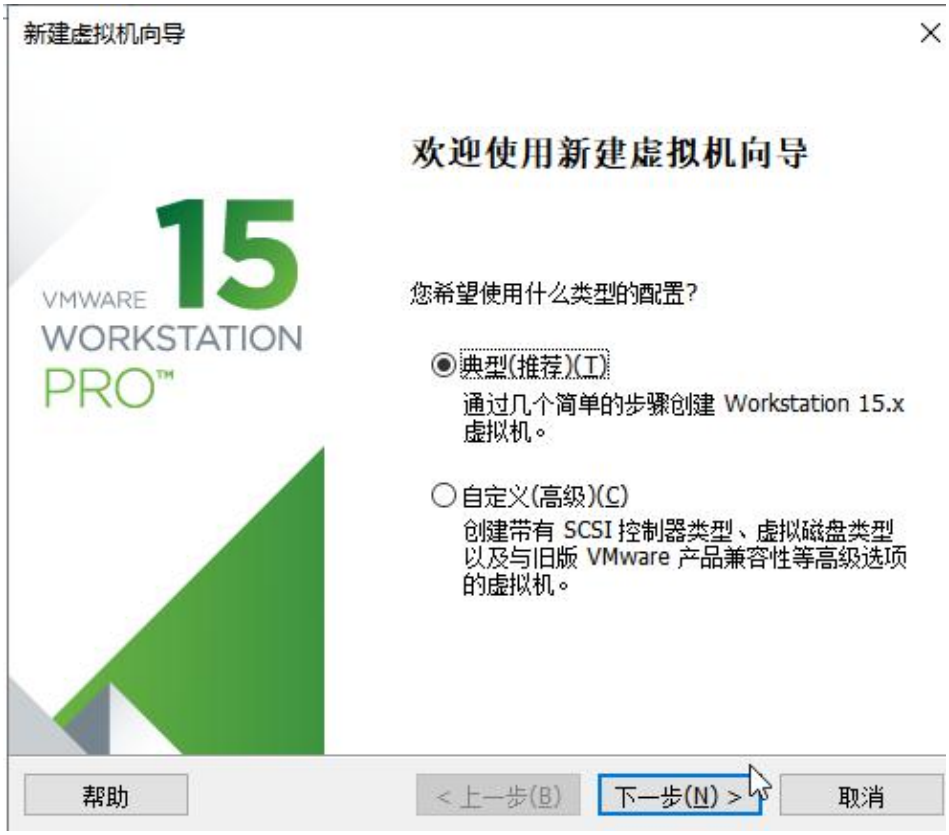
创建新的虚拟机之前，先在磁盘上创建一个文件夹，随便命名（比如在 D 盘上创建一个名为 VMcentos7 的文件夹），然后这个文件夹就是接下来我们要创建的虚拟机的虚拟磁盘。



虚拟机就是虚拟出来的一台计算机，刚创建时是空白的，可以给它添加磁盘，网卡，光盘，CPU，内存等计算机资源，当虚拟机关机时，分配给它的资源是不会被使用的，当虚拟机开机时，这些资源就被它使用了。这些资源是共用我们自己的这台物理机的，所以物理机的性能 不能太差。

1.在主页面上点击创建新的虚拟机

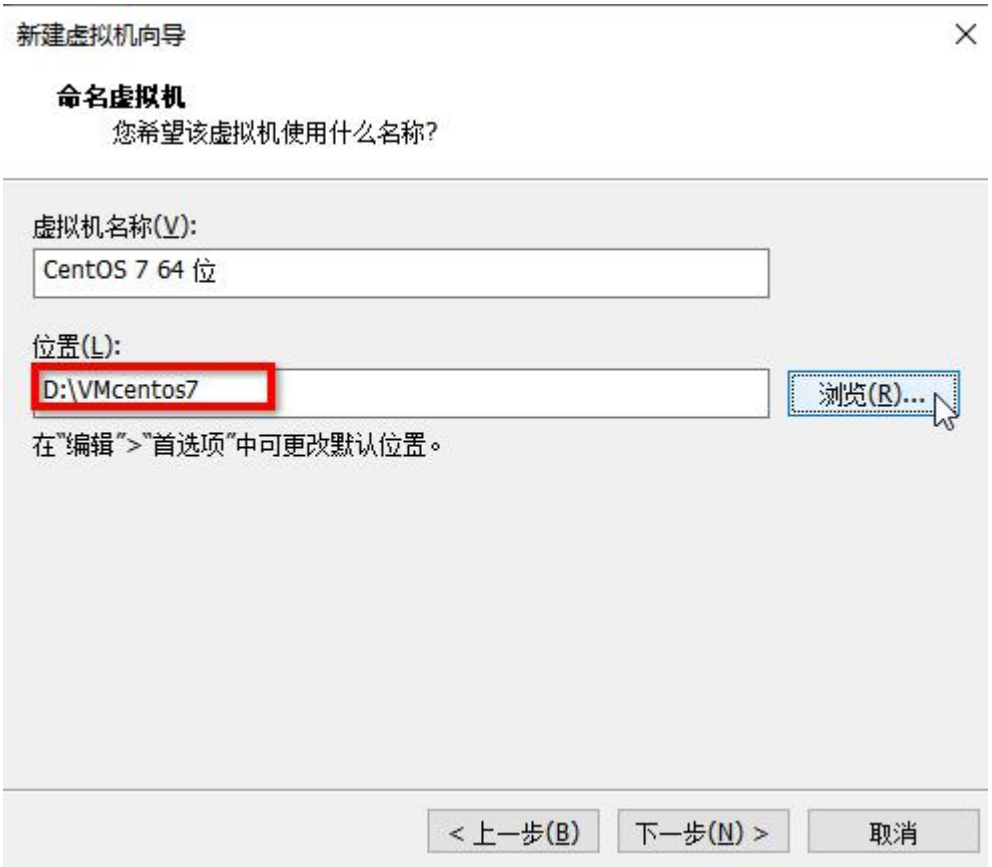
2.使用典型配置，下一步



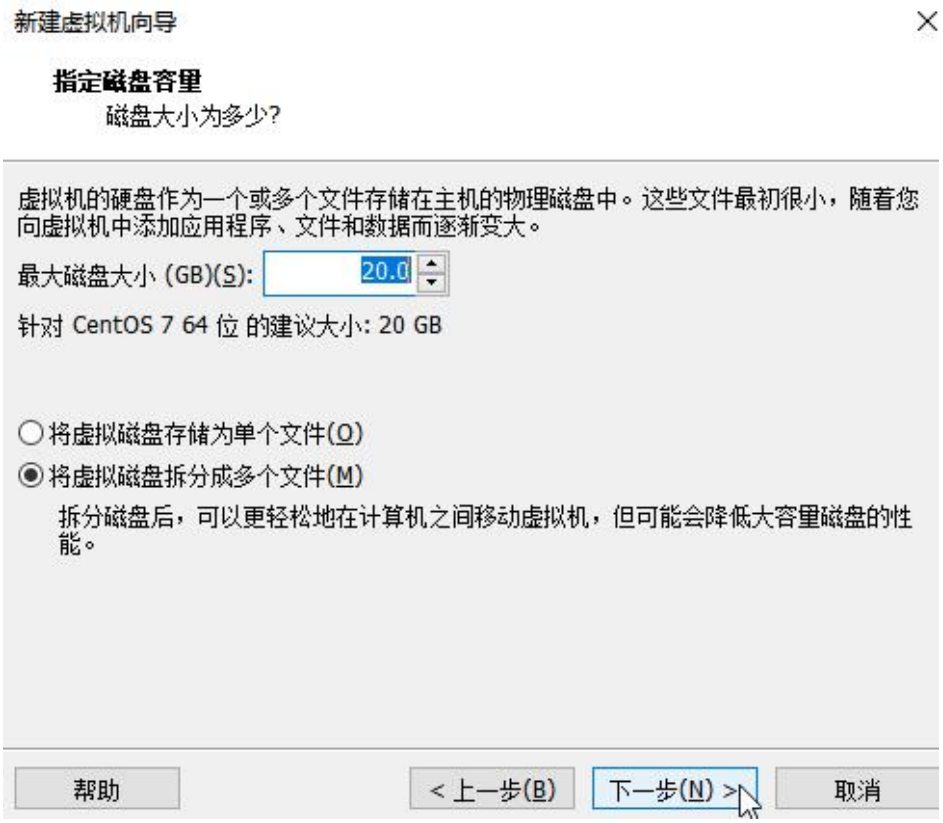
3.选择“稍后安装操作系统”，下一步

4.选择 Linux，版本为 CentOS 7 64 位，下一步

5.虚拟机名称用默认的就行，位置为 D:\VMcentos7，我们前面创建的那个文件夹



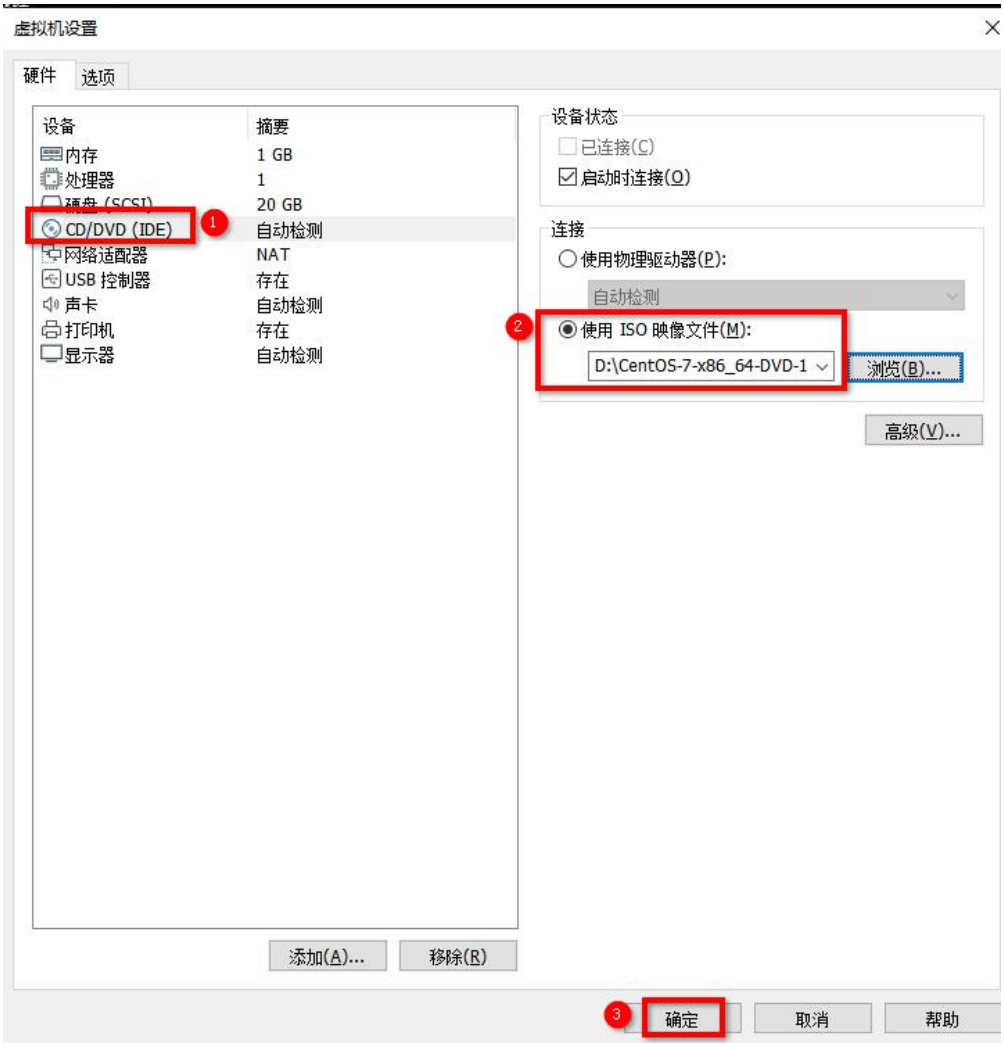
6.最大磁盘大小分配 20GB 就够了，下面选择将虚拟机拆分成多个文件，这样我们的物理机的磁盘就不会立即被使用 20GB，而是该虚拟机使用了多少，我们的物理磁盘就被用掉多少，动态分配的。点击“下一步”



7.下图就是我们创建的虚拟机的参数，点击“完成”

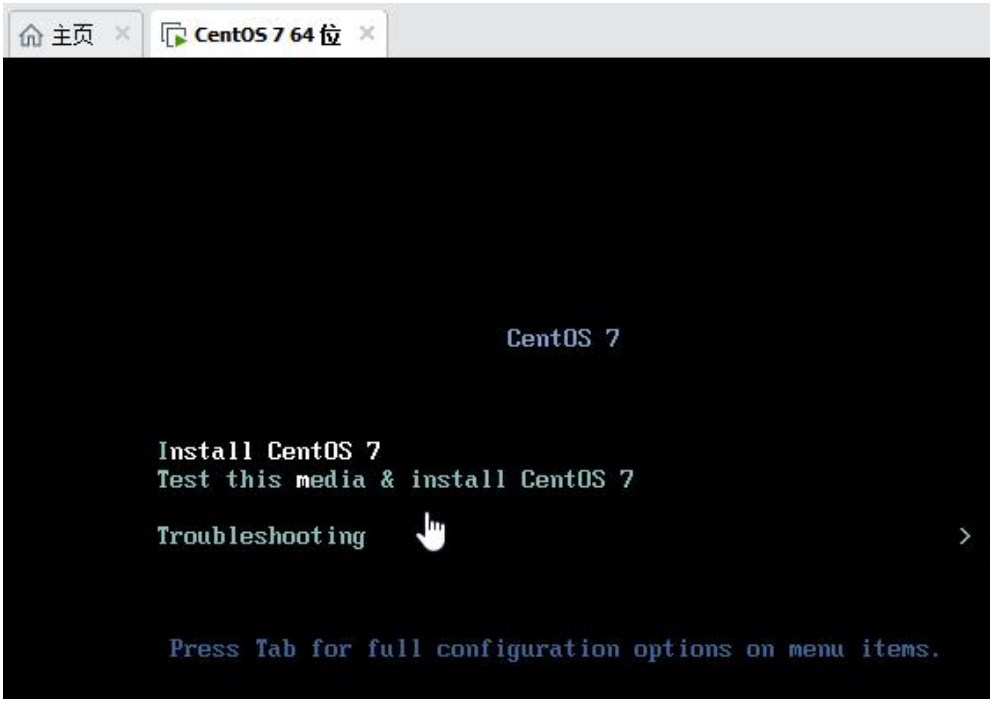
8.我们需要给这个虚拟机安装 Centos7 的系统，怎么安装？可以通过它的虚拟光驱，即 CD/DVD 这个设备，把我们之前下载的系统镜像文件放进去就行。单击虚拟机的 CD/DVD

9.在弹出来的虚拟机设置里，点击左边的 CD/DVD，右边选择使用 ISO 映像文件，然后点浏览，在本机磁盘里找到我们之前下载的安装镜像文件，打开。然后点右下角的“确定”

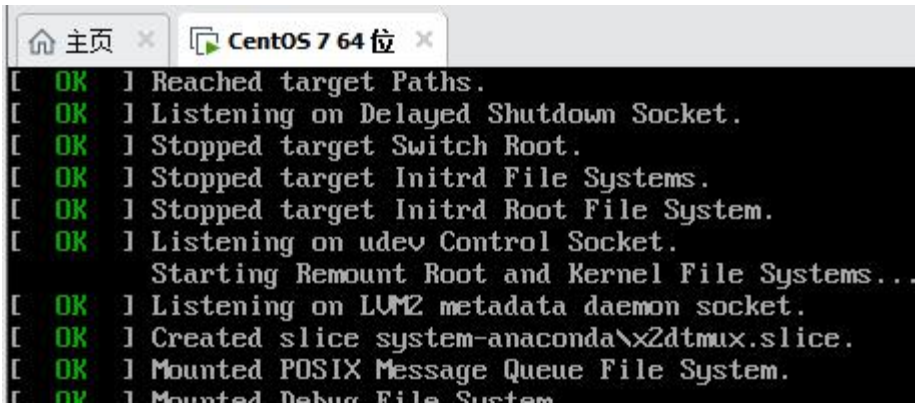


10.点击“开启此虚拟机”，进行操作系统的安装（安装的原理和真实的差不多，虚拟机的硬盘刚创建时是空白的，所以虚拟机只能从 DVD 启动，然后安装系统到虚拟机的虚拟硬盘里）

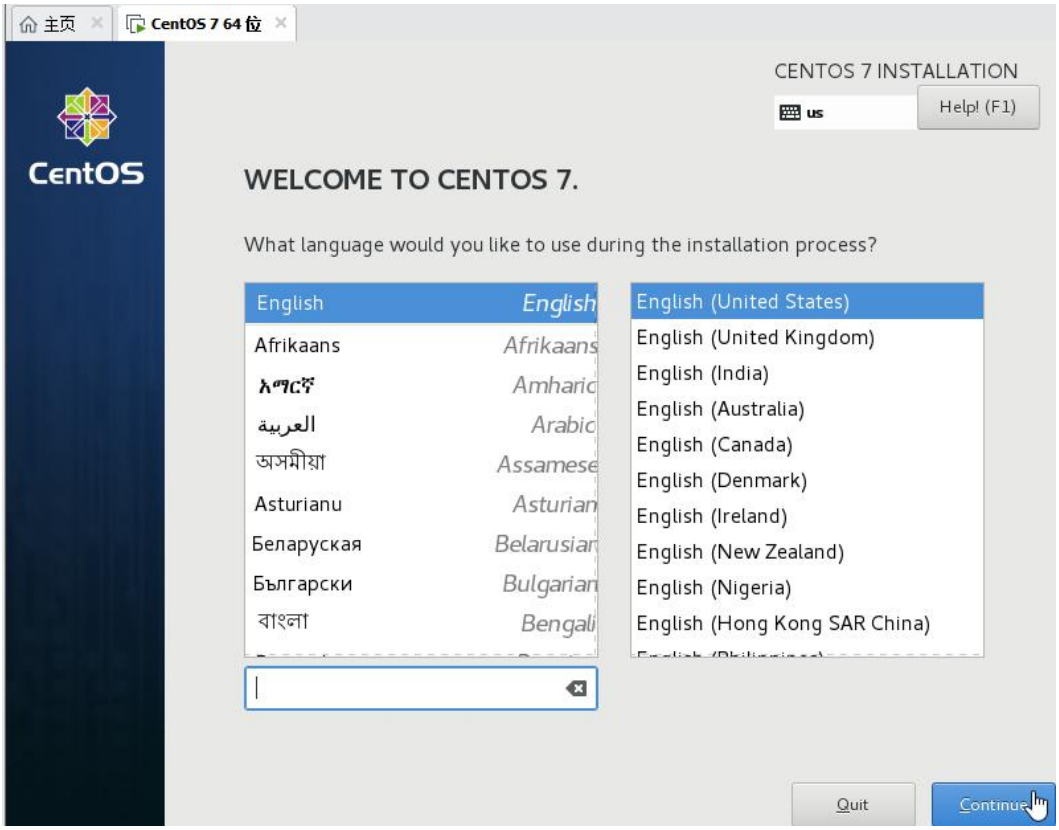
11.先用鼠标点击下图的界面，当鼠标进入控制台后，按上键↑，选择 **Install CentOS 7**，回车



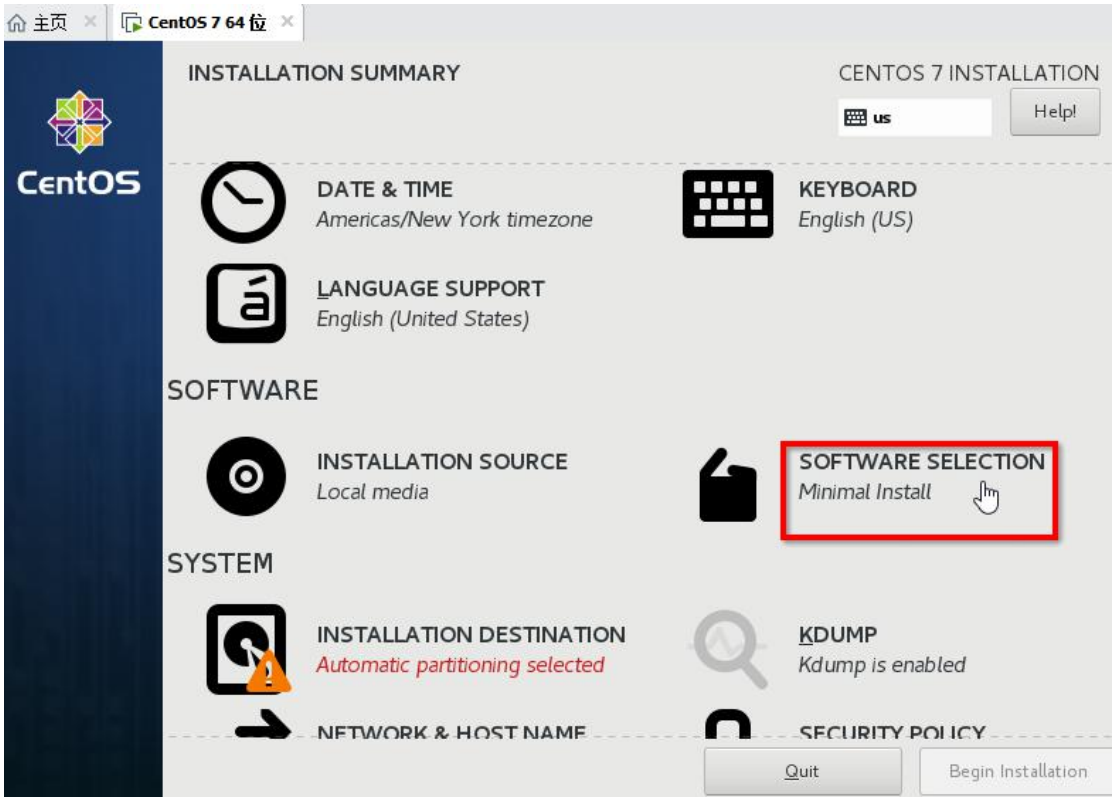
12.回车之后，控制台会有字符输出，和真实的设备安装系统时一样



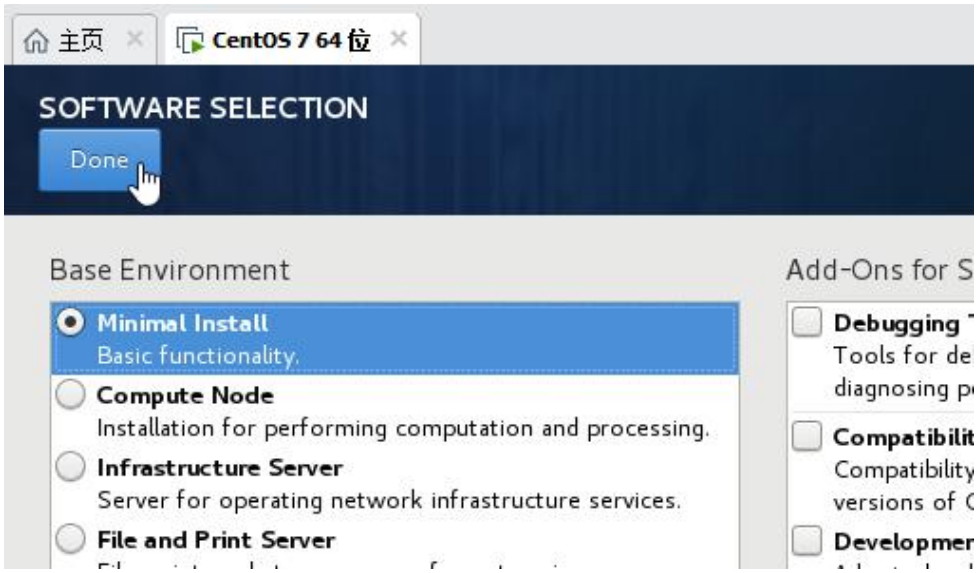
13.出现下图时，可以选择系统的语言，用默认的 English 就行，点击右下角的 Continue



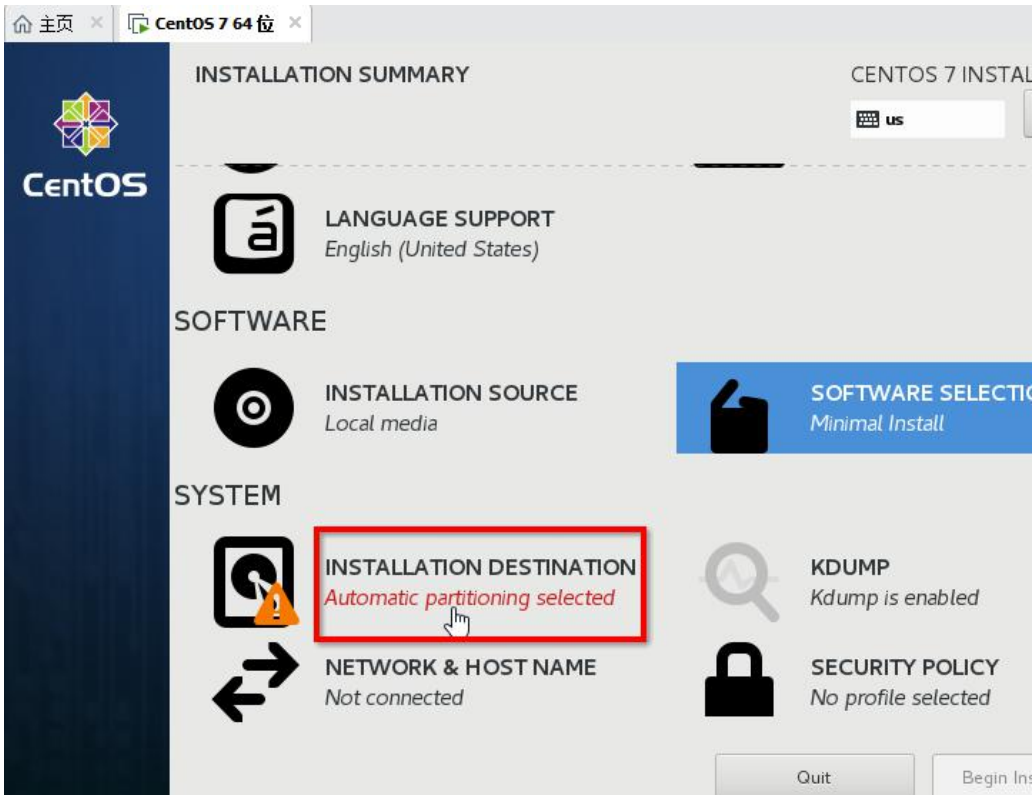
14.鼠标点击 SOFTWARE SELECTION，选择要安装的版本



15.推荐使用最小化安装 Minimal Install，然后 点击左上角的 Done

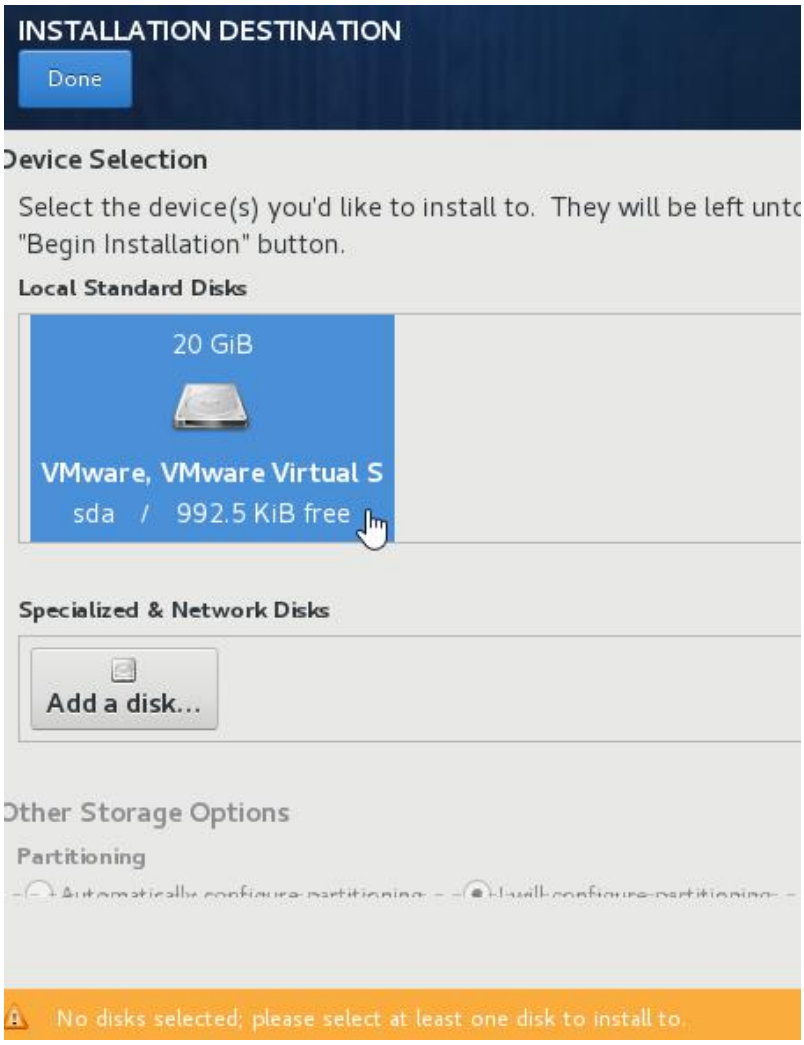


16.再点击 INSTALLATION DESTINATION 选择安装的磁盘以及给磁盘进行分区

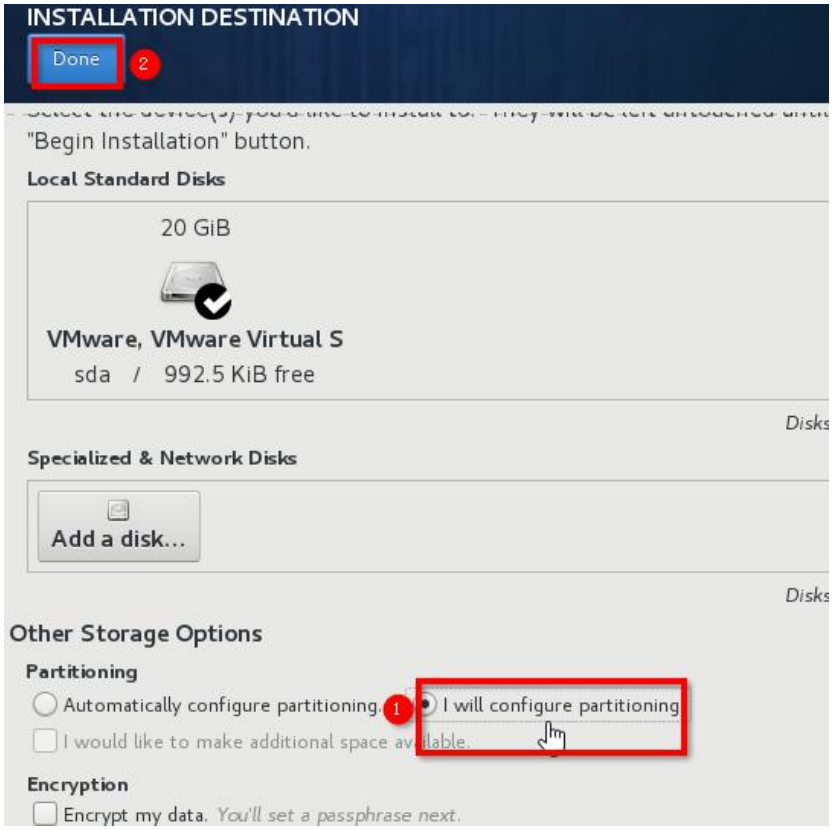


17.下图中显示的 20GB 的磁盘就是我们给虚拟机分配的虚拟硬盘

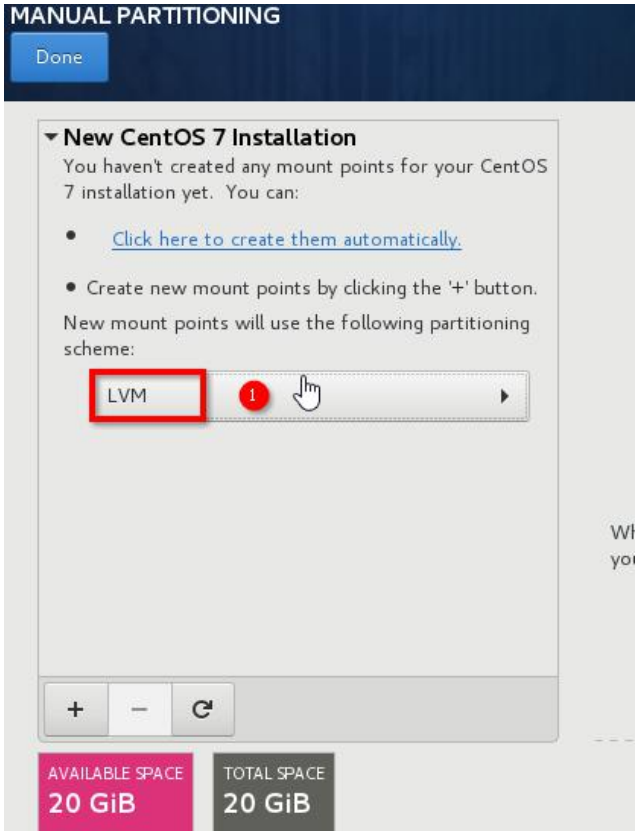
18.然后点击一下这个磁盘，底行文字提示没有选择任何磁盘，不慌，再点击一次



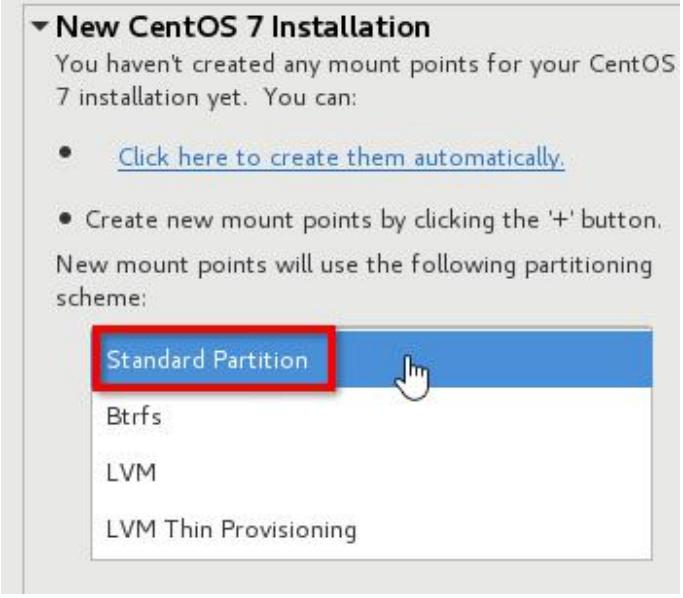
19.这时可以在 Other Storage Options 下面选择 I will configure partitioning，点击 Done



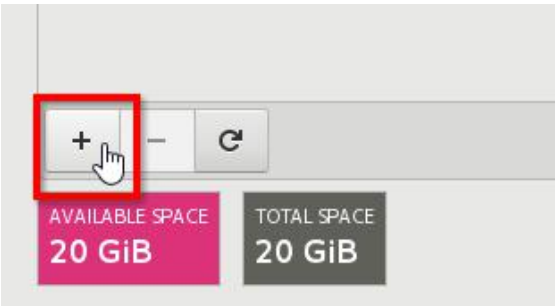
20.然后进入了分区界面，左下角红色区域显示可用空间为 20GB，点击中间的 LVM 的那行



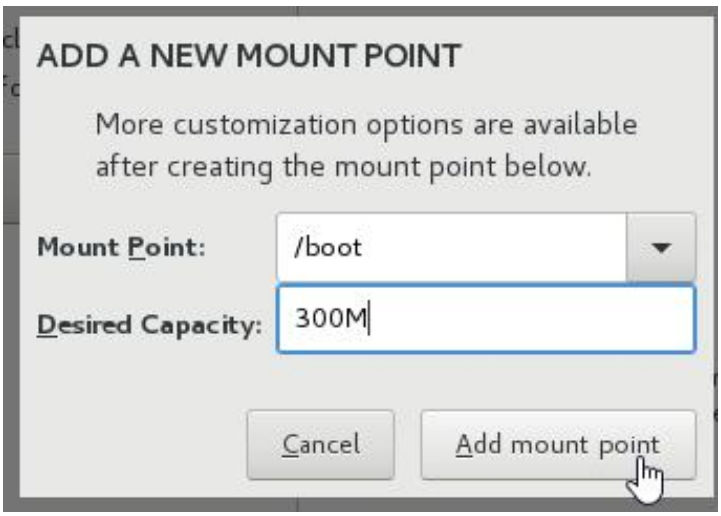
20.选择标准分区 Standard Partition



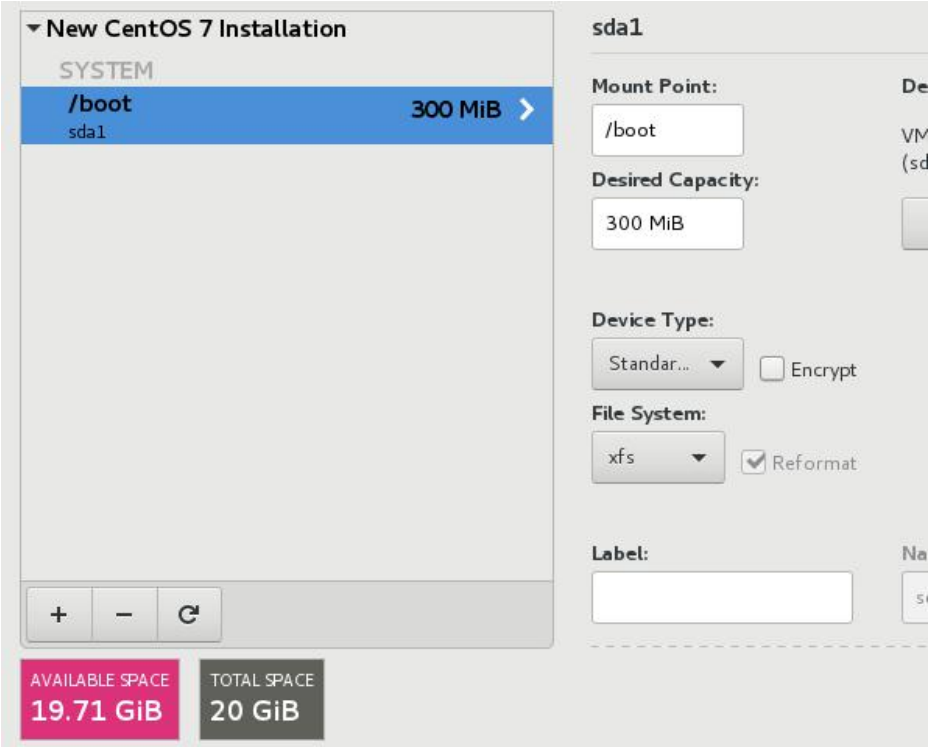
21.然后在分区界面的左下角 点击加号+ 添加一个分区



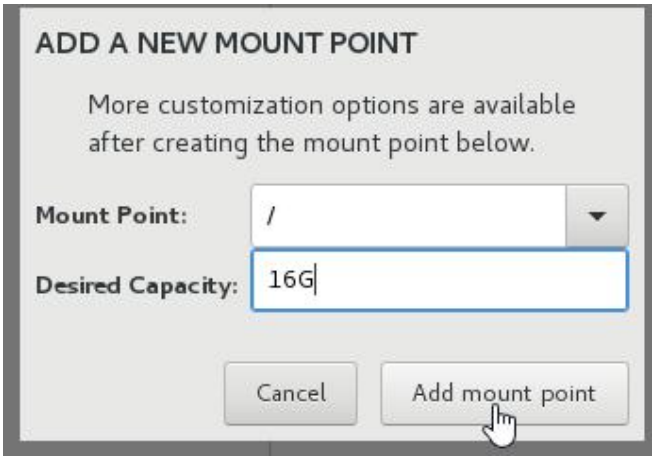
22.在添加分区对话框里，Mount Point 选择**/boot** 大小为 300M，再点击 Add mount point



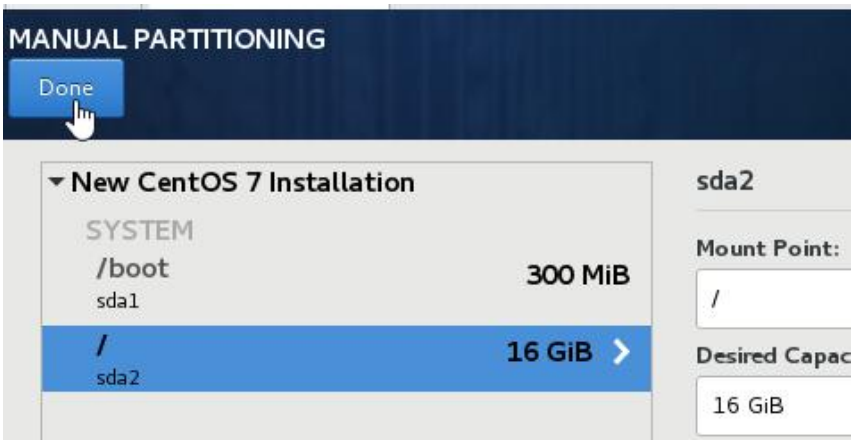
23.这时我们看到分区界面里多了一个/boot 分区，左下角红色区域显示可用空间为 19.71G



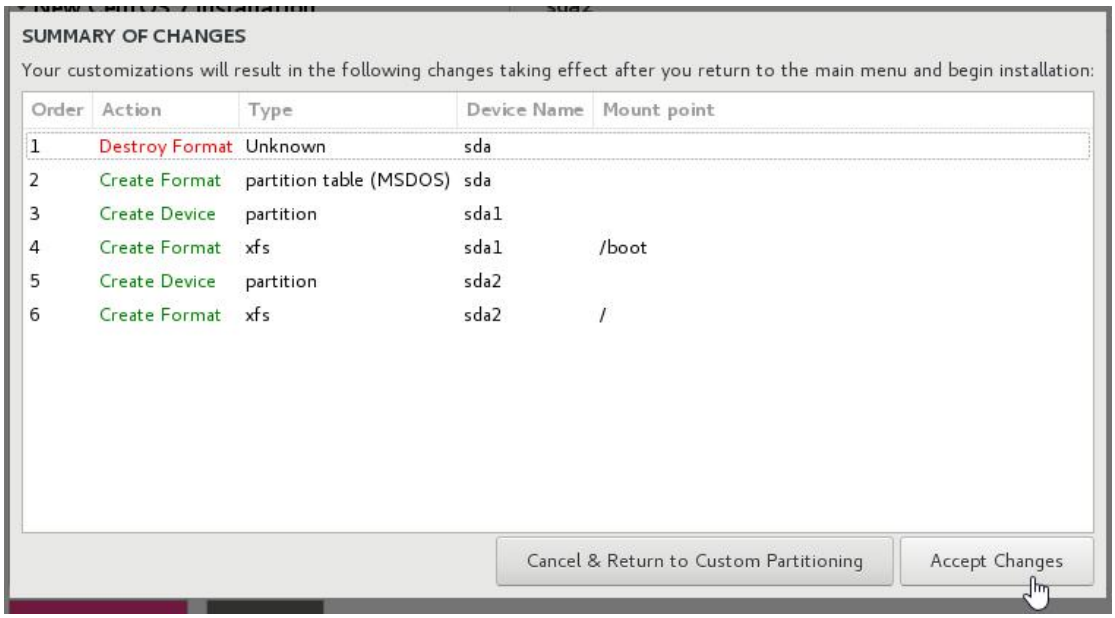
24.继续添加分区，点击左下角的加号+ Mount Point 为 **/** 表示根分区，大小 16G（不要把剩下的磁盘容量都用完，可以留 2 个 G 做实验）



25.回到了分区界面，可见刚刚创建的 2 个分区，然后点击左上角的 Done，点击 2 次

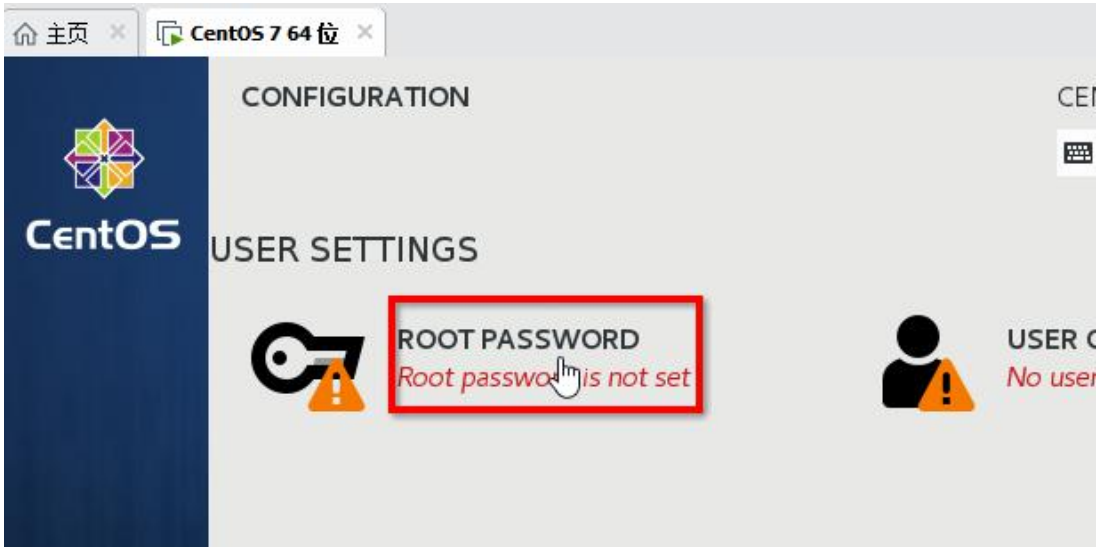


26.弹出提示框，确定，点击左下角的 Accept Changes

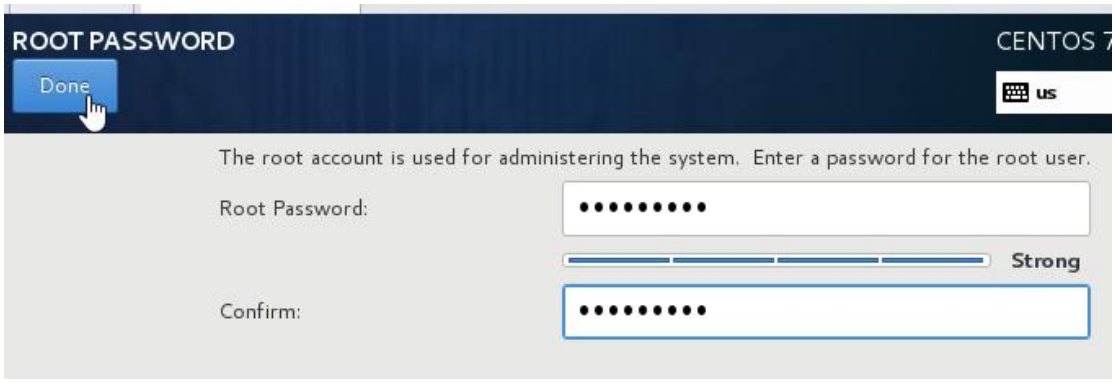


27.回到了主配置界面，点击左下角的 Begin Installation，开始安装系统

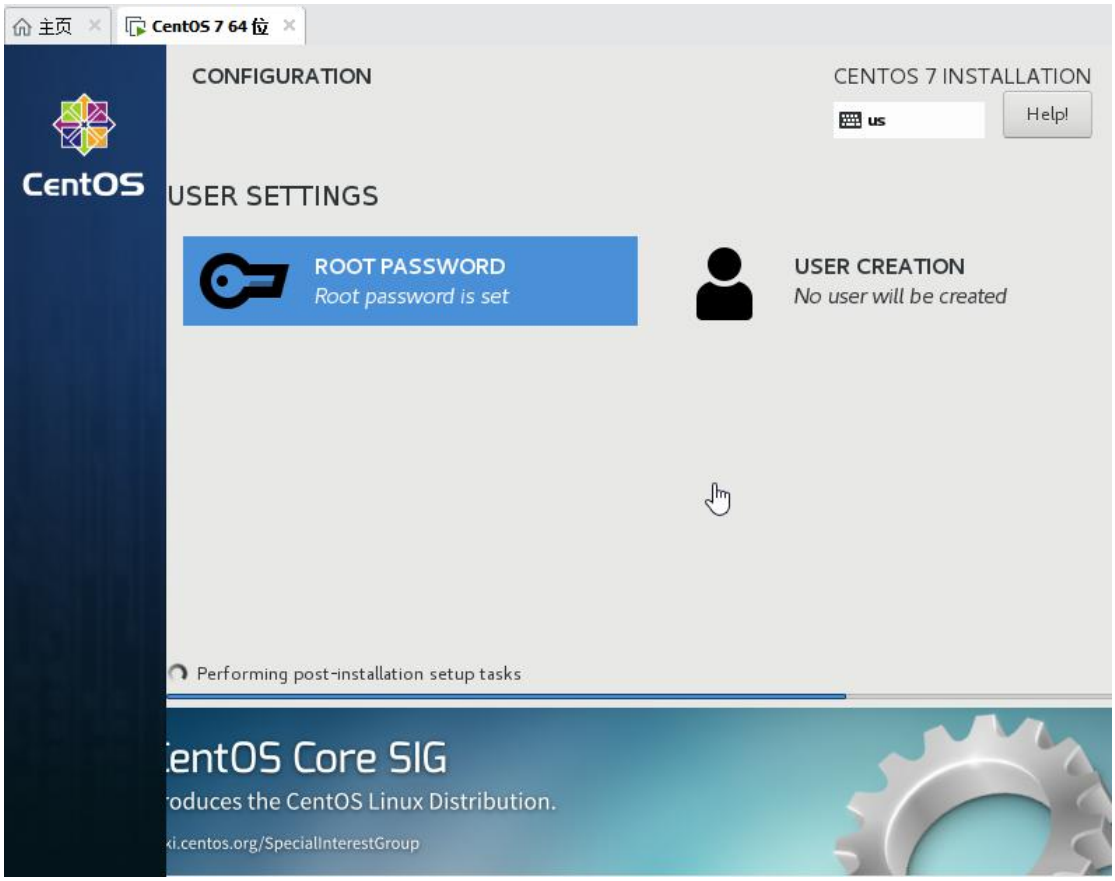
28.在安装过程中，我们可以创建 Root 密码，点击 ROOT PASSWORD



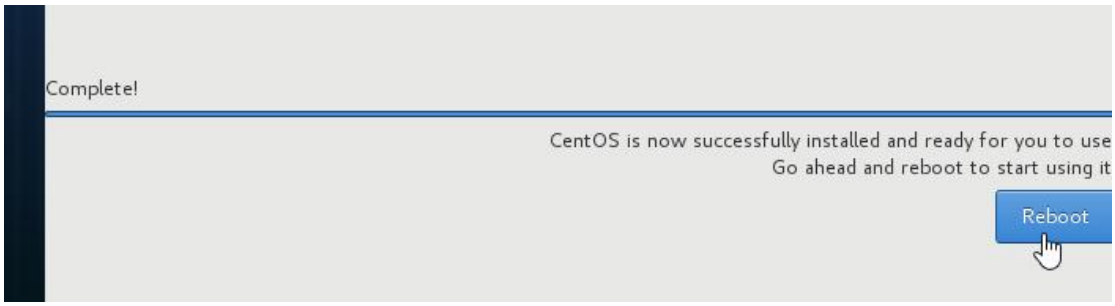
29.输入密码后，点击左上角的 Done，密码强度不够的话要点击 2 次



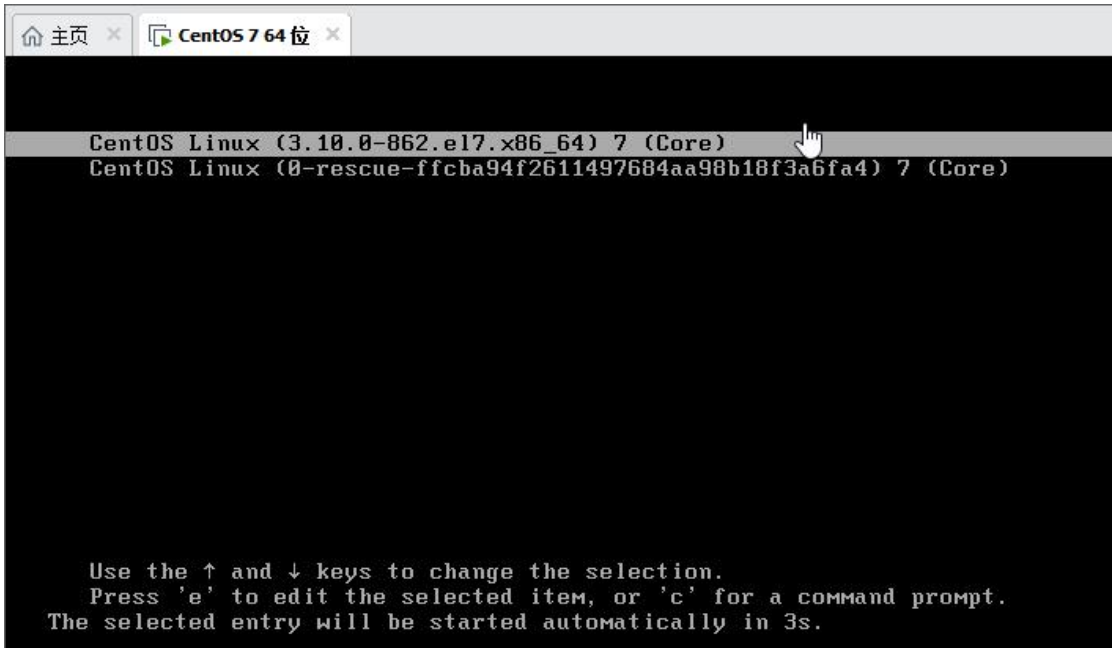
30.接下来就等待系统安装完成



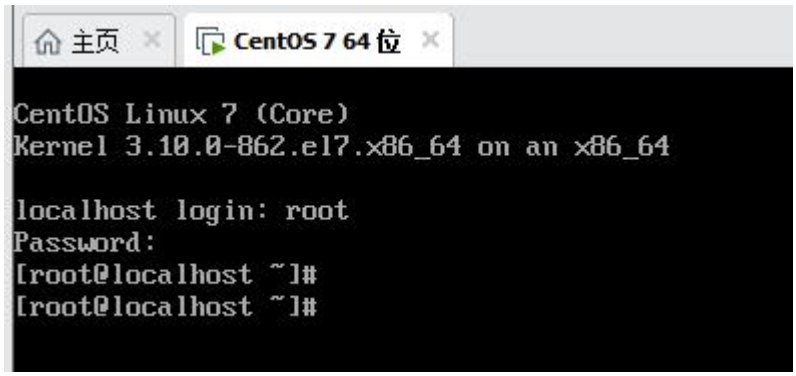
31.安装完毕，点击右下角的 Reboot 重启。



32.重启之后，有 2 行启动项可以选择，用默认的第一行，回车，启动系统



33.最后就进入系统了，输入用户名 root 密码为前面第 29 步骤创建的那个 root 密码，回车，进入系统。（输入密码时屏幕上是没有显示的，只管输入）



最小化安装的 CentOS 系统只有命令行界面，没有图形界面。输入用户名和密码后，初始的命令行界面显示的那一行字符（光标前的那个）叫作命令提示符：

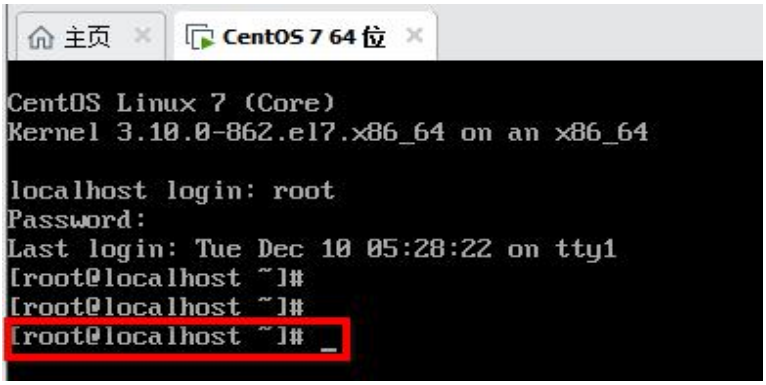
[root@localhost ~]#

在命令提示符后可以输入命令进行各种操作，具体怎么操作就是本教程接下来要教的。

要想关闭系统的话，输入 init　0 回车就行了。

```
[root@localhost ~]# init 0
```

一、初识 Linux 命令行

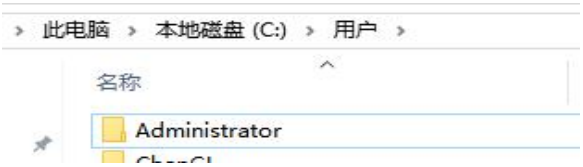


1.命令提示符各字段的含义

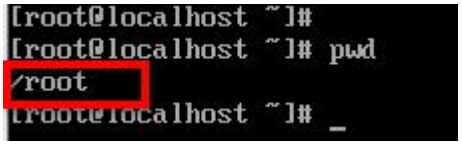
[root@localhost ~]#

root 表示当前登录的用户名
localhost 表示主机名称
~ 表示当前目录，家目录用波浪号~表示
#表示当前用户的权限级别，管理员用户的级别用#号表示，普通用户的级别用$号表示

2.什么是**家目录**，和 windows 系统里的 C:\Users\用户名 这个目录是一样的意思，登录到系统后默认所处的目录就是用户的家目录

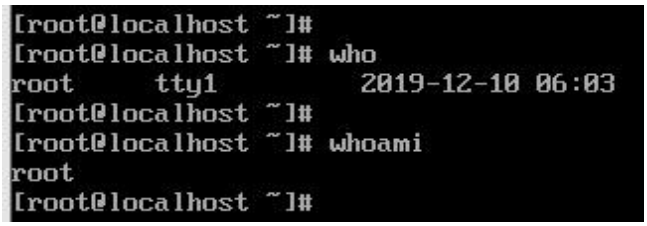


3.在命令行里输入 pwd 命令，回车，可以查看当前目录的完整路径



如上图，可见 root 用户的家目录就是/root
普通用户的家目录为 /home/用户名

4.输入命令 who 可以查看当前登录到系统的所有用户，
输入 whoami 可以知道自己是哪个用户（以后说输入某条命令，默认是要按下回车键的）



root　　tty1　　2019-12-10 06:03
用户名　终端　　登录时间

5.Linux 下输入命令前几个字母后，如果没有二义性，可以按下 Tab 键自动补全命令，如果有二义性，按下 Tab 键后会显示出所有匹配的命令

```
[root@localhost ~]# who
who      whoami
```

```
[root@localhost ~]# last
last     lastb    lastlog
```

6.可以按键盘上的箭头键的上下键↑↓，查看刚刚输入过的命令，当我们需要重复执行某个命令时，可以按上箭头找到那条命令，再回车就可以了，不必每次都输入一长串命令。

7.在输入命令时如果命令很长，突然决定不执行此命令时，没有必要一个字母一个字母地删除，可以按下 Ctrl 和 U 键，删除一整行。这叫快捷键，常用的 Linux 命令行快捷键如下表：

按键（组合键）	功能
Ctrl + W	删除光标前面的一个单词
Ctrl + K	删除光标当前位置至行末的所有字符
Ctrl + U	删除光标当前位置至行首的所有字符
Ctrl + A	移动光标至行首
Ctrl + E	移动光标至行尾
Ctrl + S	锁定命令行
Ctrl + Q	解锁命令行
Alt + B	光标移到前面一个单词的前面（前移一单词）
Alt + F	光标移到后面一个单词的前面（后移一单词）

8.命令行里的命令本质是什么，或者说 什么是命令？

Linux 下的命令分 2 种，一种是内部命令，一种是外部命令

*外部命令其实就是程序名称，命令的第一个单词是系统里自带有的程序或新安装的程序，之后的单词是传给这个程序的参数。这些程序可以是二进制程序，也可以是文本程序（shell 脚本等）

*内部命令就是命令行提供者（shell）自带的，是 shell 这个程序提供给用户的一些功能。

命令行里输入 type 加上目标命令，就可以查看目标命令的类型

比如查看 pwd 这个命令是何种类型的，可以输入命令：type pwd

```
[root@localhost ~]# type pwd
pwd is a shell builtin
```

可见 pwd 是 shell 内嵌的，自带的内部命令。

```
[root@localhost ~]# type who
who is /bin/who
```

而 who 命令是外部命令，是一个程序，该程序路径为/bin/who

*Linux 下的命令是区分大小写的

二、Console 字体及屏幕分辨率设置

无论是在虚拟机的 console 控制台还是真实的设备显示器里，命令行的字体太小了，有时还可能没有完整地使用整个屏幕，即显示的字符没有充满整个屏幕。

1.输入命令 ls /lib/kbd/consolefonts/ 可以查看系统自带的所有字体

```
[root@localhost ~]# ls /lib/kbd/consolefonts/
```

```
Cyr_a8x16.psfu.gz              koi8c-8x16.gz              LatArCyrHeb-16.psfu.gz
Cyr_a8x8.psfu.gz               koi8r-8x14.gz              LatArCyrHeb-16+.psfu.gz
cyr-sun16.psfu.gz              koi8r-8x16.gz              LatArCyrHeb-19.psfu.gz
default8x16.psfu.gz            koi8r-8x8.gz               latarcyrheb-sun16.psfu.gz
default8x9.psfu.gz             koi8r.8x8.psfu.gz          latarcyrheb-sun32.psfu.gz
drdos8x14.psfu.gz              koi8u_8x14.psfu.gz         LatGrkCyr-12x22.psfu.gz
drdos8x16.psfu.gz              koi8u_8x16.psfu.gz         LatGrkCyr-8x16.psfu.gz
drdos8x6.psfu.gz               koi8u_8x8.psfu.gz          LatKaCyrHeb-14.psfu.gz
drdos8x8.psfu.gz               lat0-08.psfu.gz            Mik_8x16.gz
ERRORS                         lat0-10.psfu.gz            partialfonts
eurlatgr.psfu.gz               lat0-12.psfu.gz            README.12x22
Goha-12.psfu.gz                lat0-14.psfu.gz            README.Arabic
Goha-14.psfu.gz                lat0-16.psfu.gz            README.cp1250
Goha-16.psfu.gz                lat0-sun16.psfu.gz         README.cybercafe
GohaClassic-12.psfu.gz         lat1-08.psfu.gz            README.Cyrillic
```

2.推荐使用 3 种不同大小的字体

lat2-16

sun12x22

latarcyrheb-sun32

3.使用命令 setfont 字体名 可以设置 console 字体，以上三种字体都试一下，选一个适合自己屏幕分辨率的就行

```
[root@localhost ~]# setfont sun12x22
[root@localhost ~]# ls
anaconda-ks.cfg
[root@localhost ~]#
[root@localhost ~]#
```

4.输入 setfont 可以恢复默认的字体

```
[root@localhost ~]# setfont
[root@localhost ~]#
[root@localhost ~]#
```

5.重启系统后，字体大小又恢复默认的了，需要在开机时就自动设置字体

这里涉及到 vi 的使用，不会用 vi 的先学习下一章节！

使用 vi 编辑/etc/rc.d/rc.local 文件

在文件末尾添加一行：

setfont sun12x22

保存

```
[root@localhost ~]# vi /etc/rc.d/rc.local
```

```
#!/bin/bash
# THIS FILE IS ADDED FOR COMPATIBILITY PURPOSES
#
# It is highly advisable to create own systemd services or udev
# to run scripts during boot instead of using this file.
#
# In contrast to previous versions due to parallel execution du
# this script will NOT be run after all other services.
#
# Please note that you must run 'chmod +x /etc/rc.d/rc.local' t
# that this script will be executed during boot.

touch /var/lock/subsys/local
setfont sun12x22
```

6.然后给该文件添加可执行权限

命令 chmod +x /etc/rc.d/rc.local

```
[root@localhost ~]# chmod +x /etc/rc.d/rc.local
[root@localhost ~]#
```

这样字体的设置就永久生效了。

7.设置屏幕分辨率

编辑/boot/grub2/grub.cfg 文件，找到/linux16 /boot/vmlinuz-3.10.0.xxxx...这一行

在该行的末尾添加 vga=0x342 （0x342 表示分辨率为 1152x864）

```
[root@localhost ~]# vi /boot/grub2/grub.cfg
```

```
        linux16 /vmlinuz-3.10.0-862.el7.x86_64 root=UUID=d2b1d839-8344-43cc-94c5-1be
38840c60 ro crashkernel=auto rhgb quiet LANG=en_US.UTF-8 vga=0x342
```

保存，重启系统即可

其他分辨率设置：

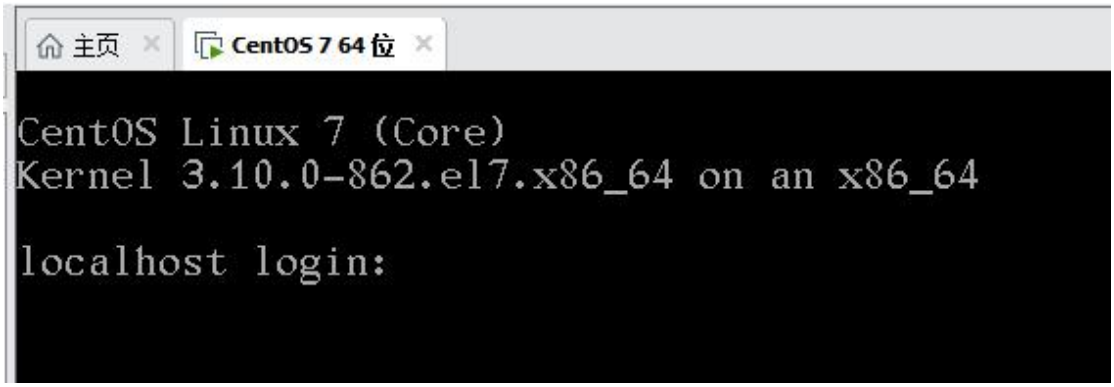
vga=0x367 → 1920x1080

vga=0x342 → 1152x864

vga=0x341 → 1024x768

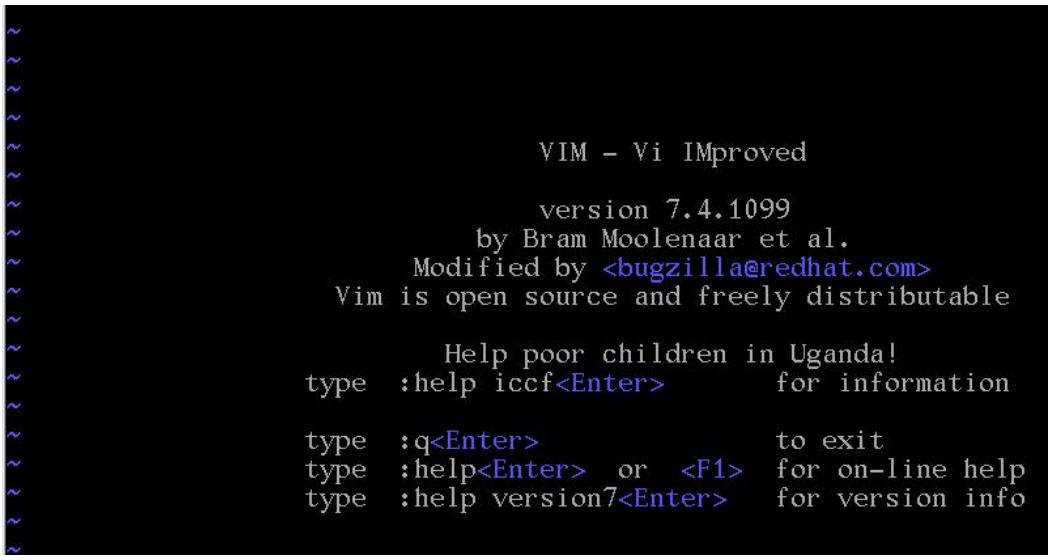
vga=0x340 → 800x600 （默认的大小）

重启系统后字体变成我们设置的大小了，console 分辨率也变高了。



三、Vi 文本编辑工具

vi 是 Linux 命令行下的一个文本编辑工具
在命令里下输入 vi 回车，进入 vi 工作界面



*命令行下没法用鼠标，所以一切对文本的编辑都是要依靠某些辅助的命令，上图中也有提示，比如按下 :q （冒号加字母 q，回车）可以退出 vi
我们先退出 vi（按一下 Esc 键，再输入英文的冒号加小写字母 q，回车）

*Vi 的用法有 2 种：
一种是直接输入 vi 回车，然后进行文本的编辑，再保存到文件中。
另一种用法是 vi 接目标文件名，再编辑。目标文件名可以是已经存在的文件，也可以是新创建的。

*最常用的用法是输入 vi 接目标文件名
比如我们想新建一个文本文件 test.txt，可以输入 vi test.txt

然后就是上图的界面，刚进入 vi 时，所处的模式为**普通模式**，普通模式下不能输入文本字

符，但可以进行其他的操作，比如移动光标，删除文本，插入文本，复制文本，查看目标字符串等。

*接下来我们想输入文本字符，可以在普通模式下按入字母 i（小写的 i），不用回车，按下字母 i 就可以输入文本了，能输入文本的模式称为 **插入模式**。如何退回普通模式呢，可以按下 Esc 键。

①从普通模式进入插入模式，除了可以按下小写的 i，也可以按下其他的字母，不同的字母含义是不同的：

普通模式下按下字母，不用回车	含义
i	从当前光标处之前插入文本
a	从当前光标处之后插入文本
I （大写字母 i）	将光标移至所在行 的行首并插入文本
A	将光标移至所在行 的行尾并插入文本
o （小写字母 o）	在光标所处行 的下面插入一行
O （大写字母 O）	在光标所处行 的上面插入一行

插入模式下可以正常地输入任何字符（除了 Esc 键）
按下 Esc 键退回 普通模式

②普通模式下的光标定位

普通模式下输入字符命令，不用回车	含义
G （大写的字母 G）	光标移至最后一行 的行首
gg （连续按 2 下小写字母 g）	光标移至第一行 的行首
5G （数字后接大写字母 G）	光标移至第 5 行 的行首
$ （按下 shift 再按数字 4）	光标移至当前行 的行尾
h j k l	对应箭头键的 ← ↓ ↑ →
Ctrl + F	下翻一页
Ctrl + B	上翻一页

③普通模式下删除文本

普通模式下输入字符命令，不用回车	含义
x （小写字母 x）	删除光标处的一个字符
dw （先后按下 2 个小写字母 d w）	删除光标处的一个单词
dd （连续按 2 下小写字母 d）	删除光标所处的一整行文字
d$ （先后按下字母 d 和 shift+4）	删除光标处 至 当前行 行尾的所有字符

④普通模式下复制文本

普通模式下输入字符命令，不用回车	含义
yy （连续按 2 下小写字母 y）	复制当前行
p （小写字母 p）	在光标所处行 下面插入刚刚复制的一行

⑤普通模式下撤销操作

普通模式下输入字符命令，不用回车	含义
u （小写字母 u）	取消上一次的操作，可按多次 u，进行多次撤销

上面 5 种操作都是只输入字符而不用回车，以下的操作要按下回车才执行

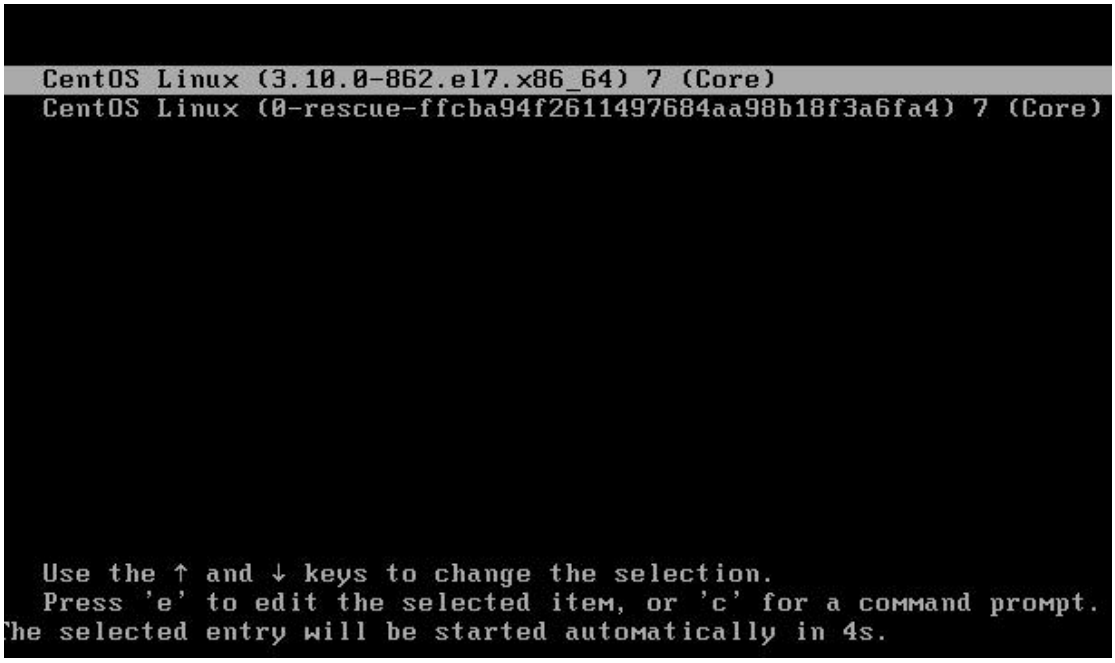
⑥普通模式进入命令模式，以/ ： ？开头就进入命令模式了

命令（输入之后要回车）	含义
/str （斜杠之后接上目标字符串）	向下查找字符串"str"，查找到一处后，按下字母 n 可以继续向下查找
?str （问号加目标字符串）	向上查找字符串"str"，查找到一处后，按下字母 N 可以继续向上查找
:1,$s/word1/word2/g	从第 1 行到最后一行，查找 word1，每查到一处就替换为 word2，不管 word1 是否为一个单词，且无提示
:1,$s/word1/word2/gc	从第 1 行到最后一行，查找 word1，每查到一处就提示是否替换为 word2，输入 y 确定替换，n 不替换
:set number	显示行号，行号本身不是文本文件的内容
:set nonumber	不显示行号
:set tabstop=4	设置显示制表符 tab 的空格数为 4 个
:r /文件名	将文件内容追加到当前文件末尾
:q （英文冒号加小写字母 q）	退出 vi，不保存本次编辑内容
:q!	加了感叹号，表示强制，强制退出，不保存
:wq	保存编辑内容，退出（即使文件没有被修改，也会保存一遍并改变文件修改时间）
:wq!	强制保存编辑内容，退出
:wq 文件名	如:wq xx.txt 表示将刚刚编辑的文本保存到 xx.txt 文件中，相当于另存为
:x	保存文件并退出。仅当文件被修改时才保存，并更新文件修改时间；否则不会更新文件修改时间。

四、系统关机重启操作

命令行里输入以下命令，回车	操作
poweroff	立即关机
init 0	立即关机
shutdown -h now	立即关机
shutdown -h +5	5 分钟后关机
halt	停机
shutdown -H now	停机
reboot	立即重启
init 6	立即重启
shutdown -r now	立即重启
shutdown -r +3	3 分钟后重启
shutdown -c	取消 关机或重启
exit	仅退出登录的会话，系统并没有关闭

系统启动前会出现如下界面让我们选择：（默认会选择第 1 个）



有 2 行可以选择，这 2 行表示 2 个启动项，第一行表示进入正常的系统，第 2 行表示进入救援维护模式，当系统出现故障时，可以进入救援模式进行排错。

五、文件目录操作

pwd　　//查看当前所处的目录

```
[root@localhost dhcp]# pwd
/etc/dhcp
```

特殊目录表示：

~　　用户的家目录

.　　当前目录，也可用 ./ 表示

..　　上一层目录（父目录），也可用 ../ 表示

① ls 命令，用于列出文件及文件夹（目录），默认是列出当前目录下的文件(夹)，不显示隐藏文件。（这里的 l 是字母 L 的小写格式）

```
ls  -选项  目标目录        //ls 的使用格式，目标目录缺省为当前目录
ls  -a                    //-a 表示列出所有文件，包括隐藏的
ls  -l                    //-l 表示以列表形式显示文件详细信息
ls  -d  /目录             //仅列出目录本身，查看目录本身的属性
ls  -h                    //以方便阅读的单位显示文件的大小
ll                        //ll 为 ls  -l 的别名
```

```
[root@localhost ~]# ls
anaconda-ks.cfg  backup.tar.gz  chfont.sh  fstab2  mymusic  test.txt
```

在 console 下列出文件时，不同的颜色表示不同的文件类型：

浅白：一般的文本字符文件

红色：压缩文件　　绿色：可执行文件（程序或脚本）

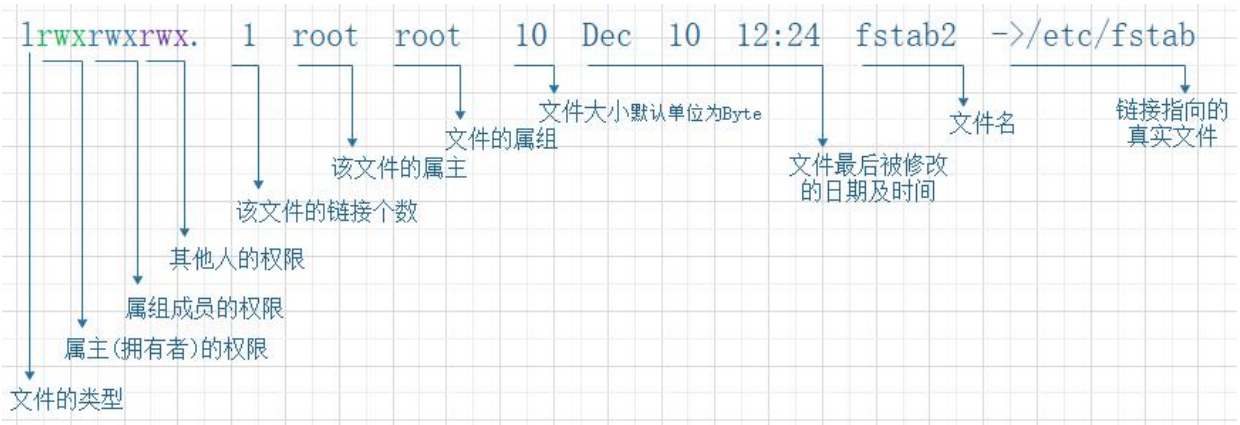
浅蓝：链接文件　　蓝色：目录文件

黄色：设备文件

②文件属性（用 ll 命令查看）

```
[root@localhost ~]# ll
total 16
-rw-------. 1 root root 1260 Dec 10 05:24 anaconda-ks.cfg
-rw-r--r--. 1 root root  114 Dec 10 12:22 backup.tar.gz
-rwxr-xr-x. 1 root root    5 Dec 10 12:21 chfont.sh
lrwxrwxrwx. 1 root root   10 Dec 10 12:24 fstab2 -> /etc/fstab
```

以 fstab2 这个文件为例：



文件类型：

- 表示普通文件
d 表示目录
/ 表示链接文件
b 表示块设备（磁盘等）
c 表示字符设备（键盘等）

文件的权限请看后面的第六章

③目录操作

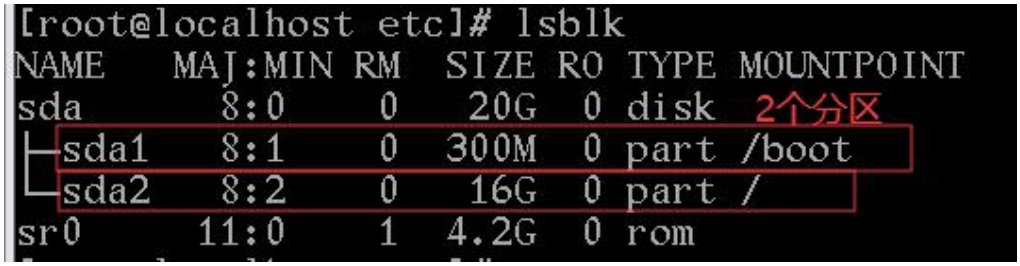
[root@localhost ~]# ~表示当前所处目录（家目录）

[root@localhost etc]# etc 表示当前所处目录（/etc）

*Linux 的文件系统只有一个根目录，即/（斜杠），相当于 windows 里的 C 盘 D 盘等根目录（windows 是可以有多个盘符的，每个盘符加上:\就是根目录，比如 C:\ D:\）

Linux 的根目录并不代表只有一块磁盘一个分区，/根目录只代表操作系统能够访问到的资源，根是系统级别的文件系统，一个磁盘分区可以挂载到/目录上，也可以挂载到/目录下的任何一个子目录。

*我们在安装系统时就创建了 2 个分区，一个（分区 1）挂载到了/boot 目录，另一个分区（分区 2）挂载到了/目录下，在不添加其他磁盘分区的情况下，我们对根目录/包括其子目录（除了 boot）做的操作都是在分区 2 上进行的，对根目录下的 boot 目录的操作都是在分区 1 上进行的。可以用命令 lsblk 查看分区对应的目录（sda 表示磁盘，sda1 为磁盘上的分区 1）




```
#pwd                    //查看当前所处的目录
#cd  /etc               //切换到/etc 目录，cd 表示切换目录（change directory）
#cd  ..                 //切换到上一层目录
#cd  -                  //切换到上一次所处的目录
#cd  ../a               //切换到上一层目录下的 a 目录里
#cd  ~                  //切换到当前用户的家目录

#mkdir  xxx             //在当前目录下创建一个名为 xxx 的目录
#mkdir  -p  /root/dir1/dir2          //递归创建目录及其子目录，当 dir1 不存在时就创建它

#rmdir  xxx             //删除一个空目录
#rmdir  dir1/dir2       //删除 dir1 下的 dir2
#rmdir  -p dir1/dir2    //递归删除，先删除 dir2，然后如果 dir1 下没有文件和其他
                        //目录了，就把 dir1 也删除了
#rm  -r  xxx            //删除非空目录，目录里的文件也会被删除，删除前会有提示，
                        //输入 y 表示确定删除，n 表示不删除
```

#rm -rf xxx //强制删除 xxx 目录及其包含的所有文件，不提示
#du -sh /root //显示/root 目录使用的容量（占用的大小），默认以 KB 为单位
//使用 h 参数后表示以方便阅读的单位显示

```
[root@localhost etc]# du -sh /etc
30M     /etc
```

④文件操作

#touch 文件名 //如果文件已存在，则更新该文件的最后修改时间，
//文件不存在则创建一个新的文件
#cat 文件名 //查看文件的内容
#cat 文件名 | more //当文件内容太多时，一屏显示不下，可以用 more 进行分页
//查看（没有进度提示）分页查看时，按下空格键可以向后翻一页，按下 P 键退出查看。
#more 文件名 //分页查看文件内容（有进度提示）
#cat 文件名 | grep 'str' //只查看含有 str 字符串的行
#cat 文件名 | grep -v '#' //不看含有#字符的行
#grep 'xx' 文件名 //在文件中查找含有 xx 字符的行
#grep -v -'xx' 文件名 //在文件中查找不含有 xx 字符的行

#rm 文件名 //删除文件，有提示
#cp 源文件 目的文件 //复制文件
#cp -r 源目录 目的目录 //递归复制目录及其所有子文件（子目录）
#mv 源文件 目的文件 //移动文件，也可在当前目录下移动，起到重命名的作用

#cat 文件名 | wc -l //查看文件有多少行

```
[root@localhost etc]# cat /etc/fstab |wc -l
10
```

#head -n 5 文件名 //只查看文件的前 5 行
#tail -n 5 文件名 //只查看文件的最后 5 行

⑤查找文件 find

根据文件名查找
#find 文件名 //在当前目录下查找文件
#find 目录名 -name xxx //在指定目录下查找所有名为 xxx 的文件（区分大小写）
#find 目录名 -iname xxx //在指定目录下查找所有名为 xxx 的文件（不区分大小写）
#find 目录名 -iname *xxx* //在指定目录下查找所有名为 xxx 的文件，*为模糊匹配
#find 目录名 -user coflee //在指定目录下查找所有属于 coflee 用户的文件
根据时间查找（+5 表示在指定时间之前，-5 表示在指定时间之内）
#find 目录名 -mtime -3 //在指定目录下查找 3 天之内被修改过的文件
#find 目录名 -ctime +2 //在指定目录下查找 2 天之前创建的文件（包括复制的）
#find 目录名 -mmin -5 //在指定目录下查找 5 分钟之内被修改过的文件
#find 目录名 -mmin +5 //在指定目录下查找 5 分钟之前被修改过的文件

根据文件类型：

-type f //表示普通文件
-type b //表示块设备文件
-type c //表示字符设备文件
-type l //表示链接文件
-type d //表示目录

根据文件大小：

-size -50M //表示 50MB 以内的
-size +20M //表示大于 20MB 的
-size +10M -size -15M //可叠加使用表示范围，10~15MB 之内的

以上查找文件的各参数都可以**组合使用**。

比如：find /dev -mmin -15 -size +1M -type f
//表示在/dev 目录下查找 15 分钟之内被修改过的大于 1MB 的普通文件

六、Linux 文件权限

①文件权限说明

Linux 的文件权限有 3 个：读、写、执行（read、write、execute）（暂时说 3 个）

Linux 是多用户系统，在一个系统上可以允许多个用户登录并使用系统资源，所以一个文件的权限根据用户的类型而分成 3 类，一类是对文件拥有者而言的，一类是对文件所在组的组员的，一类是对其他人的。在查看文件的属性时，前面的那九个字符（rwx-字样的）就表示文件的权限：

这 9 个字符分成 3 组（每组 3 个字符）

第 1 组表示文件拥有者（属主 user）的权限

第 2 组表示文件所在组的组员（group）拥有的权限

第 3 组表示其他人（other）对该文件的权限

rwx 表示读写执行三个权限都有

r-x 其中的-横杠表示没有相应的权限，即-横杠所在的位置本来是写，所以用-横杠代替时表示没有写权限

---表示读写执行 3 个权限都没有

例：

rwxr-x--- 表示文件拥有者（属主）对文件有**读写执行**的权限，文件属组成员对文件有**读**和**执行**的权限，其他人对该文件**没有**权限

②用数字表示权限

r　　读权限的值为 4

w　　写权限的值为 2

x　　执行权限的值为 1

-　　没有相应的权限的值为 0

每一组权限可以用其 3 个权限类型的值之合来表示，比如 wrx 可用 7 表示，r-x 用 6 表示，r--用 4 表示，---用 0 表示等。

这样用 9 个字母表示的文件权限可以用 3 个数字表示：

rwxr-x---可以记为 760

③修改文件权限

```
#chmod   775   文件名          //把文件的权限改为 775（rwxrwxr-x）
#chmod   +x   文件名          //增加文件的 x 权限，即每一组都加上 x 执行权限
#chmod   g+w   文件名        //增加组成员的 w 写权限，g 代表文件属组 group
#chmod   u+x   文件名        //增加文件拥有者的 x 执行权限，u 代表拥有者 user
#chmod   o-w   文件名        //减去其他人的 w 写权限，o 代表其他人 other

#chown   用户名   文件名     //更改文件的拥有者
#chown   cof   test.txt      //把 test.txt 文件的属主改为 cof 用户
#chgrp   root   test.txt     //把 test.txt 文件的属组改为 root 组
```

④文件生成掩码

我们创建一个新的文件时，默认的权限就是 文件的生成权限

```
[root@localhost ~]# touch aaa
[root@localhost ~]# ll
total 16
-rw-r--r--. 1 root root     0 Dec 11 06:13 aaa
```

如上图，我们创建文件时默认生成的权限是 rw-r--r--（644）

少了哪些权限呢，少了 133（属主的 x，属组的 wx，其他人的 wx）

生成文件权限时默认缺少的权限值就叫 文件生成掩码

umask　　//查看文件生成掩码

```
[root@localhost ~]# umask
0022
```

为什么是 0022 而不是 0133 呢，因为我们刚刚创建的文件默认就是普通文件，而不是可执行文件，用编译工具生成的可执行文件的默认权限就是掩码 0022 对应的 rwxr-xr-x

umask　033　　//设置文件生成掩码（对应权限 744：rwxr--r--）

⑤特殊权限

刚刚查看文件生成掩码时，我们发现显示的是 0022，四位数字而不是我们想像中的 3 位数字，这是怎么回事？

因为文件的权限本来就是 4 个，之前说 3 个是大家常认为的。其实是 4 个，还有一个特殊权限，特殊权限用得少所以很少有人说起。4 位数字表示时最左边的那个就表示特殊权限。

特殊权限：

SUID 权限 表示无论是哪个用户来执行文件，都是以该文件的属主的身份去执行

SGID 权限 表示无论是哪个用户来执行文件，都是以该文件的属组成员的身份去执行

Sticky-bit 权限 表示仅允许文件的属主执行删除、移动等操作

这 3 个权限在用字母表示时不会额外占一个位置，只会共用 user, group, other 这三组的 x 权限的位置。

比如：

原来的每组上的 x 权限位置被特殊权限给占用，那么我们怎么判断该组权限上的 x 位原来的时候有没有可执行权限呢？

看该特殊位字母的大小写，如果原来有可执行权限，则置为特殊权限时，显示为小写字母，如果原来没有可执行权限，置为特殊权限时显示为大写字母。

特殊权限的数字表示：

suid 值为 4　　sgid 值为 2　　sticky-bit 值为 1

特殊权限的设置

#chmod 4755 文件名 //添加 suid 权限，（其他 3 个权限为 755）
#chmod u+s 文件名 //添加 suid 权限，以字母的形式添加
#chmod g+s 文件名 //添加 sgid 权限
#chmod o+t 文件名 //添加 sticky-bit 权限

#chmod 0755 文件名 //设置权限为 0755，无特殊权限
0755 和 755 是一样的，权限在用数字设置时，最先从右边开始赋值，左边没有的时候默认为 0，所以 chmod 5 的时候，默认就是 0005

七、系统基本信息

命令提示符各字段的含义

[root@localhost ~]#

root 表示当前登录的用户名
localhost 表示主机名称
~ 表示当前目录，家目录用波浪号~表示

①**主机名称**是保存在/etc/hostname 文件里的，可以编辑该文件，设置成我们要设置的主机名，重启系统才生效。

#hostname　　//查看主机名（全称）

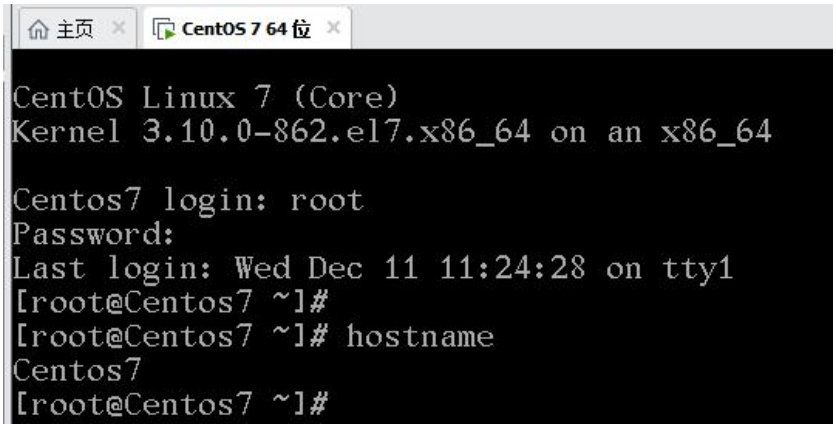
```
[root@localhost ~]# hostname
localhost.localdomain
[root@localhost ~]#
[root@localhost ~]# cat /etc/hostname
localhost.localdomain
```

#vi　/etc/hostname　　//编辑主机名配置文件，内容为主机名

```
[root@localhost ~]# vi /etc/hostname
```

```
Centos7_
~
```

保存，重启系统后主机名称修改成功



②**查看 Linux 发行版本**信息

uname　-r　　//查看 Linux 内核版本

uname　-a　　//查看详细的版本信息

```
[root@Centos7 ~]# uname -r
3.10.0-862.el7.x86_64
[root@Centos7 ~]# uname -a
Linux Centos7 3.10.0-862.el7.x86_64 #1 SMP Fri Apr 20 16:44:24 UTC 2018 x
 x86_64 GNU/Linux
[root@Centos7 ~]#
```

#cat /etc/centos-release //查看 centos 发行版本号

#cat /etc/redhat-release //查看 redhat 发行版本号（因为 centos 系统是由 redhat 释出的源代码编译而来的，所以有些 redhat 上的信息，在 centos 上也保留了）

```
[root@Centos7 ~]# cat /etc/centos-release
CentOS Linux release 7.5.1804 (Core)
[root@Centos7 ~]#
[root@Centos7 ~]# cat /etc/redhat-release
CentOS Linux release 7.5.1804 (Core)
```

③CPU 及内存信息

#lscpu //查看 cpu 信息

#cat /proc/cpuinfo //查看 cpu 信息

```
[root@Centos7 ~]# lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                1
```

```
[root@Centos7 ~]# cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 158
model name      : Intel(R) Core(TM) i5-8400 CPU @ 2.80GHz
stepping        : 10
microcode       : 0x96
cpu MHz         : 2807.998
```

#getconf LONG_BIT //查看 cpu 位宽

```
[root@Centos7 ~]# getconf LONG_BIT
64
```

#free -m //查看内存资源使用情况，total 表示总的大小，

//-m 表示以 MB 为单位显示

```
[root@Centos7 ~]# free -m
              total        used        free      shared  buff/cache   available
Mem:            974         117         719           7         137         697
Swap:             0           0           0
```

④查看系统默认语言环境

#echo $LANG //查看系统使用的语言及字符编码

```
[root@Centos7 ~]# echo $LANG
en_US.UTF-8
```

⑤查看磁盘及分区情况

#lsblk //查看磁盘及分区情况

```
[root@Centos7 ~]# lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda      8:0    0   20G  0 disk
├─sda1   8:1    0  300M  0 part /boot
└─sda2   8:2    0   16G  0 part /
sr0     11:0    1  4.2G  0 rom
```

上图中，sda 为硬盘，sr0 为光盘，sda 磁盘划分了 2 个分区（sda1 和 sda2）sda1 分区挂载到了/boot 目录下，sda2 分区挂载到了/根目录下。

⑥查看系统时间

#date　　//查看当前的时间

#timedatectl　　//查看详细的时间信息

```
[root@Centos7 ~]# date
Wed Dec 11 11:46:36 EST 2019
[root@Centos7 ~]# timedatectl
      Local time: Wed 2019-12-11 11:46:46 EST
  Universal time: Wed 2019-12-11 16:46:46 UTC
        RTC time: Wed 2019-12-11 16:46:46
       Time zone: America/New_York (EST, -0500)
     NTP enabled: yes
```

#uptime　　//查看系统运行时长，显示的第一个时间为当前时间，

//UP 之后的时间为开机运行时长

```
[root@Centos7 ~]# uptime
 11:48:37 up 23 min,  1 user,  load average: 0.00, 0.01, 0.02
```

⑦系统时间设置

#timedatectl　status　　//先查看时间基本信息，默认情况下是使用 NTP 服务的

#timedatectl　list-timezones　　//列出所有预设的时区

#timedatectl　set-timezone　Asia/Shanghai　　//设置时区为上海时区

#timedatectl　set-ntp　false　　//不使用 ntp，需要用的话把 false 改为 true

#timedatectl　set-time　'2019-12-11　17:01:00'　　//设置日期及时间（不用 ntp 时）

#timedatectl　set-local-rtc　1　　//将硬件时钟设为本地时间（这里指上海时区的）或者

#timedatectl　set-local-rtc　0　　//将硬件时钟设为 RTC 世界时间

配置完成后查看一下

```
[root@Centos7 ~]# timedatectl status
      Local time: Wed 2019-12-11 17:04:49 CST
  Universal time: Wed 2019-12-11 09:04:49 UTC
        RTC time: Wed 2019-12-11 09:04:50
       Time zone: Asia/Shanghai (CST, +0800)
     NTP enabled: no
NTP synchronized: no
 RTC in local TZ: no
```

⑧修改命令行提示符

命令行提示符是由 shell 程序提供的，所以也叫 shell 提示符

默认是 [用户名@主机名 目录]# 的形式，没有色彩，可以自定义样式

#echo $PS1 //查看默认的 shell 提示符样式

```
[root@Centos7 ~]# echo $PS1
[\u@\h \W]\$
```

\u 表示用户名

\h 表示主机名（如果是 FQDN 名称，则主显示最前面的一级）

\W 表示当前目录（仅显示最后一级目录）

\$ 表示权限提示符

其他格式

\H 表示主机名，FQDN 全称

\w 表示当前目录（绝对路径）

\t 表示当前时间，24 小时制，HH:MM:SS

\A 表示当前时间，24 小时制，HH:MM

\# 表示本次登录以来下达的第 几个命令

\斜杠开头的为格式，在显示的时候会转为相应的信息，没有\斜杠开头的字符都是会直接在命令行里显示出来的。

颜色设置

\[\e[前景色编号;背景色编号 m\] //设置颜色的格式，放在每个字符前面

默认为 \[\e[m\] //设置完颜色后，要接上这个字段恢复默认的颜色

颜色对应的编号：

颜色	前景色编号	背景色编号
黑	30	40
红	31	41
绿	32	42
黄	33	43
蓝	34	44
紫红	35	45
青蓝	36	46
白	37	47

需要自定义 shell 提示符，只要在/etc/bashrc 文件末尾添加如下一行即可：

PS1='[\自定义的格式]..' //注意不要漏了单引号'

保存，重新登录就能看到效果。

自定义 shell 提示符示例 1：（仅自定义颜色）

PS1='[\[\e[33;40m\]\u\[\e[m\]@\[\e[36;40m\]\h \[\e[m\]\W]\$'

```
[root@Centos7 ~]# vi /etc/bashrc
```

```
# vim:ts=4:sw=4
PS1='[\[\e[33;40m\]\u\[\e[m\]@\[\e[36;40m\]\h \[\e[m\]\W]\$'
-- INSERT --
```

保存，退出登录（使用命令 exit），再重新登录就看到效果了

```
Centos7 login: root
Password:
Last login: Wed Dec 11 17:28:38 on tty1
[root@Centos7 ~]#
[root@Centos7 ~]#
[root@Centos7 ~]#_
```

自定义 shell 提示符示例 2：

PS1='[\[\e[33;40m\]~@_@~ \[\e[35;40m\]\t \[\e[m\]>>\[\e[32;40m\]\h\[\e[m\]]\$'

```
# vim:ts=4:sw=4
PS1='[\[\e[33;40m\]~@_@~ \[\e[35;40m\]\t \[\e[m\]>>\[\e[32;40m\]\h\[\e[m\]]\$
-- INSERT --
```

```
Centos7 login: root
Password:
Last login: Wed Dec 11 17:35:54 on tty1
[~@_@~ 17:36:22 >>Centos7]#
[~@_@~ 17:36:23 >>Centos7]#
[~@_@~ 17:36:27 >>Centos7]#
```

八、别名和链接的创建

①命令别名

有时命令太长不好记，可以用一个简写单词 来表示，这个简写的单词就是命令别名

#type 别名 //查看别名对应的真实的命令

比如 Centos 中默认创建有一个命令别名 ll

#type ll //查看 ll 命令别名对应的真实的命令

```
[root@Centos7 ~]#type ll
ll is aliased to 'ls -l --color=auto'
```

可见 ll 对应的真实命令是 ls -l --color=auto

#which 别名 //查看别名对应的真实的命令

```
[root@Centos7 ~]#alias ls
alias ls='ls --color=auto'
```

可见 ls 本身也是一个别名，ls 是 ls --color=auto 的别名

创建我们自定义的命令别名

#alias 别名='命令' //命令行下临时地添加一条命令别名，重启系统后失效

有时输入 ll 或 ls 查看文件时，默认是以字节为单位显示文件大小的，要以方便阅读的单位显示，需要加上参数-h，为了方便还可以将 ll 设置为 ls -lh 的别名

#alias ll='ll -lh'

```
[root@Centos7 ~]#alias ll='ls -lh'
[root@Centos7 ~]#
[root@Centos7 ~]#ll
total 16K
-rwxr-xr-x. 1 root root    0 Dec 11  2019 aaa
-rw-------. 1 root root 1.3K Dec 10 18:24 anaconda-ks.cfg
```

要使命令别名永久生效，可以将别名写进 shell 的配置文件里

~/.bashrc 该文件的配置仅对用户自己生效

/etc/bashrc 该文件的设置对全局有效

#vi ~/.bashrc //编辑个人的 shell 配置文件，在末尾添加自定义的命令别名

```
[root@Centos7 ~]#vi ~/.bashrc
```

```
fi
alias ll='ls -lh'
~
```

保存，

#source ~/.bashrc //执行此配置文件，使别名生效

```
[root@Centos7 ~]#source ~/.bashrc
```

*当别名和真实的命令名称相同时，默认是执行别名，比如 ls

默认执行的是 ls 这个别名，相当于执行 ls --color=auto

那么要指定执行真实的命令本身，怎么办，可以在命令前面加上\反斜杠

看出区别了吧，ls 真实的命令本身是不输出彩色的，ls 别名（ls --color=auto）才输出彩色

作者本人就喜欢把一些危险操作命令设置成别名，这样可以防止其他人误操作引起的严重后果。当需要输入真实的命令本身时，在命令前面加上\反斜杠就行了，其他人一般都不知道。比如：

```
alias   rm='echo Warning: you donot have permission to rm '
```

这样无论别人用 rm 命令删除什么文件，都会提示:

Warning: you donot have permission to rm 文件名

想要删除文件时需要加上\反斜杠。

②文件链接

文件的链接就相当于 Windows 里的快捷方式，Windows 桌面上的图标文件都是可执行程序的链接

链接有 2 种：

软链接：也叫符号链接，当原文件被删除时，链接失效

硬链接：当原文件被删除时，链接仍有效，数据未删除，只是删除原文件的名称

创建链接

```
#ln   目标文件   链接名称                  //创建软链接
#ln   -s   目标文件   链接名称             //创建硬链接
```

九、系统运行级别

系统运行级别就是系统是以何种模式运行的，比如以命令行模式运行，以桌面模式运行等。关机和重启也算是运行模式

Linux 的系统默认有 7 个运行级别 runlevel 0 到 runlevel 7，只不过 Centos7 开始不用 runlevel 表示运行级别，换了一套方案，叫作 target。

Centos7 系统运行目标有 5 个，对应以前的 7 个 runlevel

级别	Systemd 目标	SysVinit 的 runlevel	含义
0	poweroff.target	runlevel0.target	关机
1	rescure.target	runlevel1.target	救援模式（单用户）
2	multi-user.target	runlevel2.target	命令行界面（多用户）
3		runlevel3.target	
4		runlevel4.target	
5	graphical.target	runlevel5.target	图形界面
6	reboot.target	runlevel6.target	重启

#systemctl get-default //查看系统默认的运行目标

#runlevel //查看系统默认的运行级别

```
[root@Centos7 ~]#systemctl get-default
multi-user.target
[root@Centos7 ~]#runlevel
N 3
```

上图显示默认是以命令行模式启动

*如果在安装系统时，安装的是最小化版本 minimal 版本，则不能以图形界面的运行级别启动。如果安装的是图形界面版本，则可以用命令行或图形界面启动。

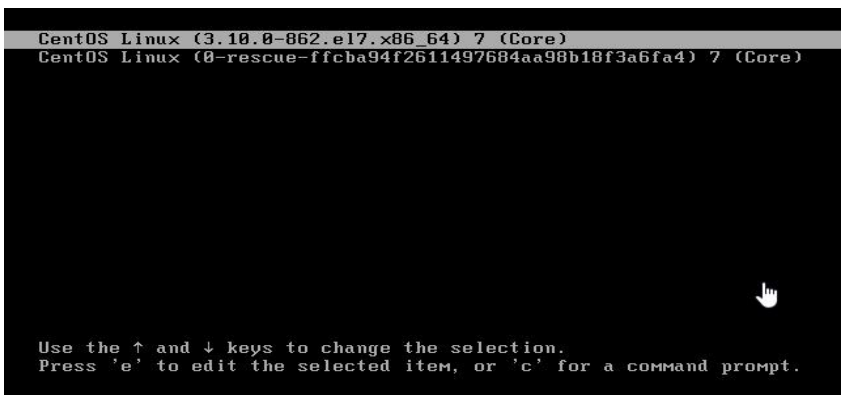
#systemctl set-default 目标 target //设置默认的启动目标，重启生效

```
[root@Centos7 ~]#systemctl set-default multi-user.target
Removed symlink /etc/systemd/system/default.target.
Created symlink from /etc/systemd/system/default.target to /usr/l
lti-user.target.
```

#init 0~6 //立即进入目标级别，比如 init 6 立即重启，init 0 立即关机，

//init 5 立即重启进入 graphical.target 图形界面

```
[root@Centos7 ~]#init 6
```



十、Systemd 初始化服务

*程序的每一个运行实例都是一个进程，Linux 系统上运行着普通的进程和守护进程。
守护进程就是为用户提供服务的进程，主要有 2 类：
系统守护进程：为本地用户提供服务的，提供给用户的基本功能
网络守护进程：为远程的网络用户提供服务的，如 web 服务，ssh 服务，ftp 服务等。

*系统初始化进程是一个特殊的守护进程，其 pid 为 1,它用于管理系统上的所有守护进程
比如要开启哪些守护进程，关闭哪些守护进程都是由系统初始化进程去完成的。

*Linux 用过的系统初始化进程有：
SysVinit　　用在 RHEL/Centos5 及以前的版本上
Upstart　　用在 RHEL/Centos6
Systemd　　用在 RHEL/Centos7

*在 Centos7 中，系统启动后，先由内核去启动 systemd 这个进程，再由 systemd 去启动其他的进程，systemd 这个服务提供了一个命令行工具名为 systemctl
用 systemctl 这个工具可以管理基于 Systemd 的服务

#systemctl　--type　service　　//显示当前运行的所有服务

```
[root@Centos7 ~]#systemctl --type service
  UNIT                        LOAD   ACTIVE SUB     DESCRIPTION
  auditd.service              loaded active running Security Aud
  crond.service               loaded active running Command Sche
  dbus.service                loaded active running D-Bus System
  firewalld.service           loaded active running firewalld -
  getty@tty1.service          loaded active running Getty on tty
● kdump.service               loaded failed failed  Crash recove
  kmod-static-nodes.service   loaded active exited  Create list
  network.service             loaded active exited  LSB: Bring u
```

#systemctl　--type　service　--all　　//显示系统中所有的服务，包括未运行的
#systemctl　--type　service　--failed　　//显示已加载但处于 faiiled 状态的服务

```
[root@Centos7 ~]#systemctl --type service --failed
  UNIT           LOAD   ACTIVE SUB    DESCRIPTION
● kdump.service loaded failed failed Crash recovery kernel arming
```

#systemctl　status　服务名　　//查看目标服务的当前状态

```
[root@Centos7 ~]#systemctl status network
● network.service - LSB: Bring up/down networking
   Loaded: loaded (/etc/rc.d/init.d/network; bad;
   Active: active (exited) since Thu 2019-12-12 15
     Docs: man:systemd-sysv-generator(8)
```

#systemctl　start　服务名　　//启动目标服务
#systemctl　restart　服务名　　//重启服务
#systemctl　try-restart　服务名　　//仅当服务在运行时，才重启
#systemctl　stop　服务名　　//关闭服务

#systemctl reload 服务名 //重新加载服务的配置

#systemctl list-unit-files //查看所有服务是否为开机自启

```
[root@Centos7 ~]#systemctl list-unit-files
UNIT FILE                                   STATE
proc-sys-fs-binfmt_misc.automount           static
dev-hugepages.mount                         static
dev-mqueue.mount                            static
proc-sys-fs-binfmt_misc.mount               static
sys-fs-fuse-connections.mount               static
sys-kernel-config.mount                     static
sys-kernel-debug.mount                      static
tmp.mount                                   disabled
brandbot.path                               disabled
systemd-ask-password-console.path           static
systemd-ask-password-plymouth.path          static
systemd-ask-password-wall.path              static
session-1.scope                             static
session-2.scope                             static
auditd.service                              enabled
autovt@.service                             enabled
```

STATE 状态说明：

static 表示必须随开机启动，用户不能设置的

disabled 表示没有随开机启动，用户可以设置为 enabled

enabled 表示随开机启动，用户可以设置为 disabled

masked 表示隐藏的，不能直接设置它

#systemctl enable 服务名 //将目标服务设置为 随开机型启动

#systemctl disable 服务名 //将目标服务设置为不随开机启动

十一、用户和组操作

①用户操作

#useradd 用户名 //创建一个用户

```
[root@Centos7 ~]#useradd coflee
```

#id 用户名 //查看该用户的基本信息

```
[root@Centos7 ~]#id coflee
uid=1000(coflee) gid=1000(coflee) groups=1000(coflee)
```

uid 表示用户的唯一标识，gid 表示该用户当前的属组 id，groups 表示用户的属组

新创建用户时，默认是会创建一个和用户名同名的组，这个组叫做 私有组，新创建的用户默认加入和用户名同名的私有组中。如果想将用户再加入其他组中，那么其他组对于该用户而言，就是附加组。一个用户可以加入多个组中。

#usermod -G 组名 用户名 //将用户加入附加组中

```
[root@Centos7 ~]#usermod -G root coflee
[root@Centos7 ~]#
[root@Centos7 ~]#id coflee
uid=1000(coflee) gid=1000(coflee) groups=1000(coflee),0(root)
```

#useradd -u 1004 用户名 //创建用户同时指定其 uid 为 1004,uid 可自定义

旧版的系统中，普通用户的默认 uid 是从 500 开始编号的，Centos7 中是从 1000 开始编号

#usermod -u 1009 用户名 //修改用户的 uid 为 1009

```
[root@Centos7 ~]#usermod -u 1009 coflee
[root@Centos7 ~]#id coflee
uid=1009(coflee) gid=1000(coflee) groups=1000(coflee),0(root
```

#passwd -S 用户名 //查看该用户的密码状态

```
[root@Centos7 ~]#passwd -S coflee
coflee LK 2019-12-12 0 99999 7 -1 (Password locked.)
```

密码状态说明：

LK 表示锁定，用户不能登录系统

PS 表示密码已设置，可登录系统

NP 表示无密码（可本地登录，不能远程登录）

#passwd 用户名 //给用户设置密码（输入密码时是没有显示的，只管输入）

```
[root@Centos7 ~]#passwd coflee
Changing password for user coflee.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

#passwd -l 用户名 //锁定用户，用户被锁定后不能登录系统，密码状态为 LK
#passwd -u 用户名 //解除锁定，可登录
#passwd -d 用户名 //清除用户的口令，用户的密码状态变成 NP

#usermod -l Licof coflee //将 coflee 用户 更名为 Licof，其家目录不变
//coflee 之前家目录为/home/coflee，更名为 Licof
//后，再登录系统，其家目录仍然为/home/coflee

如何查看用户的家目录呢？
在/etc/passwd 文件里查看，如下图，可见用户更名后，其家目录没有变

```
[[root@Centos7 ~]#cat /etc/passwd
[root:x:0:0:root:/root:/bin/bash
[bin:x:1:1:bin:/bin:/sbin/nologin
[daemon:x:2:2:daemon:/sbin:/sbin/nologin
 adm:x:3:4:adm:/var/adm:/sbin/nologin
 lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
 sync:x:5:0:sync:/sbin:/bin/sync
 shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
 halt:x:7:0:halt:/sbin:/sbin/halt
 mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
 operator:x:11:0:operator:/root:/sbin/nologin
 games:x:12:100:games:/usr/games:/sbin/nologin
 ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
 nobody:x:99:99:Nobody:/:/sbin/nologin
 systemd-network:x:192:192:systemd Network Management:/:/sbin/nolo
 dbus:x:81:81:System message bus:/:/sbin/nologin
 polkitd:x:999:998:User for polkitd:/:/sbin/nologin
 sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologi
 postfix:x:89:89::/var/spool/postfix:/sbin/nologin
 chrony:x:998:996::/var/lib/chrony:/sbin/nologin
 Licof:x:1009:1000:/home/coflee:/bin/bash
```

#usermod -d /home/Licof Licof //将 Licof 用户的家目录指定为/home/Licof
//设置时，先为该用户创建/home/Licof 目录，然后将该目录的属主改为 Licof，
且该用户要先退出登录，才能重新指定其家目录为/home/Licof。
用户再次登录系统后，就会进入/home/Licof 的目录

```
[root@Centos7 ~]#mkdir /home/Licof
[root@Centos7 ~]#ll -d /home/Licof
drwxr-xr-x. 2 root root 6 Dec 12 16:04 /home/Licof
```

```
[root@Centos7 ~]#chown Licof /home/Licof
[root@Centos7 ~]#
[root@Centos7 ~]#ll -d /home/Licof
drwxr-xr-x. 2 Licof root 6 Dec 12 16:04 /home/Licof
```

#usermod -d /home/xxx -m xxx //指定 xxx 用户的家目录为/home/xxx，且把原来的家目录里的文件移到新的家目录中

#userdel xxx //删除用户，保留其家目录
#userdel -r xxx //删除用户且不保留其家目录

#useradd -d /www -M 用户名 //创建用户，指定其登录的默认目录为/www，
//不为该用户创建家目录
#useradd -s /sbin/nologin 用户名 //创建用户时，指定其登录的 shell 为
/sbin/nologin，表示没有 shell，即不能登录系统

②组操作

#groupadd -g 3000 组名 //创建一个组并指定其组 id 为 3000
#groupmod -n xxx coflee //将 coflee 组更名为 xxx
#groupdel 组名 //删除该组

③用户口令时效

#chage -d 0 用户名 //该用户下次登录系统时，必须更改密码，
// -d 表示允许不更改密码的天数，0 表示下次登录就得更改
#chage -m 2 -M 30 -W 3 用户名 //该用户在将来的 2 至 30 天内必须更改
//密码，且在 30 天到期前 3 天会提醒
#chage -E 2019-12-30 用户名 //指定用户帐号被锁定的日期

④用户及组信息查看

#who //查看系统当前登录的用户

```
[root@Centos7 ~]#who
Licof    tty1         2019-12-12 16:20
root     pts/0        2019-12-12 15:17 (192.168.0.118)
```

#whoami //查看自己是哪个用户

```
[root@Centos7 ~]#whoami
root
```

#id 用户名 //查看指定用户的信息，不指定用户名时，默认是查看自己

```
[root@Centos7 ~]#id
uid=0(root) gid=0(root) groups=0(root) context=un
ned t:s0-s0:c0.c1023
```

#groups 用户名 //查看指定用户的属组信息，
//不指定用户名时，默认是查看自己的当前属组（主组）

```
[root@Centos7 ~]#groups
root
[root@Centos7 ~]#
[root@Centos7 ~]#groups Licof
Licof : coflee root
```

一个用户可以属于多个组，但在同一时刻，只能属于某个具体的属组，叫做主组

#newgrp 组名 //切换当前用户的主组

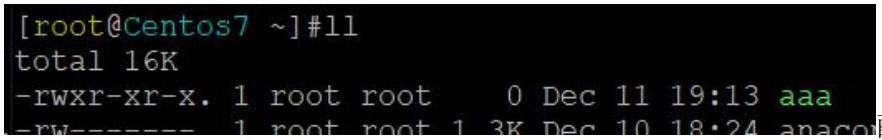
#su 用户名 //切换用户
#su - 用户名 //切换用户并进入其家目录

十二、FACL 文件访问权限控制

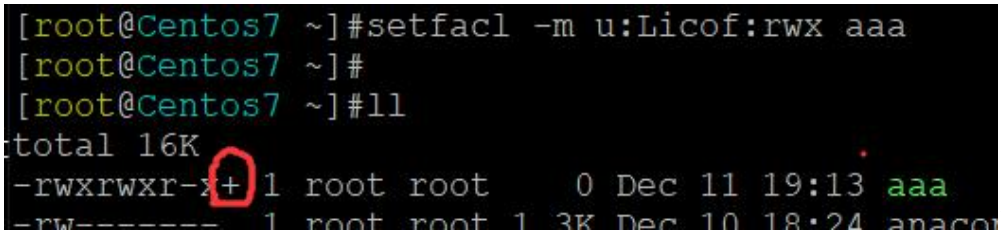
文件访问控制列表是比较灵活的权限分配方法，传统的基本 user:group:other 三类用户的权限控制不够灵活，所以在 Linux2.6 及以后的内核版本中，配合 ext2，ext3，ext4，xfs 等文件系统就能使用 FACL

FACL 可以为任意用户/组 分配 rwx 权限

#setfacl -m u:用户名:rwx 文件名 //允许指定用户对指定文件有 rwx 的权限



配置前



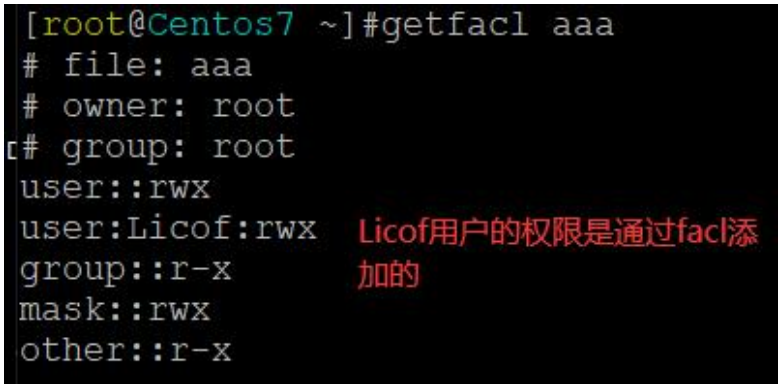
配置后

*配置前后有没有发现不同之处？

配置之前表示文件权限的 9 个字母后是一个点，配置 FACL 后变成一个加号+了

所以用 ls -l 查看文件属性时，如果发现权限后有加号+的，就说明该文件是配置了 FACL 的

#getfacl 文件名 //查看该文件上应用的 FACL



#setfacl -m g:组名:r 文件名 //给指定的组添加访问该文件的 r 读权限

#setfacl -m -R g:组名:r 目录名 //给指定的组添加访问该目录的 r 读权限

//-R 表示递归，即该目录下的所有文件都添加

#setfacl -x u:用户名 文件名 //删除指定用户对该文件的所有 facl

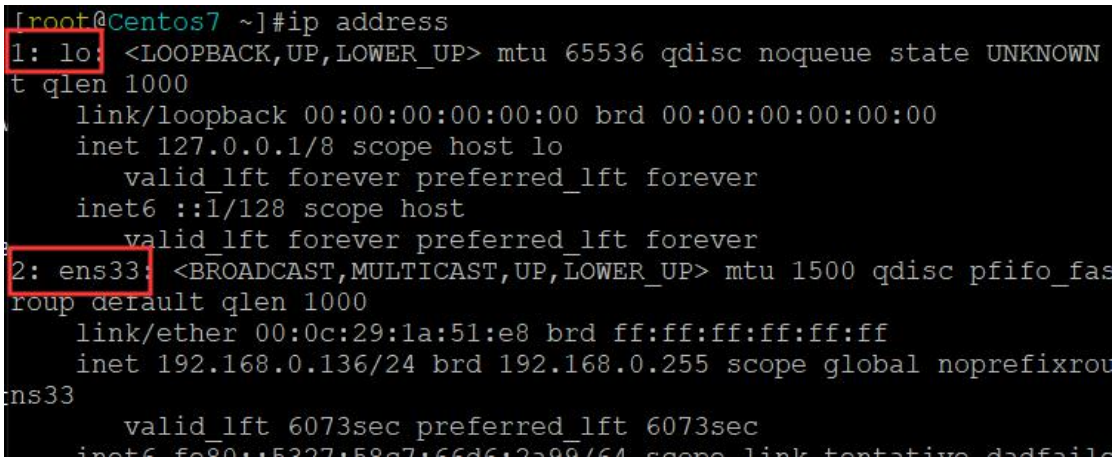
#setfacl -b 文件名 //删除该文件上的所有 facl

*使用 mv, cp -p 命令移动复制文件时，将保持文件的 FACL 设置

*在 Centos7 中，若目录上已配置 facl，在目录里创建新文件时，新文件不会继承目录的 facl

十三、网络配置

#ip address //先查看有几张网卡，以及网卡的 IP



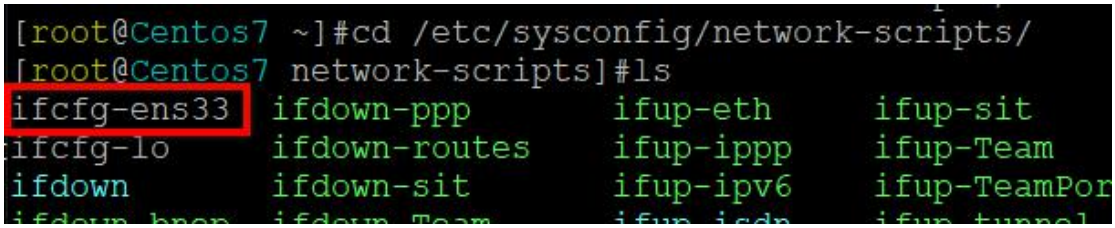
*如上图，每个网卡名前都有一个数字编号，从 1 开始。本系统中有 2 个网卡（lo 和 ens33）

lo 表示本地环回网卡，不用管它

ens33 表示一张真实的网卡

网卡名后面跟着一串相关的信息，state 表示网卡的状态，UP 表示启用，DOWN 表示未启用，inet 表示 IPv4 地址，inetv6 表示 ipv6 地址。

Centos7 的网卡配置文件在/etc/sysconfig/network-scripts/目录下，文件名为 ifcfg-网卡名，比如 ens33 的网卡对应的配置文件为/etc/sysconfig/network-scripts/ifcfg-ens33



要想对网卡进行配置，只需编辑其对应的配置文件即可

#cat /etc/sysconfig/network-scripts/ifcfg-网卡名 //查看网卡的配置文件

[root@Centos7 network-scripts]#cat ifcfg-ens33

TYPE=Ethernet //网卡类型为 Ethernet

PROXY_METHOD=none

BROWSER_ONLY=no

BOOTPROTO=dhcp //启动模式为 dhcp

DEFROUTE=yes

IPV4_FAILURE_FATAL=no

IPV6INIT=yes

IPV6_AUTOCONF=yes

IPV6_DEFROUTE=yes

IPV6_FAILURE_FATAL=no

IPV6_ADDR_GEN_MODE=stable-privacy

NAME=ens33 //网卡名称

UUID=ef235852-c62c-4431-8424-b4f36f315247

DEVICE=ens33　　//对应的网卡设备
ONBOOT=no　　//是否随开机启用，no 表示不随开机启用

当网络中有可用的 dhcp 服务器时，可以将 BOOTPROTO 设为 dhcp
也可以设置使用手工指定的 IP：BOOTPROTO=static
*关于网络的操作比较复杂，特别是在虚拟机里安装的系统时，所以先建议大家去学习一下基本的网络知识，虚拟机软件的使用，以及 VMware Workstation 的三种网络模式，请先学完了网络基础知识再继续以下的学习！！！

网卡的命名

CentOS7 默认使用一致的网络设备名（不再用传统的 eth0，eth1 来命名）
①设备名最前面 2 个字母表示网络类型
- **en**　为以太网设备
- **wl**　为无线局域网设备
- **ww**　为无线广域网设备

②随后的第 3 个字母用于区分不同的硬件类型
- **o**　表示主板板载设备（Onboard device）
- **s**　表示热拔插设备（hot-plag Slot）
- **p**　表示 PCI 总线或 USB 接口上的设备（Pci device）

③最后的一串数字为编号
例：

eno16777736	表示板载的以太网设备，索引编号为 16777736
enp0s8	表示 PCI 接口的以太网设备，PCI 总线地址为 0，插槽编号为 8
ens33	表示热拔插插槽上的以太网设备，插槽编号为 33
wlp12s0	表示 PCI 接口无线以太网设备，PCI 总线地址为 12，插槽编号为 0

手工配置网卡参数

#vi /etc/sysconfig/network-scripts/ifcfg-ens33 //配置 ens33 网卡

其他原有的配置先不变，修改或添加以下配置：

BOOTPROTO=static //使用静态地址

IPADDR0=192.168.0.136 //配置 IP 地址，ipaddr 后面接个数字 0

GATEWAY0=192.168.0.1 //配置网关，gateway 后面接个数字 0

PREFIX0=24 //配置子网掩码的位数,后面也是 0

DNS1=192.168.0.1 //配置 dns，只能从 1 开始编号

NM_CONTROLLED=yes //受 NetworkManager 服务管理

ONBOOT=yes //随开机启用

保存，重启系统

#ip address //查看 IP，命令后的 address 参数可以省写，只写前面几个字母，
//确保参数没有二义性即可

```
[root@Centos7 ~]#ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKN
t 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 pfifo
roup default qlen 1000
    link/ether 00:0c:29:1a:51:e8 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.136/24 brd 192.168.0.255 scope global noprefi
ns33
```

#ip route //查看默认网关

```
[root@Centos7 ~]#ip route
default via 192.168.0.1 dev ens33 proto
192.168.0.0/24 dev ens33 proto kernel sc
```

#ping 8.8.8.8 //使用 ping 工具，默认是常 ping，需要按下 Ctrl+C 停止

#ping -c 6 -s 300 -W 500 -I ens33 目标 IP //带参数的 ping

-c 表示 ping 的次数，-s 表示 ping 包的大小（字节），-W 表示超时（毫秒）

-I 表示出接口（相当于指定源 IP 为出接口 IP）

```
[root@Centos7 ~]#ping -c 3 -s 300 -W 500 -I ens33 8.8.8.8
PING 8.8.8.8 (8.8.8.8) from 192.168.0.136 ens33: 300(328) bytes of d
76 bytes from 8.8.8.8: icmp_seq=1 ttl=53 (truncated)
76 bytes from 8.8.8.8: icmp_seq=2 ttl=53 (truncated)
76 bytes from 8.8.8.8: icmp_seq=3 ttl=53 (truncated)

--- 8.8.8.8 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 20.303/20.638/21.105/0.378 ms
```

网络命令操作

Linux 系统以前常用的网络工具有 net-tools，这个软件包里有 ifconfig，route，arp，netstat 等命令。但自 2001 年起，Linux 社区已经对其停止维护。同时，一些 Linux 发行版比如 Arch Linux 和 CentOS/RHEL 7 则已经完全抛弃了 net-tools，只支持 iproute2。

十四、iproute2 的命令

#ip link show //显示出所有可用的网络接口列表（包括未激活的）

```
[root@Cof-Lee ~]# ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group d
ault qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFA
T group default qlen 1000
    link/ether 00:0c:29:d9:78:f1 brd ff:ff:ff:ff:ff:ff
3: ens37: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFA
T group default qlen 1000
```

#ip link set down *网卡名* //停用某个网络接口

```
[root@Cof-Lee ~]# ip link set down ens37
[root@Cof-Lee ~]# ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEF
ault qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
T group default qlen 1000
    link/ether 00:0c:29:d9:78:f1 brd ff:ff:ff:ff:ff:ff
3: ens37: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo_fast state DOWN mode DE
fault qlen 1000
```

#ip link set up *网卡名* //激活某个网络接口

```
[root@Cof-Lee ~]# ip link set up ens37
[root@Cof-Lee ~]# ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT
ault qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mo
T group default qlen 1000
    link/ether 00:0c:29:d9:78:f1 brd ff:ff:ff:ff:ff:ff
3: ens37: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mo
T group default qlen 1000
```

#ip addr 或 **ip addr show** //查看所有网络接口的 IP 地址

#ip addr add *IP 地址/子网掩码位数* **dev** *网卡名* //给指定网卡添加 IP 地址

#ip addr show dev *网卡名* //查看指定网卡 IP 地址

```
[root@Cof-Lee ~]# ip addr add 192.168.33.1/24 dev ens37
[root@Cof-Lee ~]# ip addr show dev ens37
3: ens37: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
lt qlen 1000
    link/ether 00:0c:29:d9:78:fb brd ff:ff:ff:ff:ff:ff
    inet 192.168.33.1/24 scope global ens37
       valid_lft forever preferred_lft forever
[root@Cof-Lee ~]#
```

#**ip addr del** ***IP 地址/子网掩码位数*** **dev** ***网卡名*** //移除指定网卡的指定 IP 地址

```
[root@Cof-Lee ~]# ip addr del 192.168.33.1/24 dev ens37
[root@Cof-Lee ~]# ip addr show dev ens37
3: ens37: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
lt qlen 1000
    link/ether 00:0c:29:d9:78:fb brd ff:ff:ff:ff:ff:ff
[root@Cof-Lee ~]#
```

//使用 iproute2 可以给同一个网卡配多条 IP 地址（每个 IP 都是可用的）

```
[root@Cof-Lee ~]# ip addr add 192.168.33.1/24 dev ens37
[root@Cof-Lee ~]# ip addr add 192.168.33.2/24 dev ens37
[root@Cof-Lee ~]# ip addr add 192.168.33.3/24 dev ens37
[root@Cof-Lee ~]# ip addr show dev ens37
3: ens37: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_f
lt qlen 1000
    link/ether 00:0c:29:d9:78:fb brd ff:ff:ff:ff:ff:ff
    inet 192.168.33.1/24 scope global ens37
       valid_lft forever preferred_lft forever
    inet 192.168.33.2/24 scope global secondary ens37
       valid_lft forever preferred_lft forever
    inet 192.168.33.3/24 scope global secondary ens37
       valid_lft forever preferred_lft forever
```

#**ip link set dev** ***网卡名*** **address** ***MAC 地址*** //修改接口的 MAC 地址（要先停用此接口）

```
[root@Cof-Lee ~]# ip link set down ens37
[root@Cof-Lee ~]# ip link set dev ens37 address 00:00:12:34:56:78
[root@Cof-Lee ~]# ip addr show dev ens37
3: ens37: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo_fast state DOWN
00
    link/ether 00:00:12:34:56:78 brd ff:ff:ff:ff:ff:ff
    inet 192.168.33.1/24 scope global ens37
       valid_lft forever preferred_lft forever
    inet 192.168.34.1/24 scope global ens37
       valid_lft forever preferred_lft forever
```

#**ip route** 或 **ip route show** //查看路由表

```
[root@Cof-Lee ~]# ip route show
default via 192.168.1.1 dev ens33 proto dhcp metric 100
192.168.1.0/24 dev ens33 proto kernel scope link src 192.168.1.12 metric 100
192.168.33.0/24 dev ens37 proto kernel scope link src 192.168.33.1
192.168.34.0/24 dev ens37 proto kernel scope link src 192.168.34.1
```

#**ip route add default via** ***默认网关 IP*** **dev** ***出接口*** //配置默认路由

```
[root@Cof-Lee ~]# ip route
default via 192.168.1.1 dev ens33 proto dhcp metric 100
192.168.1.0/24 dev ens33 proto kernel scope link src 192.168.1.12 metric 100
192.168.33.0/24 dev ens37 proto kernel scope link src 192.168.33.1
192.168.34.0/24 dev ens37 proto kernel scope link src 192.168.34.1
[root@Cof-Lee ~]# ip route add default via 192.168.33.254 dev ens37
[root@Cof-Lee ~]# ip route
default via 192.168.33.254 dev ens37
default via 192.168.1.1 dev ens33 proto dhcp metric 100
192.168.1.0/24 dev ens33 proto kernel scope link src 192.168.1.12 metric 100
192.168.33.0/24 dev ens37 proto kernel scope link src 192.168.33.1
192.168.34.0/24 dev ens37 proto kernel scope link src 192.168.34.1
```

#**ip route replace default via** ***默认网关 IP*** **dev** ***出接口*** //替代原默认路由

```
[root@Cof-Lee ~]# ip route
default via 192.168.33.254 dev ens37
114.114.0.0/16 via 192.168.1.1 dev ens33
192.168.1.0/24 dev ens33 proto kernel scope link src 192.168.1.12 metric 100
192.168.33.0/24 dev ens37 proto kernel scope link src 192.168.33.1
192.168.34.0/24 dev ens37 proto kernel scope link src 192.168.34.1
[root@Cof-Lee ~]# ip route replace default via 192.168.1.1 dev ens33
[root@Cof-Lee ~]# ip route
default via 192.168.1.1 dev ens33
114.114.0.0/16 via 192.168.1.1 dev ens33
192.168.1.0/24 dev ens33 proto kernel scope link src 192.168.1.12 metric 100
192.168.33.0/24 dev ens37 proto kernel scope link src 192.168.33.1
192.168.34.0/24 dev ens37 proto kernel scope link src 192.168.34.1
```

#ip route del default //删除默认路由

#ip route add *网段/子网掩码位数* **via** *下一跳 IP* **dev** *出接口* //添加一条静态路由

```
[root@Cof-Lee ~]# ip route add 114.114.0.0/16 via 192.168.1.1 dev ens33
[root@Cof-Lee ~]# ip route
114.114.0.0/16 via 192.168.1.1 dev ens33
192.168.1.0/24 dev ens33 proto kernel scope link src 192.168.1.12 metric 100
192.168.33.0/24 dev ens37 proto kernel scope link src 192.168.33.1
192.168.34.0/24 dev ens37 proto kernel scope link src 192.168.34.1
```

#ip route del *网段/子网掩码位数* **via** *下一跳 IP* **dev** *出接口* //删除一条静态路由

```
[root@Cof-Lee ~]# ip route
default via 192.168.1.1 dev ens33
114.114.0.0/16 via 192.168.1.1 dev ens33
192.168.1.0/24 dev ens33 proto kernel scope link src 192.168.1.12 metric 100
192.168.33.0/24 dev ens37 proto kernel scope link src 192.168.33.1
192.168.34.0/24 dev ens37 proto kernel scope link src 192.168.34.1
[root@Cof-Lee ~]# ip route del 192.168.34.0/24
[root@Cof-Lee ~]# ip route
default via 192.168.1.1 dev ens33
114.114.0.0/16 via 192.168.1.1 dev ens33
192.168.1.0/24 dev ens33 proto kernel scope link src 192.168.1.12 metric 100
192.168.33.0/24 dev ens37 proto kernel scope link src 192.168.33.1
[root@Cof-Lee ~]#
```

#ss //查看套接字统计信息

```
[root@Cof-Lee ~]# ss |more
Netid  State      Recv-Q Send-Q Local Address:Port                 Peer Address:Port
u_str  ESTAB      0      0       * 20551                 * 20550
u_str  ESTAB      0      0       * 20565                 * 20566
u_str  ESTAB      0      0       * 20521                 * 20520
tcp    ESTAB      0      52     192.168.1.12:ssh                 192.168.1.6:30597
```

#ip neigh //查看 arp 表

```
[root@Cof-Lee ~]# ip neigh
192.168.33.254 dev ens37 lladdr 00:50:56:c0:00:01 STALE
192.168.1.6 dev ens33 lladdr 78:0c:b8:6b:95:4f REACHABLE
192.168.1.1 dev ens33 lladdr 74:5a:aa:99:85:7b STALE
```

#ip neigh add *IP 地址* **lladdr** *MAC 地址* **dev** *网卡名* //添加一条静态 arp 项

```
[root@Cof-Lee ~]# ip neigh add 192.168.33.57 lladdr 00:00:23:33:53:a8 dev ens37
[root@Cof-Lee ~]# ip neigh
192.168.33.57 dev ens37 lladdr 00:00:23:33:53:a8 PERMANENT
192.168.33.254 dev ens37 lladdr 00:50:56:c0:00:01 STALE
192.168.1.6 dev ens33 lladdr 78:0c:b8:6b:95:4f REACHABLE
```

#**ip neigh del** ***IP 地址*** **dev** ***网卡名*** //删除一条静态 arp 项

```
[root@Cof-Lee ~]# ip neigh
192.168.33.57 dev ens37 lladdr 00:00:23:33:53:a8 PERMANENT
192.168.33.254 dev ens37 lladdr 00:50:56:c0:00:01 STALE
192.168.1.6 dev ens33 lladdr 78:0c:b8:6b:95:4f REACHABLE
192.168.1.1 dev ens33 lladdr 74:5a:aa:99:85:7b REACHABLE
[root@Cof-Lee ~]# ip neigh del 192.168.33.57 dev ens37
[root@Cof-Lee ~]# ip neigh
192.168.33.254 dev ens37 lladdr 00:50:56:c0:00:01 STALE
192.168.1.6 dev ens33 lladdr 78:0c:b8:6b:95:4f REACHABLE
192.168.1.1 dev ens33 lladdr 74:5a:aa:99:85:7b REACHABLE
```

#**ip maddr list dev** ***网卡名*** //查看接口上的多播地址

```
[root@Cof-Lee ~]# ip maddr list dev ens33
2:      ens33
        link  01:00:5e:00:00:01
        link  33:33:00:00:00:01
        link  33:33:ff:eb:e8:c5
        inet  224.0.0.1
        inet6 ff02::1:ffeb:e8c5
        inet6 ff02::1
        inet6 ff01::1
```

#**ip maddr add** ***MAC 地址*** **dev** ***网卡名*** //添加多播地址

```
[root@Cof-Lee ~]# ip maddr add 00:00:12:34:44 dev ens33
[root@Cof-Lee ~]# ip maddr list dev ens33
2:      ens33
        link  01:00:5e:00:00:01
        link  33:33:00:00:00:01
        link  33:33:ff:eb:e8:c5
        link  00:00:12:34:44:00 static
        inet  224.0.0.1
        inet6 ff02::1:ffeb:e8c5
        inet6 ff02::1
```

#**ip maddr del** ***MAC 地址*** **dev** ***网卡名*** //删除多播地址

```
[root@Cof-Lee ~]# ip maddr del 00:00:12:34:44 dev ens33
[root@Cof-Lee ~]# ip maddr list dev ens33
2:      ens33
        link  01:00:5e:00:00:01
        link  33:33:00:00:00:01
        link  33:33:ff:eb:e8:c5
        inet  224.0.0.1
        inet6 ff02::1:ffeb:e8c5
```

以上所有对网卡的 IP、MAC 地址操作都只是临时的，系统重启后，就不存在了。
永久保存网卡配置需修改网卡的对应的配置文件

安装 net-tools

在 Linux 系统里查看 TCP/IP 的相关信息的命令是 ifconfig，然而 CentOS 的最小化安装是没有安装的 ifconfig 网络工具的

```
[root@localhost ~]# ifconfig
-bash: ifconfig: command not found
[root@localhost ~]#
```

我们要装一下 ifconfig

使用 yum 安装（安装软件的操作可见后面的章节）

```
[root@localhost ~]# yum search ifconfig
Loaded plugins: fastestmirror
Determining fastest mirrors
 * base: mirrors.cn99.com
 * extras: mirrors.163.com
 * updates: mirrors.cn99.com
base                                                      | 3.6 kB  00:00:00
extras                                                    | 3.4 kB  00:00:00
updates                                                   | 3.4 kB  00:00:00
(1/4): extras/7/x86_64/primary_db                         | 186 kB  00:00:01
(2/4): base/7/x86_64/group_gz                             | 166 kB  00:00:01
(3/4): updates/7/x86_64/primary_db                        | 5.2 MB  00:00:04
(4/4): base/7/x86_64/primary_db                           | 5.9 MB  00:00:05
============================================ Matched: ifconfig ============================================
net-tools.x86_64 : Basic networking tools
[root@localhost ~]#
```

ifconfig 在 net-tools 这个软件包里

```
[root@localhost ~]# yum install net-tools.x86_64
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirrors.cn99.com
 * extras: mirrors.163.com
 * updates: mirrors.cn99.com
Resolving Dependencies
--> Running transaction check
---> Package net-tools.x86_64 0:2.0-0.22.20131004git.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================================================
 Package             Arch             Version                                 Repository       Size
================================================================================================================
Installing:
 net-tools           x86_64           2.0-0.22.20131004git.el7                base            305 k

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

Total download size: 305 k
Installed size: 917 k
Is this ok [y/d/N]:
```

```
Installed size: 917 k
Is this ok [y/d/N]: y
Downloading packages:
warning: /var/cache/yum/x86_64/7/base/packages/net-tools-2.0-0.22.20131004git.el7.x86_64.rpm: Head
 V3 RSA/SHA256 Signature, key ID f4a80eb5: NOKEY
Public key for net-tools-2.0-0.22.20131004git.el7.x86_64.rpm is not installed
net-tools-2.0-0.22.20131004git.el7.x86_64.rpm                    | 305 kB  00:00:00
Retrieving key from file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7
Importing GPG key 0xF4A80EB5:
 Userid     : "CentOS-7 Key (CentOS 7 Official Signing Key) <security@centos.org>"
 Fingerprint: 6341 ab27 53d7 8a78 a7c2 7bb1 24c6 a8a7 f4a8 0eb5
 Package    : centos-release-7-5.1804.el7.centos.x86_64 (@anaconda)
 From       : /etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7
Is this ok [y/N]: y
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : net-tools-2.0-0.22.20131004git.el7.x86_64                    1/
  Verifying  : net-tools-2.0-0.22.20131004git.el7.x86_64                    1/

Installed:
  net-tools.x86_64 0:2.0-0.22.20131004git.el7

Complete!
[root@localhost ~]# _
```

```
[root@localhost ~]# ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.11  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::5d3b:c6d:6002:99d8  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:6e:44:f9  txqueuelen 1000  (Ethernet)
        RX packets 9064  bytes 12901656 (12.3 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3001  bytes 202363 (197.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 64  bytes 5568 (5.4 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 64  bytes 5568 (5.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

安装完毕，OK 了。

十五、net-tools 命令

#ifconfig -a //显示出所有可用的网络接口列表（包括未激活的）

```
[root@Cof-Lee ~]# ifconfig -a
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.12  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::32bf:88c7:81eb:e8c5  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:d9:78:f1  txqueuelen 1000  (Ethernet)
        RX packets 10246  bytes 12956322 (12.3 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2029  bytes 201722 (196.9 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ens37: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::dc79:6487:476c:3ebd  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:d9:78:fb  txqueuelen 1000  (Ethernet)
```

#ifconfig *网卡名* down //停用某个接口

#ifconfig *网卡名* up //激活某个接口

```
[root@Cof-Lee ~]# ifconfig ens37 down
[root@Cof-Lee ~]# ifconfig ens37 up
```

#ifconfig *网卡名 IP 地址／子网掩码位数*　　//给指定网卡添加 IP 地址

#ifconfig *网卡名*　　//查看指定网卡的 IP 地址

```
[root@Cof-Lee ~]# ifconfig ens37 192.168.33.1/24
[root@Cof-Lee ~]# ifconfig ens37
ens37: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.33.1  netmask 255.255.255.0  broadcast 192.16
        ether 00:0c:29:d9:78:fb  txqueuelen 1000  (Ethernet)
        RX packets 10  bytes 910 (910.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 196  bytes 35352 (34.5 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

#ifconfig *网卡名 0*　　//删除指定网卡的 IP 地址

```
[root@Cof-Lee ~]# ifconfig ens37 0
[root@Cof-Lee ~]# ifconfig ens37
ens37: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        ether 00:0c:29:d9:78:fb  txqueuelen 1000  (Ethernet)
        RX packets 10  bytes 910 (910.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 196  bytes 35352 (34.5 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions
```

//ifconfig 不能直接给一个网卡配置多个 IP，可以通过配置子接口的方式给其添加多个 IP

#ifconfig 网卡名:0　IP 地址 0/子网掩码位数

#ifconfig 网卡名:1　IP 地址 1/子网掩码位数

#ifconfig 网卡名:2　IP 地址 2/子网掩码位数

```
[root@Cof-Lee ~]# ifconfig ens37:0 192.168.33.1/24
[root@Cof-Lee ~]# ifconfig ens37:1 192.168.33.2/24
[root@Cof-Lee ~]# ifconfig ens37:2 192.168.33.3/24
[root@Cof-Lee ~]# ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.12  netmask 255.255.255.0  broadcast 192.168.1
        inet6 fe80::32bf:88c7:81eb:e8c5  prefixlen 64  scopeid 0x20<l
        ether 00:0c:29:d9:78:f1  txqueuelen 1000  (Ethernet)
        RX packets 11521  bytes 13083232 (12.4 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2655  bytes 277260 (270.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ens37: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        ether 00:0c:29:d9:78:fb  txqueuelen 1000  (Ethernet)
        RX packets 10  bytes 910 (910.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 196  bytes 35352 (34.5 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ens37:0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.33.1  netmask 255.255.255.0  broadcast 192.168.3
        ether 00:0c:29:d9:78:fb  txqueuelen 1000  (Ethernet)

ens37:1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.33.2  netmask 255.255.255.0  broadcast 192.168.3
        ether 00:0c:29:d9:78:fb  txqueuelen 1000  (Ethernet)

ens37:2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.33.3  netmask 255.255.255.0  broadcast 192.168.3
```

#**ifconfig** *网卡名:子接口* **del** *对应的 IP 地址*　　//删除子接口的 IP

```
[root@Cof-Lee ~]# ifconfig ens37:1 del 192.168.33.2
[root@Cof-Lee ~]# ifconfig ens37:2 del 192.168.33.3
```

#**ifconfig** *网卡名* **hw ether** *MAC 地址*　　//修改接口的 MAC 地址，要先停用此接口

```
[root@Cof-Lee ~]# ifconfig ens37 down
[root@Cof-Lee ~]# ifconfig ens37 hw ether 00:00:12:33:45:df
[root@Cof-Lee ~]# ifconfig ens37
ens37: flags=4098<BROADCAST,MULTICAST>  mtu 1500
        inet 192.168.33.4  netmask 255.255.255.0  broadcast 192.1
        ether 00:00:12:33:45:df  txqueuelen 1000  (Ethernet)
        RX packets 38  bytes 3108 (3.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
```

#**route -n** 或 **netstat -rn**　　//查看路由表

```
[root@Cof-Lee ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.1.1     0.0.0.0         UG    100    0        0 ens33
192.168.1.0     0.0.0.0         255.255.255.0   U     100    0        0 ens33
```

```
[root@Cof-Lee ~]# netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         192.168.1.1     0.0.0.0         UG        0 0          0 ens33
192.168.1.0     0.0.0.0         255.255.255.0   U         0 0          0 ens33
```

#**route add default gw** *默认网关 IP 出接口*　　//配置默认路由

```
[root@Cof-Lee ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.1.1     0.0.0.0         UG    100    0        0 ens33
192.168.1.0     0.0.0.0         255.255.255.0   U     100    0        0 ens33
[root@Cof-Lee ~]# route add default gw 192.168.1.1 ens33
[root@Cof-Lee ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.1.1     0.0.0.0         UG    0      0        0 ens33
0.0.0.0         192.168.1.1     0.0.0.0         UG    100    0        0 ens33
192.168.1.0     0.0.0.0         255.255.255.0   U     100    0        0 ens33
```

#**route del default gw** *默认网关 IP 出接口*　　//删除默认路由

```
0.0.0.0         192.168.1.1     0.0.0.0         UG    0      0        0 ens33
0.0.0.0         192.168.1.1     0.0.0.0         UG    100    0        0 ens33
192.168.1.0     0.0.0.0         255.255.255.0   U     100    0        0 ens33
[root@Cof-Lee ~]# route del default gw 192.168.1.1 ens33
[root@Cof-Lee ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.1.1     0.0.0.0         UG    100    0        0 ens33
192.168.1.0     0.0.0.0         255.255.255.0   U     100    0        0 ens33
```

#**route add -net** ***目的网段/掩码位数*** **gw** ***下一跳 IP*** **dev** ***出接口*** //添加一条静态路由

```
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.1.1     0.0.0.0         UG    100    0        0 ens33
192.168.1.0     0.0.0.0         255.255.255.0   U     100    0        0 ens33
192.168.33.0    0.0.0.0         255.255.255.0   U     0      0        0 ens37
[root@Cof-Lee ~]# route add -net 172.16.0.0/16 gw 192.168.33.254 dev ens37
[root@Cof-Lee ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.1.1     0.0.0.0         UG    100    0        0 ens33
172.16.0.0      192.168.33.254  255.255.0.0     UG    0      0        0 ens37
192.168.1.0     0.0.0.0         255.255.255.0   U     100    0        0 ens33
192.168.33.0    0.0.0.0         255.255.255.0   U     0      0        0 ens37
```

#**route del -net** ***目的网段/子网掩码位数*** //删除一条静态路由

```
172.16.0.0      192.168.33.254  255.255.0.0     UG    0      0        0 ens37
192.168.1.0     0.0.0.0         255.255.255.0   U     100    0        0 ens33
192.168.33.0    0.0.0.0         255.255.255.0   U     0      0        0 ens37
[root@Cof-Lee ~]# route del -net 172.16.0.0/16
[root@Cof-Lee ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.1.1     0.0.0.0         UG    100    0        0 ens33
192.168.1.0     0.0.0.0         255.255.255.0   U     100    0        0 ens33
192.168.33.0    0.0.0.0         255.255.255.0   U     0      0        0 ens37
```

#**netstat** 或 **netstat -l** //查看套接字情况

```
[root@Cof-Lee ~]# netstat -l
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:ssh             0.0.0.0:*               LISTEN
tcp        0      0 localhost:smtp          0.0.0.0:*               LISTEN
tcp6       0      0 [::]:ssh                [::]:*                  LISTEN
tcp6       0      0 localhost:smtp          [::]:*                  LISTEN
```

#**arp -an** 或 **arp -a** //查看 arp 表

```
[root@Cof-Lee ~]# arp -an
? (192.168.33.254) at 00:50:56:c0:00:01 [ether] on ens37
? (192.168.1.1) at 74:5a:aa:99:85:7b [ether] on ens33
? (192.168.1.6) at 78:0c:b8:6b:95:4f [ether] on ens33
[root@Cof-Lee ~]# arp -a
? (192.168.33.254) at 00:50:56:c0:00:01 [ether] on ens37
gateway (192.168.1.1) at 74:5a:aa:99:85:7b [ether] on ens33
remoteClient (192.168.1.6) at 78:0c:b8:6b:95:4f [ether] on ens33
[root@Cof-Lee ~]#
```

#**arp -s** ***IP 地址 MAC 地址*** //添加一条静态 arp 项

```
[root@Cof-Lee ~]# arp -s 192.168.33.25 00:00:34:33:a4:33
[root@Cof-Lee ~]# arp -a
? (192.168.33.25) at 00:00:34:33:a4:33 [ether] PERM on ens37
? (192.168.33.254) at 00:50:56:c0:00:01 [ether] on ens37
gateway (192.168.1.1) at 74:5a:aa:99:85:7b [ether] on ens33
```

#**arp -d** ***IP 地址*** //删除一条静态 arp 项

```
[root@Cof-Lee ~]# arp -d 192.168.33.25
```

#**netstat -g** //查看网络接口上的多播地址

```
[root@Cof-Lee ~]# netstat -g
IPv6/IPv4 Group Memberships
Interface       RefCnt Group
--------------- ------ ---------------------
lo              1      all-systems.mcast.net
ens33           1      all-systems.mcast.net
ens37           1      all-systems.mcast.net
lo              1      ff02::1
lo              1      ff01::1
ens33           1      ff02::1:ffeb:e8c5
ens33           1      ff02::1
ens33           1      ff01::1
```

#ipmaddr add ***MAC 地址*** **dev** ***网卡名*** //添加多播地址

```
[root@Cof-Lee ~]# ipmaddr add 00:00:23:55:f3:34 dev ens37
[root@Cof-Lee ~]# netstat -g
IPv6/IPv4 Group Memberships
Interface       RefCnt Group
--------------- ------ ---------------------
lo              1      all-systems.mcast.net
ens33           1      all-systems.mcast.net
ens37           1      all-systems.mcast.net
lo              1      ff02::1
```

上图中直接用 netstat -g 看不到添加的多播地址

#ipmaddr show dev ***网卡名*** //查看指定接口的多播地址

```
[root@Cof-Lee ~]# ipmaddr add 00:00:23:55:f3:34 dev ens37
[root@Cof-Lee ~]# ipmaddr show dev ens37
3:      ens37
        link  01:00:5e:00:00:01
        link  33:33:00:00:00:01
        link  00:00:23:55:f3:34 static
        inet  224.0.0.1
```

#ipmaddr del ***MAC 地址*** **dev** ***网卡名*** //删除指定的多播地址

```
        link  33:33:00:00:00:01
        link  00:00:23:55:f3:34 static
        inet  224.0.0.1
        inet6 ff02::1
        inet6 ff01::1
[root@Cof-Lee ~]# ipmaddr del 00:00:23:55:f3:34 dev ens37
[root@Cof-Lee ~]# ipmaddr show dev ens37
3:      ens37
        link  01:00:5e:00:00:01
        link  33:33:00:00:00:01
        inet  224.0.0.1
```

以上所有对网卡的 IP、MAC 地址操作都只是临时的，系统重启后，就不存在了。
永久保存网卡配置需修改网卡的对应的配置文件。
网卡的配置文件在**/etc/sysconfig/network-scripts/**目录下
网卡配置文件的命名规则是 **ifcfg-网卡名** （比如 ens37 网卡的配置文件就是 ifcfg-ens37）
我们在新安装系统后会自动生成一个网卡的配置文件（可修改）如果是新插上的网卡，它的配置文件是要我们手动创建的。

```
[root@Cof-Lee ~]# cd /etc/sysconfig/network-scripts/
[root@Cof-Lee network-scripts]# ll
total 228
-rw-r--r--. 1 root root   310 Aug 24 08:58 ifcfg-ens33
-rw-r--r--. 1 root root   254 Jan  2  2018 ifcfg-lo
lrwxrwxrwx. 1 root root    24 Aug 24 08:54 ifdown -> ../../..
-rwxr-xr-x. 1 root root   654 Jan  2  2018 ifdown-bnep
```

```
[root@Cof-Lee network-scripts]# touch ifcfg-ens37
[root@Cof-Lee network-scripts]# ll
total 228
-rw-r--r--. 1 root root   310 Aug 24 08:58 ifcfg-ens33
-rw-r--r--. 1 root root     0 Aug 31 10:20 ifcfg-ens37
-rw-r--r--. 1 root root   254 Jan  2  2018 ifcfg-lo
```

先看看系统自动生成的 ens33 配置文件是怎么写的（等号后的值可以没有引号" "）

```
[root@Cof-Lee network-scripts]# cat ifcfg-ens33
TYPE="Ethernet"  以太网卡
PROXY_METHOD="none"
BROWSER_ONLY="no"
BOOTPROTO="dhcp"  表示是通过dhcp自动获取IP的
DEFROUTE="yes"
IPV4_FAILURE_FATAL="no"
IPV6INIT="yes"
IPV6_AUTOCONF="yes"
IPV6_DEFROUTE="yes"
IPV6_FAILURE_FATAL="no"
IPV6_ADDR_GEN_MODE="stable-privacy"
NAME="ens33"
UUID="c490f7a3-208c-4a4a-a9f6-51c8e250ca87"
DEVICE="ens33"  对应的设备是ens33这块网卡
ONBOOT="yes"  表示开机自启
```

编辑 ifcfg-ens37 文件

```
[root@Cof-Lee network-scripts]# vi ifcfg-ens37
TYPE=Ethernet
BOOTPROTO=static 静态指定IP地址
IPADDR0=192.168.33.1  IPv4地址
PREFIX0=24  掩码为24位
GATEWAY0=192.168.33.254  默认网关
DNS0=114.114.114.114
NAME=ens37
DEVICE=ens37  对应的设备为ens37这块网卡
ONBOOT=yes  开机自启
~
~
```

重启后，验证一下该配置是否生效

```
Last login: Fri Aug 31 09:01:57 2018 from remoteclient
**********this is motd,Welcome to CofLee.**********
[root@Cof-Lee ~]# ifconfig ens37
ens37: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.33.1  netmask 255.255.255.0  broadcast 192.168.33.255
        inet6 fe80::20c:29ff:fed9:78fb  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:d9:78:fb  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 10  bytes 768 (768.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@Cof-Lee ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.1.1     0.0.0.0         UG    100    0        0 ens33
0.0.0.0         192.168.33.254  0.0.0.0         UG    101    0        0 ens37
```

上图显示配置生效了。

十六、NetworkManager

#nmcli device status //查看网络接口的状态

```
[root@Centos7 ~]# nmcli device status
DEVICE  TYPE      STATE      CONNECTION
ens33   ethernet  connected  ens33
lo      loopback  unmanaged  --
```

#nmcli device show ens33 //查看指定网络接口的信息

```
[root@Centos7 ~]# nmcli devi show ens33
GENERAL.DEVICE:                         ens33
GENERAL.TYPE:                           ethernet
GENERAL.HWADDR:                         00:0C:29:6E:44:F9
GENERAL.MTU:                            1500
GENERAL.STATE:                          100 (connected)
GENERAL.CONNECTION:                     ens33
GENERAL.CON-PATH:                       /org/freedesktop/NetworkManager/ActiveConnection/1
WIRED-PROPERTIES.CARRIER:               on
IP4.ADDRESS[1]:                         192.168.33.1/24
IP4.GATEWAY:                            192.168.33.254
IP4.ROUTE[1]:                           dst = 192.168.33.0/24, nh = 0.0.0.0, mt = 100
IP4.ROUTE[2]:                           dst = 0.0.0.0/0, nh = 192.168.33.254, mt = 100
IP4.DNS[1]:                             114.114.114.114
```

#nmcli connection show //显示所有网卡的连接

```
[root@Centos7 ~]# nmcli conn show
NAME    UUID                                  TYPE      DEVICE
ens33   12ae6313-13f7-4fe8-9885-2a5072688b7f  ethernet  ens33
```

#nmcli device disconnect ens33 //断开指定设备的连接，使其 down 掉

```
[root@Centos7 ~]# nmcli device disconnect ens33
Device 'ens33' successfully disconnected.
```

#nmcli connection up ifname ens33 //激活指定网卡的连接，使其 up

```
[root@Centos7 ~]# nmcli connection up ifname ens33
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkMa
ection/2)
```

#nmcli connection modify ens33 ipv4.method manual //修改 ip 地址获得方式为手工指定

#nmcli connection modify ens33 ipv4.method auto //修改 ip 地址获得方式为自动获取（DHCP）

★以下设置 ip 地址，网关、DNS 服务器的操作，修改完后要断开网卡的连接，再重新激活连接，才能使设置生效。

#nmcli connection modify ens33 ipv4.address 10.1.1.1/24 //设置网卡地址

#nmcli connection modify ens33 ipv4.gateway 10.1.1.254 //设置网关

#nmcli connection modify ens33 ipv4.dns "8.8.8.8 114.114.114.114" //设置 dns

#nmcli connection modify ens33 +ipv4.address 10.1.1.2/24 //增加一个 ip 地址

十七、team 链路聚合

#nmcli conn add type team con-name t_Name1 ifname IF1 config
'{"runner":{"name":"loadbalance"}}' //创建聚合口,并配置为负载均衡模式
#nmcli conn add type team-slave con-name t_port1 ifname ens33 master IF1
#nmcli conn add type team-slave con-name t_port1 ifname ens33 master IF1
//添加 2 个成员端口

#nmcli conn modify t_Name1 ipv4.method mannual //手工配置 IP
#nmcli conn modify t_Name1 ipv4.address 10.1.1.252/24 //配置静态 IP
#nmcli conn modify t_Name1 team.config '{"runner":{"name":"loadbalance"}}'
//设置为负载均衡模式，前面第一条命令已经配置，这里可以不重复配置
聚合口的负载模式还可以配置为：

activebackup	热备份，同时只启用一个成员端口
lacp	802.3 链路聚合
roundrobin	捆绑模式
loadbalance	负载均衡，同时启用所有成员端口

#teamdctl IF1 state //查看 team 状态
#teamnl IF1 ports //查看 team 的成员端口
#nmcli conn down/up IF1 //启用或关闭聚合口

#ip addr 查看显示的网卡名为 ifname,
网卡配置文件名为 con-name
网卡配置文件里的 NAME=con-name
DEVICE=ifname

十八、SSH 远程登录服务

SSH 是 secure shell 的缩写，它是一个比较安全可靠的远程登录协议。CentOS 系统默认是开启 SSH 服务的，只要系统能访问网络，我们就能用 SSH 远程登录到系统。

在 Windows 系统上常用的 SSH 远程登录客户端工具有：

Putty

SecureCRT

SSHSecureShellClient

Xshell

具体的用法请参考其他文档。

#systemctl status sshd //查看 ssh 服务状态，Linux 服务名一般是以字母 d 结尾

```
[root@Centos7 ~]#systemctl status sshd
● sshd.service - OpenSSH server daemon
   Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabl
led)
   Active: active (running) since Tue 2019-12-17 09:04:49 CST;
     Docs: man:sshd(8)
           man:sshd_config(5)
 Main PID: 793 (sshd)
   CGroup: /system.slice/sshd.service
           └─793 /usr/sbin/sshd -D
```

SSH 服务的主配置文件为 /etc/ssh/sshd_config

①常用的设置如下：

#vi /etc/ssh/sshd_config

Port 2233 //设置监听的端口号，默认为 22

UseDNS=no //不使用 dns 解析客户端的 IP，默认开启，所以有时用 SSH 登录系统时要等十几秒才显示输入密码的界面

MaxAuthTries 3 //密码尝试次数，输错 3 次就断开此次连接

PermitEmptyPassword no //不允许使用空密码登录

PermitRootLogin yes //允许使用 root 帐号登录

保存，重启 sshd 服务

#systemctl restart sshd

②登录空闲超时自动退出设置

可以在/etc/bashrc 文件里设置

#vi /etc/bashrc //在文件末尾添加三行：

TMOUT=600 //表示 timeout 为 600 秒

readonly TMOUT

export TMOUT

保存，然后执行此脚本文件

#source /etc/bashrc

③登录安全设置

更改 ssh 服务的端口号和禁用 root 帐号登录，已经有一定的安全保障了，但还不够。要防止别人使用弱口令暴力攻击，实现的操作有：当某个帐号尝试密码达到一定的数量还未正确时，

可以先禁止该帐号登录一段时间。使用 PAM

例：当用户连续输错密码达 3 次时禁止 5 分钟（300 秒）内再次尝试密码

#vi /etc/pam.d/sshd //在该文件里添加 1 行（配置有点长，这里分 2 行写了 //配置的时候写成 1 行）

auth required pam_tally2.so deny=3 unlock_time=300 even_deny_root root_unlock_time=360

保存即可

//deny 的次数要小于或等于 sshd 配置文件里的 MaxAuthTries 次数

#pam_tally2 //查看用户输错密码达到 3 次及以上的记录

#pam_tally2 -u *cofile* //查看指定用户的密码违例记录

④删除某个连接会话

#who //查看当前登录的会话情况

```
[root@Centos7 ~]#who
root     tty1         2019-12-17 09:55
root     pts/0        2019-12-17 10:37 (192.168.0.11
[root@Centos7 ~]#who am i
```

#who am i //查看自己当前是使用的哪个会话

```
[root@Centos7 ~]#who am i
root     pts/0        2019-12-17 10:37 (192.168.0.1
```

#pkill -kill -t tty1 //删除指定的会话

⑤登录前导语

#vi /etc/login.banner //自己先随便创建一个文件，内容为登录前显示的信息

****Hello, xxx***

dfdfsfdsfsfasdfasf*

保存，

#vi /etc/ssh/sshd_config

Banner /etc/login.banner //使用 banner

保存，重启 sshd 服务

#systemctl restart sshd

⑥登录后导语

#vi /etc/motd //直接编辑/etc/motd 文件，保存即可

在 Linux 命令行下**使用 ssh 客户端**登录其他系统：

ssh user@x.x.x.x -p 2233 //用户名@主机名或 IP -p 端口号

十九、磁盘操作

Linux 下的磁盘并不是直接插上就能使用的，需要挂载到某个目录下（当然，如果还没有格式化的，要先格式化）磁盘能不能直接挂载到某个目录下呢？不能，要先分区，只有磁盘上的分区才能挂载到某个目录下。

①磁盘的操作

#lsblk　　//查看磁盘分区情况（目前只有一块磁盘，名为 sda）

```
[root@Centos7 ~]#lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda 磁盘  8:0    0   20G  0 disk
├─sda1 分区1 8:1    0  300M  0 part /boot
└─sda2 分区2 8:2    0   16G  0 part /
sr0 光盘  11:0    1  4.2G  0 rom
```

分区挂载到的目录

在安装操作系统时，我们创建了 2 个分区，一个挂载到/boot 目录下，另一个挂载到/根目录下。在虚拟机里做实验时，我们可以再加上一块虚拟磁盘。然后再查看时，发现多了一块磁盘 sdb。　**注意，以下所有操作只能在虚拟环境中进行实验，不可在真实服务器中操作！**

```
[root@Centos7 ~]#lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda      8:0    0   20G  0 disk
├─sda1   8:1    0  300M  0 part /boot
└─sda2   8:2    0   16G  0 part /
sdb      8:16   0   20G  0 disk
sr0     11:0    1  4.2G  0 rom
```

#fdisk　-l　　//列出磁盘及分区详细信息

```
[root@Centos7 ~]#fdisk -l

磁盘sda
Disk /dev/sda: 21.5 GB, 21474836480 bytes, 41943040 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x000a398c

磁盘sda下的2个分区
   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *        2048      616447      307200   83  Linux
/dev/sda2          616448    34170879    16777216   83  Linux

磁盘sdb，新添加的，还未分区
Disk /dev/sdb: 21.5 GB, 21474836480 bytes, 41943040 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

#fdisk　磁盘名　　// Fdisk 是 Linux 下的磁盘分区工具，
//命令 fdisk 后接磁盘名可以对目标磁盘进行分区的操作

输入 fdisk /dev/sdb 后进入的是一个交互界面，可以输入字符命令进行相应的操作，常用的命令有：

Command (m for help): m　　//查看帮助

Command action

d	delete a partition 删除一个分区
g	create a new empty GPT partition table 创建 gtp 分区表
l	list known partition types 列出分区类型 id
n	add a new partition 创建一个新的分区
o	create a new empty DOS partition table 创建 mbr 分区表
p	print the partition table 查看分区表
q	quit without saving changes 退出，不保存本次编辑
t	change a partition's system id 修改分区的类型 id
w	write table to disk and exit 保存本次编辑

对于新的磁盘，我们创建新的分区就行了

Command (m for help): n　　//输入 n，创建一个新的分区

Partition type:

p	primary (0 primary, 0 extended, 4 free)
e	extended

Select (default p): p　　//输入 p，创建的分区为主分区

Partition number (1-4, default 1): 1　　//分区号为 1

First sector (2048-41943039, default 2048):　　//这里不填，用默认的 2048 就行

Using default value 2048

Last sector, +sectors or +size{K,M,G} (2048-41943039, default 41943039): +2G　　// 表 示 分 区 大小为 2GB

Partition 1 of type Linux and of size 2 GiB is set

Command (m for help): p　　//查看一下分区情况

Disk /dev/sdb: 21.5 GB, 21474836480 bytes, 41943040 sectors

Units = sectors of 1 * 512 = 512 bytes

Sector size (logical/physical): 512 bytes / 512 bytes

I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk label type: dos

Disk identifier: 0x810614be

Device Boot	Start	End	Blocks	Id	System	
/dev/sdb1	2048	4196351	2097152	83	Linux	//一行为一个分区项

Command (m for help): w　　//输入 w，保存并退出
The partition table has been altered!
Calling ioctl() to re-read partition table.
Syncing disks.

#partprobe　/dev/sdb　　//刷新该磁盘的分区

```
[root@Centos7 ~]#lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda      8:0    0   20G  0 disk
├─sda1   8:1    0  300M  0 part /boot
└─sda2   8:2    0   16G  0 part /
sdb      8:16   0   20G  0 disk
└─sdb1   8:17   0    2G  0 part
sr0     11:0    1  4.2G  0 rom
```

退出 fdisk 编辑后，先刷新磁盘的分区，再用 lsblk 查看磁盘分区情况，可见 sdb 下多一个分区 sdb1，大小为 2GB。

*分区和文件系统是息息相关的，每一个分区都要一个文件系统，才能存储文件。文件系统就是在分区里的目录，这个目录记录着文件的大小和位置等信息。（Windows 系统下常用的文件系统有 FAT16, FAT32, NTFS, ExFAT 等。
Linux 系统里常用的文件系统有 Ext 系列的 Ext2, Ext3, Ext4 和 xfs，jfs 等。
CentOS7 默认用 xfs 文件系统，当然也可以用其他的，比如 Ext4

#mkfs　-t　文件系统类型　/dev/sdb1　　//把 sdb1 分区格式化为指定的文件系统

```
[root@Centos7 ~]#mkfs -t ext4 /dev/sdb1
mke2fs 1.42.9 (28-Dec-2013)
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done
```

也可使用其他的命令：
#mkfs.ext4　/dev/sdb1　　//格式化为 ext4
#mkfs.xfs　/dev/sdb1　　//格式化为 xfs
如果之前已经给分区格式化了，想换成其他文件系统时，可以加-f 参数强制格式化
#mkfs.xfs　-f　/dev/sdb1
格式化就是在分区上建立文件系统。建立文件系统后，该分区就可以挂载到某个目录下，然

后就可以正常使用了。

#mkdir /www //创建一个新的目录

#mount /dev/sdb1 /www //挂载 sdb1 分区到/www 目录下

#df -Th //查看磁盘分区及挂载情况

```
[root@Centos7 ~]#mount /dev/sdb1 /www
[root@Centos7 ~]#df -Th
Filesystem     Type      Size  Used Avail Use% Mounted on
/dev/sda2      xfs        16G 1004M   16G   7% /
devtmpfs       devtmpfs  477M     0  477M   0% /dev
tmpfs          tmpfs     488M     0  488M   0% /dev/shm
tmpfs          tmpfs     488M  7.6M  480M   2% /run
tmpfs          tmpfs     488M     0  488M   0% /sys/fs/cgroup
/dev/sda1      xfs       297M  107M  191M  36% /boot
tmpfs          tmpfs      98M     0   98M   0% /run/user/0
/dev/sdb1      xfs       2.0G   33M  2.0G   2% /www
```

使用命令挂载，只是临时的，重启后，就不在了。需要写进配置文件里，才能实现开机自动挂载。

#cat /etc/fstab //先查看自动挂载配置文件/etc/fstab

```
[root@Centos7 ~]#cat /etc/fstab

#
# /etc/fstab
# Created by anaconda on Tue Dec 10 05:20:27 2019
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
UUID=d2b1d839-8344-43cc-94c5-1bed38840c60 /                       xfs     defaults        0 0
UUID=ac85b360-f70d-467e-9bba-f7bf51620d74 /boot                   xfs     defaults        0 0
```

/etc/fstab 文件最下面那 2 行就是挂载配置，UUID 表示分区的唯一标识符，也可以直接写分区名称，然后是挂载目录（也叫挂载点），接下来是文件系统，defautls 表示默认的挂载方式，0 0 表示不自检。

照着这个我们也可以写一条，把刚刚创建的 sdb1 分区，开机时自动挂载到/www 目录下。

#vi /etc/fstab //编辑配置文件，在最后添加一行:

/dev/sdb1 /www xfs defaults 0 0

保存即可。

如果想写磁盘分区的 UUID 的话，也可以，

#blkid //查看磁盘分区的 UUID

```
[root@Centos7 ~]#blkid
/dev/sda1: UUID="ac85b360-f70d-467e-9bba-f7bf51620d74" TYPE="xfs"
/dev/sda2: UUID="d2b1d839-8344-43cc-94c5-1bed38840c60" TYPE="xfs"
/dev/sdb1: UUID="6d85b1f0-18e1-4533-bc9e-76a272bed7cf" TYPE="xfs"
/dev/sr0: UUID="2018-05-03-20-55-23-00" LABEL="CentOS 7 x86_64" TYPE
```

#mount -U "6d85b1f0-18e1-4533-bc9e-76a272bed7cf" /mnt //以 uuid 挂载

*假如/dev/sdb1 挂载在/mnt 目录下时，想取消挂载，可以用以下命令中的任一个

#umount /mnt //取消/mnt 目录下的磁盘挂载

#umount /dev/sdb1 //取消分区 sdb1 的挂载

②光盘的操作

#lsblk //查看磁盘及分区情况里可以看到光盘设备

```
[root@Centos7 ~]#lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda      8:0    0   20G  0 disk
├─sda1   8:1    0  300M  0 part /boot
└─sda2   8:2    0   16G  0 part /
sdb      8:16   0   20G  0 disk
└─sdb1   8:17   0    2G  0 part /www
sr0     11:0    1  4.2G  0 rom
```

sr0 就是光盘，它的路径为/dev/sr0，这个 sr0 只是光盘，代理光盘，并不是一个目录，不能直接 cd 切换进入查看光盘里的文件。

光盘和磁盘分区一样也是要先挂载到某个目录下才能进行文件的操作。

#mount /dev/sr0 /mnt //挂载光盘 sr0 到/mnt 目录下

```
[root@Centos7 ~]#mount /dev/sr0 /mnt
mount: /dev/sr0 is write-protected, mounting read-only
```

提示只能以只读的形式挂载，因为光盘一般是 ROM 只读的。

#ls /mnt //查看光盘挂载目录下的文件，这个目录下的文件就是光盘里的文件

```
[root@Centos7 ~]#ls /mnt
CentOS_BuildTag  EULA  images    LiveOS    repodata
EFI              GPL   isolinux  Packages  RPM-GPG-KEY-CentOS-7
```

要想开机自动挂载光盘，也可以在/etc/fstab 文件里写上挂载项

#vi /etc/fstab //在最后添加一行：

/dev/sr0 /mnt iso9660 defaults 0 0

保存

*有时，如果没有插上光盘，而在配置文件/etc/fstab 里写上了光盘的自动挂载项，那么开机会失败，因为找不到要挂载的光盘。这里可以进入救援模式，编辑/etc/fstab 文件，删除那行挂载项就行。

③挂载光盘镜像文件

有时候没有光盘，只有其镜像文件，也可以挂载

#mount -t iso9660 -o loop /root/Centos7.iso /mnt //把 iso 光盘镜像文件挂载到/mnt 目录下

④创建 swap 分区并挂载

swap 分区就是虚拟内存，以前内存不是很大时，可能不够用，所以用硬盘上的某个分区或文件来充当虚拟的内存，把内存中放不下的数据临时放到磁盘里。
所以以前的配置是把 swap 分区大小设为内存大小的 2 倍，现在大内存的计算机，没有必要设置成 2 倍。不一定非得是 2 倍，也可以没有 swap 分区。

```
#swapon                          //查看 swap 分区

#fdisk /dev/sdb                  //编辑/dev/sdb 磁盘的分区表
Welcome to fdisk (util-linux 2.23.2).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.
Command (m for help): n          //创建新的分区
Partition type:
   p   primary (1 primary, 0 extended, 3 free)
   e   extended
Select (default p): p            //主分区
Partition number (2-4, default 2): 2          //分区编号为 2
First sector (4196352-41943039, default 4196352):          //这里就用默认的起始扇区号
Using default value 4196352
Last sector, +sectors or +size{K,M,G} (4196352-41943039, default 41943039): +1G
                                                           //分区大小为 1GB
Partition 2 of type Linux and of size 1 GiB is set
Command (m for help): t                       //修改分区的类型 id
Partition number (1,2, default 2): 2          //对 2 号分区进行修改分区 id 的操作
Hex code (type L to list all codes): 82             //分区 ID 改为 82（表示 swap 分区）
Changed type of partition 'Linux' to 'Linux swap / Solaris'
Command (m for help): w          //保存，退出
The partition table has been altered!

#partprobe   /dev/sdb            //刷新磁盘的分区表
#mkswap   /dev/sda2              //把刚刚创建的分区 2,格式化为 swap 分区
#swapon   /dev/sda2              //启用指定的 swap 分区
#swapon   -a                     //启用所有的 swap 分区
```

```
[root@Centos7 ~]#swapon
NAME       TYPE        SIZE USED PRIO
/dev/sdb2 partition 1024M   0B   -1
```

```
[root@Centos7 ~]#free -m
              total        used        free
Mem:            974         124         701
Swap:          1023           0        1023
```

有时，磁盘可能已经没有剩余未分配的空间了，只能创建 swap 文件来充当 swap 分区。

#dd if=/dev/zero of=/swapfile bs=512 count=2048000 //创建 1 个 G 的文件

```
[root@Centos7 ~]#dd if=/dev/zero of=/swapfile bs=512 count=2048000
2048000+0 records in
2048000+0 records out
1048576000 bytes (1.0 GB) copied, 9.26063 s, 113 MB/s
```

```
[root@Centos7 ~]#ls /
bin   dev  home  lib64  mnt  proc  run   srv       sys  usr  www
boot  etc  lib   media  opt  root  sbin  swapfile  tmp  var
```

/swapfile 就是一个普通文件，可以对它进行创建 swap 分区的操作

```
[root@Centos7 ~]#mkswap /swapfile
Setting up swapspace version 1, size = 1023996 KiB
no label, UUID=7513ddcf-c510-4b75-a5cc-f44ea9c30923
[root@Centos7 ~]#swapon /swapfile
swapon: /swapfile: insecure permissions 0644, 0600 suggested.
[root@Centos7 ~]#swapon
NAME      TYPE       SIZE USED PRIO
/dev/sdb2 partition 1024M 264K   -1
/swapfile file      1000M   0B   -2
```

swapon 查看时，最后一列表示优先级，负数，负数绝对值越小的优先级越高。优先级越高就越优先被使用。

挂载 swap 分区也可以写进配置文件/etc/fstab 里

#vi /dev/fstab //在最后添加：

```
/dev/sdb2       swap    swap        defaults  0  0
/swapfile       swap    swap        defaults  0  0
```

保存

#swapoff /swapfile //停用指定的 swap 分区

#swapoff -a //停用所有的 swap 分区

二十、软件的安装与管理

①使用 YUM 工具安装

YUM(Yellow dog Updater Modified)是一个软件包管理器，基于 rpm 包管理，能够从指定的服务器自动下载 rpm 包并且安装，可以自动处理依赖性关系。使得用户更方便地添加、删除、更新 rpm 包。此方法要求连网

1.先检查是否有要安装的软件包

#yum search ***软件名***　　//查找软件包

```
[root@Cof-Lee ~]# yum search gcc
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirrors.aliyun.com
 * extras: mirrors.cqu.edu.cn
 * updates: mirrors.aliyun.com
================================================== N/S matched: gcc
gcc-c++.x86_64 : C++ support for GCC
gcc-gnat.x86_64 : Ada 95 support for GCC
gcc-objc.x86_64 : Objective-C support for GCC
gcc-objc++.x86_64 : Objective-C++ support for GCC
gcc-plugin-devel.x86_64 : Support for compiling GCC plugins
libgcc.i686 : GCC version 4.8 shared support library
libgcc.x86_64 : GCC version 4.8 shared support library
relaxngcc-javadoc.noarch : Javadoc for relaxngcc
compat-gcc-44.x86_64 : Compatibility GNU Compiler Collection
compat-gcc-44-c++.x86_64 : C++ support for compatibility compiler
compat-gcc-44-gfortran.x86_64 : Fortran support for compatibility c
gcc.x86_64 : Various compilers (C, C++, Objective-C, Java, ...)
gcc-gfortran.x86_64 : Fortran support
```

上图显示有与 gcc 相关的包，包名为上图圈红的部分，我们要装一个 Linux 系统下的 c/c++ 语言编译器，就用 gcc.x86_64 这个包

2.安装软件包

#yum install ***软件名***　　//安装软件包

例：yum install **gcc**

```
[root@Cof-Lee ~]# yum install gcc
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirrors.aliyun.com
 * extras: mirrors.cqu.edu.cn
 * updates: mirrors.aliyun.com
Resolving Dependencies
--> Running transaction check
---> Package gcc.x86_64 0:4.8.5-28.el7_5.1 will be installed
--> Processing Dependency: libgomp = 4.8.5-28.el7_5.1 for pac
7_5.1.x86_64
```

```
 libgcc                  x86_64        4.8.5-28.el7_5.1            update
 libgomp                 x86_64        4.8.5-28.el7_5.1            update

Transaction Summary
================================================================================
Install  1 Package  (+6 Dependent packages)
Upgrade             ( 2 Dependent packages)

Total download size: 31 M
Is this ok [y/d/N]: y
Downloading packages:
Delta RPMs disabled because /usr/bin/applydeltarpm not installed.
```

上图提示一共要下载 31M，输入 y 表示同意。

```
Retrieving key from file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7
Importing GPG key 0xF4A80EB5:
 Userid     : "CentOS-7 Key (CentOS 7 Official Signing Key) <secur
 Fingerprint: 6341 ab27 53d7 8a78 a7c2 7bb1 24c6 a8a7 f4a8 0eb5
 Package    : centos-release-7-5.1804.el7.centos.x86_64 (@anaconda
 From       : /etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7
Is this ok [y/N]: y
Running transaction check
```

```
Installed:
  gcc.x86_64 0:4.8.5-28.el7_5.1

Dependency Installed:
  cpp.x86_64 0:4.8.5-28.el7_5.1        glibc-devel.x86_64 0:
  glibc-headers.x86_64 0:2.17-222.el7  kernel-headers.x86_64
  libmpc.x86_64 0:1.0.1-3.el7          mpfr.x86_64 0:3.1.1-4

Dependency Updated:
  libgcc.x86_64 0: .8.5-28.el7_5.1        libgomp.x86_64 0:

Complete!
[root@Cof-Lee ~]#
```

安装完成。

3.yum 删除软件包

#yum remove ***软件名***　　//删除软件包

例：yum remove gcc

```
[root@Cof-Lee ~]# yum remove gcc
Loaded plugins: fastestmirror
Resolving Dependencies
--> Running transaction check
---> Package gcc.x86_64 0:4.8.5-28.el7_5.1 will be erase
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package         Arch                Version
================================================================================
Removing:
 gcc             x86_64              4.8.5-28.el7_5.1
```

```
 Package         Arch                    Version
================================================================
Removing:
 gcc             x86_64                  4.8.5-28.el7_5.1

Transaction Summary
================================================================
Remove  1 Package
```

```
Removed:
  gcc.x86_64 0:4.8 5-28.el7_5.1

Complete!
```

4.yum 更新软件包

#yum check-update //先检查可以更新的软件包

```
[root@Cof-Lee ~]# yum check-update
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirrors.aliyun.com
 * extras: mirrors.cqu.edu.cn
 * updates: mirrors.aliyun.com
  以下为可更新的包
NetworkManager.x86_64                   1:1.10.2-16.el7_5
NetworkManager-libnm.x86_64             1:1.10.2-16.el7_5
NetworkManager-team.x86_64              1:1.10.2-16.el7_5
NetworkManager-tui.x86_64               1:1.10.2-16.el7_5
NetworkManager-wifi.x86_64              1:1.10.2-16.el7_5
audit.x86_64                            2.8.1-3.el7_5.1
```

#yum update *软件名* //更新特定的软件包（可以是**软件名**，也可以是**完整的软件包名**）

```
[root@Cof-Lee ~]# yum update iptables.x86_64
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirrors.aliyun.com
 * extras: mirrors.cqu.edu.cn
```

```
Total download size: 432 k
Is this ok [y/d/N]: y
Downloading packages:
```

```
Updated:
  iptables.x86_64 0:1.4.21-24.1.el7_5

Complete!
```

#yum update //更新所有的软件包

②RPM 文件本地安装

需提前下载好 rpm 文件到本地目录下，也可以用镜像光盘里的 rpm 文件

#rpm -qa //查询系统已安装的所有软件包

```
[root@Cof-Lee ~]# rpm -qa
NetworkManager-tui-1.10.2-13.el7.x86_64
grub2-common-2.02-0.65.el7.centos.2.noarch
authconfig-6.2.8-30.el7.x86_64
ncurses-base-5.9-14.20130511.el7_4.noarch
postfix-2.10.1-6.el7.x86_64
bind-license-9.9.4-61.el7.noarch
microcode_ctl-2.1-29.el7.x86_64
tzdata-2018c-1.el7.noarch
```

一屏显示不下，可以用**|more** 分页

#rpm -q ***软件名*** //查看是否已安装有该软件（有的话会显示软件包全称）

```
[root@Cof-Lee ~]# rpm -q gcc
package gcc is not installed
[root@Cof-Lee ~]# rpm -q openssh
openssh-7.4p1-16.el7.x86_64
```

rpm 包本地安装（rpm 包文件需提前下载好，保存到某个目录下）

#rpm -ivh ***包名*** //软件包名要求是完整的名称

```
[root@Cof-Lee ~]# ll
total 264
-rw-------. 1 root root   1341 Aug 24 08:59 anaconda-ks.cfg
drwxr-xr-x. 2 root root      6 Aug 29 08:58 newdisk
-rw-r--r--. 1 root root 266160 Aug 30 08:00 zip-3.0-11.el7.x86_64.rpm
[root@Cof-Lee ~]# rpm -ivh zip-3.0-11.el7.x86_64.rpm
Preparing...                          ################################# [100%]
Updating / installing...
   1:zip-3.0-11.el7                   ################################# [100%]
```

#rpm -ql ***软件名*** //查看软件包安装到哪个目录下了

```
[root@Cof-Lee ~]# rpm -ql zip
/usr/bin/zip
/usr/bin/zipcloak
/usr/bin/zipnote
/usr/bin/zipsplit
/usr/share/doc/zip-3.0
/usr/share/doc/zip-3.0/CHANGES
/usr/share/doc/zip-3.0/LICENSE
/usr/share/doc/zip-3.0/README
/usr/share/doc/zip-3.0/README.CR
```

有时安装某个 rpm 包的时候会提示该包有依赖项，不能安装，我们可以忽略依赖关系强制安装：**rpm -ivh --nodeps softpackage.rpm** 或 **rpm -ivh --nodeps --force softpackage.rpm**

#rpm -e ***软件名*** //卸载该软件（这里不能填软件名的完整名称，卸载时是没有提示的）

```
[root@Cof-Lee ~]# rpm -e zip-3.0-11.el7.x86_64.rpm
error: package zip-3.0-11.el7.x86_64.rpm is not installed
[root@Cof-Lee ~]# rpm -e zip
```

③源代码包安装：

源代码包要先下载好，并保存到系统本地目录下。

基本步骤

(1)配置：解压目录下执行　　./configure

(2)编译：解压目录下执行　　make

(3)安装：解压目录下执行　　make install

#一般的软件的默认安装目录在/usr/local，可以在./configure 后加参数 **－prefix=/**要安装的目录。比如要安装 apache 服务器，源代码包为 **httpd-2.2.34.tar.gz**

首先使用 tar 命令解压，解压完成后会多出一个文件夹 httpd-2.2.34

然后就可以到 httpd-2.2.34 目录下去进行配置、编译和安装操作了。

```
[root@localhost mysoft]# tar -zxvf httpd-2.2.34.tar.gz
[root@localhost mysoft]# ll
total 136800
drwxr-xr-x. 11 1001 1001      4096 Jul  6  2017 httpd-2.2.34
-rw-r--r--.  1 root root   7684419 Jan 26 04:44 httpd-2.2.34.tar.gz
```

```
[root@localhost mysoft]# whereis httpd
httpd: /usr/local/apache2/bin/httpd
```

删除软件时只需在安装目录下执行反安装命令：**make uninstall**　或者直接删除安装目录。

源代码包安装比较复杂，初学者不建议使用此种方法安装软件。一般**推荐用 YUM 安装**。但是用 YUM 安装时是要连网的，如果系统没有网络，或者不能连上公网，那该怎么办呢？

方法是搭建本地 YUM 源

1.在其实 CentOS 的安装光盘就是一个 YUM 源，只要把该光盘挂载到某个目录下，就可以把那个目录当成本地 YUM 源了。（本例中把 CentOS7 的安装光盘挂载到/mnt/Centos7 目录下）

```
[root@Cof-Lee ~]# ll /mnt
total 0
drwxr-xr-x. 2 root root 6 Aug 29 08:26 Centos7
[root@Cof-Lee ~]# mount /dev/cdrom /mnt/Centos7
mount: /dev/sr0 is write-protected, mounting read-only
[root@Cof-Lee ~]#
```

2.所有的 yum 配置文件都在/etc/yum.repos.d/目录下，在该目录下再创建一个后缀为.repo 的文件，比如 localyum.repo

```
[root@Cof-Lee yum.repos.d]# ls
CentOS-Base.repo  CentOS-Debuginfo.repo  CentOS-Media.repo    Ce
CentOS-CR.repo    CentOS-fasttrack.repo  CentOS-Sources.repo
[root@Cof-Lee yum.repos.d]# vi localyum.repo
[localCentos7Yum] #yum源ID，必须唯一
name=localyum-server #描述，可以随便写
baseurl=file:///mnt/Centos7 #表示以光盘的挂载点为yum源
enable=1 #表示可以被使用
gpgcheck=0 #不验证
priority=1 #优先级设为1，默认为99，数值越小越优先
~
```

编辑然后保存就可以了。

要使 priority 参数有效，必须安装 yum-plugin-priorities 插件

#rpm -qa | grep yum-plugin　　//查看系统是否安装了优先级的插件

```
[root@Cof-Lee yum.repos.d]# rpm -qa |grep yum-plugin
yum-plugin-fastestmirror-1.1.31-45.el7.noarch
```

没有 yum-plugin-priorities 插件

#yum install yum-plugin-priorities　　//安装插件

安装后要检查是否启用该插件，查看/etc/yum/pluginconf.d/priorities.conf 配置文件

```
[root@Cof-Lee ~]# cat /etc/yum/pluginconf.d/priorities.conf
[main]
enabled = 1
[root@Cof-Lee ~]#
```

enable=1 表示启用，=0 表示未启用

验证一下：（先清空 yum 缓存：**yum clean all**）

```
[root@Cof-Lee ~]# yum clean all
Loaded plugins: fastestmirror, priorities
Cleaning repos: base extras localCentos7Yum u
Cleaning up everything
```

启用前：

```
[root@Cof-Lee ~]# yum install zip
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirrors.shu.edu.cn
 * extras: mirrors.shu.edu.cn
 * updates: mirror.bit.edu.cn
Resolving Dependencies
--> Running transaction check
---> Package zip.x86_64 0:3.0-11.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package            Arch                Version                  Repository
================================================================================
Installing:
 zip                x86_64              3.0-11.el7               base
```

启用后：

```
Loading mirror speeds from cached hostfile
 * base: mirrors.shu.edu.cn
 * extras: mirrors.shu.edu.cn
 * updates: mirror.bit.edu.cn
831 packages excluded due to repository priority protections
Resolving Dependencies
--> Running transaction check
---> Package zip.x86_64 0:3.0-11.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package         Arch               Version                 Repository
================================================================================
Installing:
 zip             x86_64             3.0-11.el7              localCentos7Yum
```

其实以上的方法还不是很好，没有必要指定优先级，可以直接手动指定需要使用的 yum 源

#yum --disablerepo=* --enablerepo=localCentos7Yum install 软件名
//表示先禁用所有的 repo，再指定一个可用的 repo，--enablerepo=后接的是 repo 的 ID

```
#yum --disablerepo=\* --enable=localCentos7Yum search  xxx
```

如果觉得该命令太长了，可以使用 alias 别名，
#alias yumlocal='yum --disablerepo=* --enablerepo=localCentos7Yum'

如果需要**启用 gpgcheck** 呢？配置如下：
[yumid]
name=xxx
baseurl=http://xxxxxx/xccc/xxccc/ //是一个目录
gpgcheck=1
gpgkey=http://xxxxxx/xccc/xxxx/RPM-GPG-KEY-CentOS-7 //是一个文本文件

二十一、LVM 逻辑卷管理

Logical Volume Manager 逻辑卷管理，是建立在硬盘或分区之上的一个逻辑层，把若干个硬盘或分区虚拟成一个大的硬盘（卷组），然后在卷组上再划分虚拟的分区（逻辑卷），就可以在逻辑卷上创建文件系统，最后挂载到某个目录下
通过 LVM 可以扩展文件系统跨越磁盘

LVM 基本术语

PV 物理卷（Physical Volume）
是真实的硬盘或硬盘上的分区，或者是 RAID 设备，PV 处于 LVM 系统中的最底层

VG 卷组（Volume Group）
由一个或多个物理卷组成，虚拟成一个大的硬盘
卷组创建后，可以动态地添加物理卷到卷组中
卷组名就是虚拟硬盘设备名

LV 逻辑卷（Logical Volume）
是建立在卷组上的虚拟分区，在逻辑卷之上可以建立文件系统
普通的硬盘分区一旦建立就不能更改大小，而 LV 逻辑卷创建后可以动态地调整大小

PE 和 LE
Physical Extent 和 Logical Extent 物理区域和逻辑区域的大小是相同的，且一一对应
每一个物理区域是一个基本单元，是 LVM 寻址的最小存储单元，默认为 4MB
（这个 PE、LE 单元就相当于文件系统里的簇）

Centos 从版本 4 开始使用 LVM2

```
#yum install lvm2                    //安装 LVM2
```

创建卷命令：

```
#pvcreate   /dev/sdb   /dev/sdc        //创建物理卷，把 sdb 和 sdc 创建成物理卷
#vgcreate   data   /dev/sdb            //创建卷组，先把 sdb 物理卷加入
#lvcreate   -L   2G   -n   www   data    //在 data 卷组中创建名为 www 的逻辑卷，大小为 2GB
```

查看卷：

```
#pvdisplay   或 pvs
#vgdisplay   或 vgs
#lvdisplay   或 lvs
```

调整卷：

#vgextend data /dev/sdc //将指定的物理卷添加到 data 卷组中
#vgreduce data /dev/sdc //将指定的物理卷从 data 卷组中移除
#lvextend -L +3G /dev/data/www //扩展逻辑卷大小（增加 3GB）
#lvextend -L 3G /dev/data/www //扩展逻辑卷大小（扩展至 3GB）
#lvreduce -L -2G /dev/data/www //缩减逻辑卷，减 2GB

lvextend、lvreduce 命令支持 **-r |--resizefs** 参数用于调整逻辑卷大小的同时调整文件系统大小
创建了逻辑卷后，在逻辑卷上**建立文件系统**
#mikfs.ext4 /dev/data/www
再挂载到目录/srv/www 下
#mount /dev/data/www /srv/www

有时候挂载点正在被进程使用时，是不能卸载的，可以查看是哪个进程使用它
#fuser -cu /挂载点 //查看该挂载点有哪些进程在使用
#fuser -ck /挂载点 //杀死正在使用该挂载点的进程

启动时自动挂载文件系统（在/etc/fstab 里添加）
/dev/mapper/data-www /srv/www ext4 dafaults 0 0

*逻辑卷/dev/data/www 在文件系统里变成了/dev/mapper/卷组名-逻辑卷名

如果逻辑卷已经创建了文件系统，再调整大小，需要执行以下命令：
#resize2fs /dev/data/www //ext4 文件系统下使用该命令
#xfs_groupfs /dev/data/www //xfs 文件系统下使用该命令

在安装 CentOS7 系统时，默认是使用 LVM 的，在物理磁盘上创建 2 个分区，一个挂载到/boot 目录下，另一个做成了 VG 卷组 centos 的物理卷成员，然后在卷组 centos 上创建 2 个 LV 逻辑卷（root 和 home），分别挂载在/和/home 目录下。

```
[root@c      ~]# lsblk
NAME            MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda               8:0    0   446G  0 disk
├─sda1            8:1    0   500M  0 part /boot
└─sda2            8:2    0 445.5G  0 part
  ├─centos-root 253:0    0 435.5G  0 lvm  /
  └─centos-home 253:1    0    10G  0 lvm  /home
sr0              11:0    1  1024M  0 rom
```

因为/boot 里面存放的是系统启动文件，系统未完全启动时，LVM 也还未启动，所以/boot 不能是虚拟的，只能是位于真实的物理分区上，系统才能找得到该路径。

二十二、进程操作及作业控制

①ps 命令查看进程

#ps　　//查看当前用户的进程，仅显示有控制终端的进程

```
[cof@Cof-Lee ~]$ ps
   PID TTY          TIME CMD
  1652 tty1     00:00:00 bash
  1698 tty1     00:00:00 ps
```

#ps　-x　　//查看当前用户的进程，包括没有控制终端的进程

```
[root@Cof-Lee ~]# ps -x
   PID TTY      STAT   TIME COMMAND
     1 ?        Ss     0:02 /usr/lib/systemd/systemd --s
     2 ?        S      0:00 [kthreadd]
     3 ?        S      0:00 [ksoftirqd/0]
     5 ?        S<     0:00 [kworker/0:0H]
     6 ?        S      0:00 [kworker/u256:0]
```

#ps　-au　　//查看所有用户的进程，仅显示有控制终端的进程，可与 -x 选项组合

```
[root@Cof-Lee ~]# ps -au
USER        PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
cof        1652  0.0  0.4 115568  2280 tty1     Ss+  12:01   0:00 -bash
root       1681  0.0  0.4 115432  2024 pts/0    Ss   12:03   0:00 -bash
root       1708  0.0  0.3 155324  1852 pts/0    R+   12:08   0:00 ps -au
```

#ps　-u　cof　　//查看指定用户 cof 的进程，仅显示有控制终端的进程，可与 -x 选项组合

```
[root@Cof-Lee ~]# ps -u cof
   PID TTY          TIME CMD
  1652 tty1     00:00:00 bash
```

#ps　-aux | grep　ssh　　//查找系统中运行的 command 中带有"ssh"串的所有进程

```
[root@Cof-Lee ~]# ps -aux |grep ssh
root        872  0.0  0.8 112796  4296 ?        Ss   10:38   0:00 /usr/sbin/sshd -D
root       1677  0.0  1.1 158800  5528 ?        Ss   12:03   0:00 sshd: root@pts/0
root       1721  0.0  0.2 112704   968 pts/0    S+   12:11   0:00 grep --color=auto ssh
```

②杀死进程（结束进程）

#kill　1233　　//杀死 pid 为 1233 的进程

#kill　-9　1233　　//强行杀死 pid 为 1233 的进程

#pkill　进程名　　//杀死指定进程名的所有进程，所有同名进程都被杀死

#pkill　-u　cof　　//杀死用户 cof 的所有进程

#pkill　-9　-u　cof　　//强行杀死用户 cof 的所有进程

#pkill　-u　cof　进程名　　//杀死用户 cof 的指定进程

#pkill　-G　staff　　//杀死 staff 组成员运行的所有进程

③快捷键操作

Ctrl+D　　//正常结束前台正在运行的进程

Ctrl+C　　//强行结束前台正在运行的进程

Ctrl+Z　　//挂起一个正在前台运行的进程，暂停正在运行的进程且放到后台去

④后台进程操作（作业操作）

#jobs　　//查看在后台运行 或挂在后台的进程

```
[cof@Cof-Lee ~]$ jobs
[1]-  Stopped                 ping 192.168.33.10
[2]+  Stopped                 vi xx.txt
```

↑作业号 ↑状态　　↑进程名称

作业号后面的+加号表示默认作业号，—减号为第二默认作业号

#jobs　-l　　//加选项 -l 可以显示出作业对应的进程 pid

```
[cof@Cof-Lee ~]$ jobs -l
[1]-  1765 Stopped                 ping 192.168.33.10
[2]+  1766 Stopped (tty output)    vi xx.txt
```

#bg　%1　　//将作业号为 1 的进程在后台继续运行

#fg　%1　　//将作业号为 1 的进程调到前台来运行

// bg 和 fg 后若不加%参数，则认为是在操作默认作业号对应的进程

#kill　%2　　//杀死作业号为 2 的进程

#CMD &　　//表示在 cmd 命令行输入的命令后加上&符号，表示将该命令放到后台去运行

如：　#ping 192.168.33.1　&　　//表示在后台 ping192.168.33.1

```
jobs
[1]-  Stopped                 ping 192.168.33.10
[2]+  Stopped                 vi xx.txt
[3]   Running                 ping 192.168.33.10 &
```

二十三、归档压缩操作

归档就是指把文件打包在一起，打包成一个文件。有时候归档和压缩往往是连续进行的。（把若干个文件打包后再压缩）

①tar 命令：tar [-cxtzjvfpPN] +文件或目录

tar 命令参数：

-c　　建立一个归档文件的参数指令(create 的意思)

-x　　解开一个归档文件的参数指令

-t　　查看归档里面的文件!

*特别注意，在参数的下达中， -c-x-t 仅能存在一个，不能同时存在！因为不可能同时打包与解包。

-z　　同时具有 gzip 的属性，即需要用 gzip 压缩

-j　　同时具有 bzip2 的属性，即需要用 bzip2 压缩

-v　　压缩的过程中显示文件（这个常用，但不建议用在背景执行过程）

-f　　使用档名，请留意，在 f 之后要立即接归档名（打包后的文件名）不要再加参数

-p　　（小写的 p）使用原文件的原来属性（属性不会依据使用者而变）

-P　　（大写的 P）可以使用绝对路径来压缩

-N　　比后面接的日期(yyyy/mm/dd)还要新的才会被打包进新建的文件中

--exclude 文件名：在归档的过程中，不要将该文件打包

例一：将 /root/mydir1 目录下的文件全部打包成为 mytar.tar

#tar -cvf mytar.tar　/root/mydir1

//仅打包，不压缩 命令格式：tar +参数+目标文件名+源文件

```
[root@Cof-Lee ~]# tar -cvf mytar.tar /root/mydir1
tar: Removing leading `/' from member names
/root/mydir1/
/root/mydir1/myfile3
/root/mydir1/myfile4
[root@Cof-Lee ~]# ll
total 16
-rw-------. 1 root root  1341 Aug 24 08:59 anaconda-ks.cfg
drwxr-xr-x. 2 root root    36 Sep 15 02:02 mydir1
drwxr-xr-x. 2 root root     6 Sep 15 02:00 mydir2
-rw-r--r--. 1 root root     0 Sep 15 02:00 myfile1
-rw-r--r--. 1 root root     0 Sep 15 02:00 myfile2
-rw-r--r--. 1 root root 10240 Sep 15 02:03 mytar.tar
[root@Cof-Lee ~]#
```

#tar -zcvf mytar.tar.gz　/root/mydir1　//打包后，以 gzip 压缩（压缩后比不压缩的要小）

```
[root@Cof-Lee ~]# tar -zcvf mytar.tar.gz /root/mydir1
tar: Removing leading `/' from member names
/root/mydir1/
/root/mydir1/myfile3
/root/mydir1/myfile4
[root@Cof-Lee ~]# ll
total 20
-rw-------. 1 root root  1341 Aug 24 08:59 anaconda-ks.cfg
drwxr-xr-x. 2 root root    36 Sep 15 02:02 mydir1
drwxr-xr-x. 2 root root     6 Sep 15 02:00 mydir2
-rw-r--r--. 1 root root     0 Sep 15 02:00 myfile1
-rw-r--r--. 1 root root     0 Sep 15 02:00 myfile2
-rw-r--r--. 1 root root 10240 Sep 15 02:03 mytar.tar
-rw-r--r--. 1 root root   165 Sep 15 02:06 mytar.tar.gz
```

#tar -jcvf mytar.tar.bz2 /root/mydir1　　//打包后以 bzip2 压缩，如果没有安装 bz2 则会失败

```
[root@Cof-Lee ~]# tar -jcvf mytar.tar.bz2 /root/mydir1
tar: Removing leading `/' from member names
tar (child): bzip2: Cannot exec: No such file or directory
tar (child): Error is not recoverable: exiting now
/root/mydir1/
/root/mydir1/myfile3
/root/mydir1/myfile4
[root@Cof-Lee ~]# ll
total 20
-rw-------. 1 root root  1341 Aug 24 08:59 anaconda-ks.cfg
drwxr-xr-x. 2 root root    36 Sep 15 02:02 mydir1
drwxr-xr-x. 2 root root     6 Sep 15 02:00 mydir2
-rw-r--r--. 1 root root     0 Sep 15 02:00 myfile1
-rw-r--r--. 1 root root     0 Sep 15 02:00 myfile2
-rw-r--r--. 1 root root 10240 Sep 15 02:03 mytar.tar
-rw-r--r--. 1 root root     0 Sep 15 02:09 mytar.tar.bz2
-rw-r--r--. 1 root root   165 Sep 15 02:06 mytar.tar.gz
```

特别注意，在参数 f 之后的文件档名是自己取的，我们习惯上都用 .tar 来作为辨识。

如果加 z 参数，则以 .tar.gz 或 .tgz 来代表 gzip 压缩过的 tar 文件

如果加 j 参数，则以 .tar.bz2 来作为归档名

上述指令在执行的时候，会显示一个警告讯息：

『tar: Removing leading `/” from member names』那是关于绝对路径的特殊设定。

例二：查看归档文件内有哪些文件

#tar -tvf mytar.tar

```
[root@Cof-Lee ~]# tar -tvf mytar.tar
drwxr-xr-x root/root         0 2018-09-15 02:02 root/mydir1/
-rw-r--r-- root/root         0 2018-09-15 02:02 root/mydir1/myfile3
-rw-r--r-- root/root         0 2018-09-15 02:02 root/mydir1/myfile4
```

#tar -ztvf mytar.tar.gz

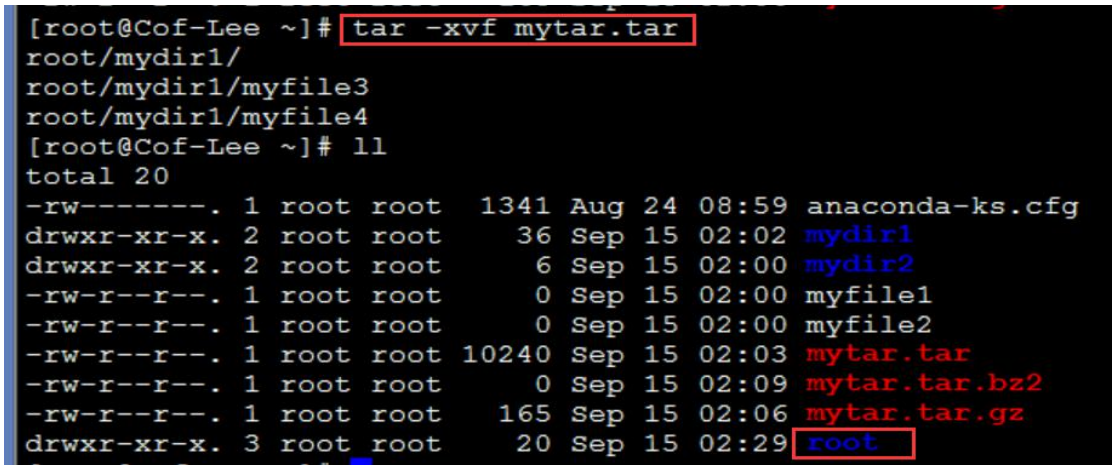
//由于使用.tar.gz 后缀的文件是使用 gzip 压缩归档的，所以要查阅该 tar file 内的文件时，就得加上 z 这个参数了

```
[root@Cof-Lee ~]# tar -ztvf mytar.tar.gz
drwxr-xr-x root/root         0 2018-09-15 02:02 root/mydir1/
-rw-r--r-- root/root         0 2018-09-15 02:02 root/mydir1/myfile3
-rw-r--r-- root/root         0 2018-09-15 02:02 root/mydir1/myfile4
```

上面两图可以看到这个.tar 归档文件里有三个文件，一个目录 root/mydir1，两个普通文件这好像与我们想要的结果不太一样，我们原来是只想打包/root 下的 mydir1 这个目录，为什么/root 这个目录也出现了。因为在归档的时候我们带上了绝对路径，所以连这些目录也一块打包了（只是打包目录名而已，/root 目录下的其他文件没有打包进来）。所以我们打包的时候尽可能地 cd 到要打包的目录或文件的上一级目录里，用相对路径名去打包。
比如我们要归档/root 目录下的 mydir1 目录，只需 cd /root，然后 tar -cvf mytar.tar mydir1

例三：解开归档文件

#tar -xvf xxx.tar

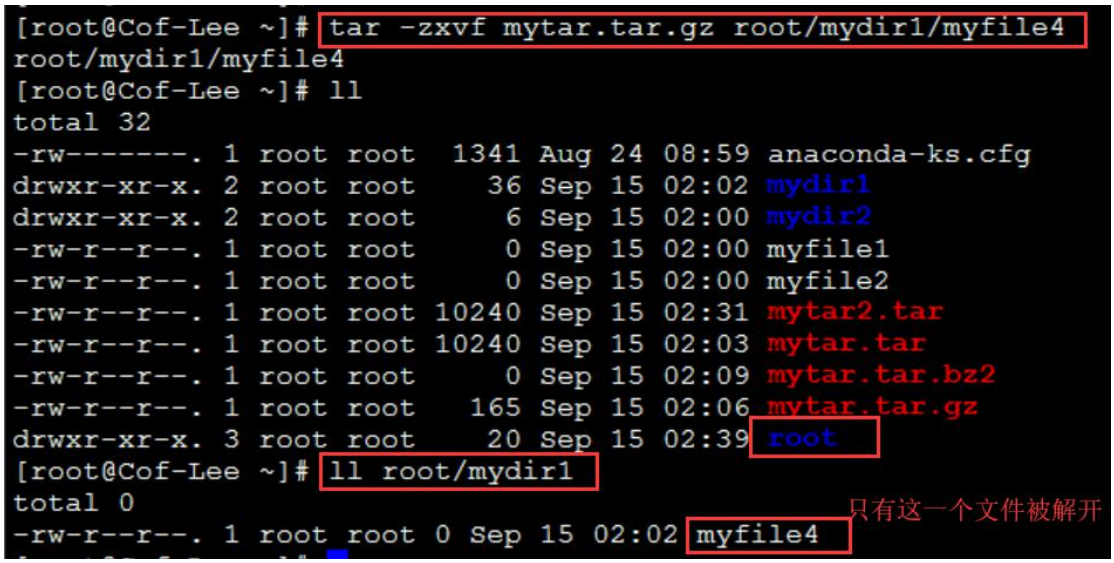


解开的结果是多了一个 root 目录（root 目录里有一个 mydir1 目录，mydir1 目录里有 myfile3 和 myfile4 两个普通文件）

#tar -zxvf xxx.tar.gz //解开以 gzip 压缩了的归档文件

#tar -zxvf xxx.tar.gz dir/filename

可以通过 tar -ztvf 来查阅 tarfile 内的文件内容，如果只要解开其中一个文件，就可以通过这个命令来实现（该文件的上几级目录还是会创建的）



例四：将 mydir1 目录里的所有文件备份下来，并且保存其权限

#tar -zxvpf xx.tar.gz mydir1

//这个 -p（小写）的属性是很重要的，尤其是当我们要保留原本文件的属性时

例五：在 /root 目录中，比 2018/09/01 新的文件才备份

#tar -N "2018/09/01" -zcvf mytar5.tar.gz /root

```
[root@Cof-Lee ~]# tar -N "2018/09/01" -zcvf mytar5.tar.gz /root
tar: Option --after-date: Treating date `2018/09/01' as 2018-09-01 00:00:00
```

例六：要打包 /root 整个目录，但不要 /root 里的某文件 xxfile

#tar --exclude /root/xxfile -zcvf myfile.tar.gz /home/*

```
[root@Cof-Lee ~]# tar --exclude /root/myfile2 -zcvf mytar7.tar.gz /root/*
tar: Removing leading `/' from member names
/root/anaconda-ks.cfg
/root/mydir1/
/root/mydir1/myfile3
/root/mydir1/myfile4
/root/mydir2/
/root/myfile1
/root/mytar2.tar
```

②gzip 命令

#gzip 文件名 //将该文件压缩，压缩后的文件名为原文件名+.gz 且**原文件没**了

```
drwxr-xr-x. 2 root root    6 Sep 15 02:00 mydir2
-rw-r--r--. 1 root root  108 Sep 15 03:10 myfile1
-rw-r--r--. 1 root root    8 Sep 15 03:11 myfile2
[root@Cof-Lee ~]# gzip myfile1
[root@Cof-Lee ~]# ll
total 12
-rw-------. 1 root root 1341 Aug 24 08:59 anaconda-ks.cfg
drwxr-xr-x. 2 root root   36 Sep 15 02:02 mydir1
drwxr-xr-x. 2 root root    6 Sep 15 02:00 mydir2
-rw-r--r--. 1 root root   93 Sep 15 03:10 myfile1.gz
-rw-r--r--. 1 root root    8 Sep 15 03:11 myfile2
[root@Cof-Lee ~]#
```

#gzip -d 文件名 //将该文件解压缩（原文件是以 gzip 压缩的）解压后的文件名少了.gz

```
-rw-r--r--. 1 root root   93 Sep 15 03:10 myfile1.gz
-rw-r--r--. 1 root root   36 Sep 15 03:11 myfile2.gz
[root@Cof-Lee ~]# unzip myfile1.gz
-bash: unzip: command not found
[root@Cof-Lee ~]# gzip -d myfile1.gz
[root@Cof-Lee ~]# ll
total 12
-rw-------. 1 root root 1341 Aug 24 08:59 anaconda-ks.cfg
drwxr-xr-x. 2 root root   36 Sep 15 02:02 mydir1
drwxr-xr-x. 2 root root    6 Sep 15 02:00 mydir2
-rw-r--r--. 1 root root  108 Sep 15 03:10 myfile1
-rw-r--r--. 1 root root   36 Sep 15 03:11 myfile2.gz
```

#gzip -v 文件名　　//压缩时显示压缩比

```
-rw-r--r--. 1 root root  836 Sep 15 03:23 myfile1
-rw-r--r--. 1 root root    8 Sep 15 03:11 myfile2
[root@Cof-Lee ~]# gzip -v myfile1
myfile1:         71.3% -- replaced with myfile1.gz
[root@Cof-Lee ~]# ll
total 12
-rw-------. 1 root root 1341 Aug 24 08:59 anaconda-ks.cfg
drwxr-xr-x. 2 root root   36 Sep 15 02:02 mydir1
drwxr-xr-x. 2 root root    6 Sep 15 02:00 mydir2
-rw-r--r--. 1 root root  266 Sep 15 03:23 myfile1.gz
-rw-r--r--. 1 root root    8 Sep 15 03:11 myfile2
```

#gzip -r 目录名　　//递归压缩该目录下的所有文件

```
drwxr-xr-x. 2 root root   36 Sep 15 03:32 mydir1
drwxr-xr-x. 2 root root    6 Sep 15 02:00 mydir2
-rw-r--r--. 1 root root  836 Sep 15 03:23 myfile1
-rw-r--r--. 1 root root    8 Sep 15 03:11 myfile2
[root@Cof-Lee ~]# ls mydir1
myfile3  myfile4
[root@Cof-Lee ~]# gzip mydir1
gzip: mydir1 is a directory -- ignored
[root@Cof-Lee ~]# gzip -r mydir1
[root@Cof-Lee ~]# ll
total 12
-rw-------. 1 root root 1341 Aug 24 08:59 anaconda-ks.cfg
drwxr-xr-x. 2 root root   42 Sep 15 03:34 mydir1
drwxr-xr-x. 2 root root    6 Sep 15 02:00 mydir2
-rw-r--r--. 1 root root  836 Sep 15 03:23 myfile1
-rw-r--r--. 1 root root    8 Sep 15 03:11 myfile2
[root@Cof-Lee ~]# ls mydir1
myfile3.gz  myfile4.gz
```

#gzip -rd 目录名　　//递归解压缩该目录下的所有文件

```
[root@Cof-Lee ~]# ls mydir1
myfile3.gz  myfile4.gz
[root@Cof-Lee ~]# gzip -dr mydir1
[root@Cof-Lee ~]# ll mydir1
total 8
-rw-r--r--. 1 root root 223 Sep 15 03:32 myfile3
-rw-r--r--. 1 root root 169 Sep 15 03:32 myfile4
```

#gzip -l 压缩文件名　　//显示该压缩文件的详细信息，并不解压文件

```
[root@Cof-Lee ~]# gzip -l myfile1.gz
         compressed        uncompressed  ratio uncompressed_name
                266                 836  71.3% myfile1
```

#gzip -f 文件名　　//-f 或--force　强行压缩文件，不理会文件名称或
// 硬链接是否存在以及该文件是否为符号连接

③zip 命令

首先检查系统是否安装了 zip，如果没有则 要安装 zip 工具和 unzip 工具

```
[root@Cof-Lee ~]# zip
-bash: zip: command not found
[root@Cof-Lee ~]# yum install zip
Resolving Dependencies
--> Running transaction check
```

```
[root@Cof-Lee ~]# yum install unzip
Resolving Dependencies
--> Running transaction check
---> Package unzip.x86_64 0:6.0-19.el7 will be insta
```

#zip 压缩后的文件名 目标文件名 //将目标文件压缩，并命名

//把 myfile1 压缩成 xx.zip 且保留原文件

```
-rw-r--r--. 1 root root  836 Sep 15 03:23 myfile1
-rw-r--r--. 1 root root    8 Sep 15 03:11 myfile2
[root@Cof-Lee ~]# zip xx.zip myfile1
  adding: myfile1 (deflated 71%)
[root@Cof-Lee ~]# ll
total 16
-rw-------. 1 root root 1341 Aug 24 08:59 anaconda-ks.cfg
drwxr-xr-x. 2 root root   36 Sep 15 03:36 mydir1
drwxr-xr-x. 2 root root    6 Sep 15 02:00 mydir2
-rw-r--r--. 1 root root  836 Sep 15 03:23 myfile1
-rw-r--r--. 1 root root    8 Sep 15 03:11 myfile2
-rw-r--r--. 1 root root  404 Sep 15 04:11 xx.zip
```

#zip -r 压缩后文件名 目录 //将目录递归压缩

```
drwxr-xr-x. 2 root root   36 Sep 15 03:36 mydir1
drwxr-xr-x. 2 root root    6 Sep 15 02:00 mydir2
-rw-r--r--. 1 root root  836 Sep 15 03:23 myfile1
-rw-r--r--. 1 root root    8 Sep 15 03:11 myfile2
-rw-r--r--. 1 root root  404 Sep 15 04:11 xx.zip
[root@Cof-Lee ~]# zip -r mydir1.zip mydir1
  adding: mydir1/ (stored 0%)
  adding: mydir1/myfile3 (deflated 62%)
  adding: mydir1/myfile4 (deflated 91%)
[root@Cof-Lee ~]# ll
total 20
-rw-------. 1 root root 1341 Aug 24 08:59 anaconda-ks.cfg
drwxr-xr-x. 2 root root   36 Sep 15 03:36 mydir1
-rw-r--r--. 1 root root  576 Sep 15 04:16 mydir1.zip
drwxr-xr-x. 2 root root    6 Sep 15 02:00 mydir2
```

#unzip 压缩文件名 //将该压缩文件（以 zip 压缩的）解压。原压缩文件仍保留

```
-rw-r--r--. 1 root root  836 Sep 15 03:23 myfile1
-rw-r--r--. 1 root root  404 Sep 15 04:24 myfile1.zip
-rw-r--r--. 1 root root    8 Sep 15 03:11 myfile2
[root@Cof-Lee ~]# unzip myfile1.zip
Archive:  myfile1.zip
replace myfile1? [y]es, [n]o, [A]ll, [N]one, [r]ename: r
new name: myfile1.bak
  inflating: myfile1.bak
[root@Cof-Lee ~]# ll
total 24
-rw-------. 1 root root 1341 Aug 24 08:59 anaconda-ks.cfg
drwxr-xr-x. 2 root root   36 Sep 15 03:36 mydir1
-rw-r--r--. 1 root root  576 Sep 15 04:16 mydir1.zip
drwxr-xr-x. 2 root root    6 Sep 15 02:00 mydir2
-rw-r--r--. 1 root root  836 Sep 15 03:23 myfile1
-rw-r--r--. 1 root root  836 Sep 15 03:23 myfile1.bak
-rw-r--r--. 1 root root  404 Sep 15 04:24 myfile1.zip
-rw-r--r--. 1 root root    8 Sep 15 03:11 myfile2
```

重命名解压后的文件，
解压后原压缩文件仍保留

#unzip -d 目的目录 压缩文件名 //将压缩文件解压至指定目录下

```
-rw-r--r--. 1 root root  404 Sep 15 04:24 myfile1.zip
-rw-r--r--. 1 root root    8 Sep 15 03:11 myfile2
[root@Cof-Lee ~]# unzip -d mydir2 myfile1.zip
Archive:  myfile1.zip
  inflating: mydir2/myfile1
[root@Cof-Lee ~]# ls mydir2
myfile1
```

#zip -m 压缩文件 1 文件 2 //将文件 2 压缩并添加到压缩文件 1 中（不保留文件 2）

```
-rw-r--r--. 1 root root  836 Sep 15 03:23 myfile1.bak
-rw-r--r--. 1 root root  404 Sep 15 04:31 myfile1.zip
-rw-r--r--. 1 root root  257 Sep 15 04:32 myfile2
[root@Cof-Lee ~]# zip -m myfile1.zip myfile2
  adding: myfile2 (deflated 84%)
[root@Cof-Lee ~]# ll
total 20
-rw-------. 1 root root 1341 Aug 24 08:59 anaconda-ks.cfg
drwxr-xr-x. 2 root root   36 Sep 15 03:36 mydir1
-rw-r--r--. 1 root root  576 Sep 15 04:16 mydir1.zip
drwxr-xr-x. 2 root root   21 Sep 15 04:27 mydir2
-rw-r--r--. 1 root root  836 Sep 15 03:23 myfile1
-rw-r--r--. 1 root root  836 Sep 15 03:23 myfile1.bak
-rw-r--r--. 1 root root  587 Sep 15 04:32 myfile1.zip
[root@Cof-Lee ~]#
```

#zip -d 压缩文件 1 文件 2 //将压缩文件 1 里的 文件 2 删除

```
-rw-r--r--. 1 root root  587 Sep 15 04:32 myfile1.zip
[root@Cof-Lee ~]# zip -d myfile1.zip myfile2
deleting: myfile2
```

二十四、Firewalld 防火墙

配置 firewalld 防火墙可以用 firewall-cmd 命令行工具

①firewalld 预定义区域

区域	说明
trusted	允许所有传入流量进入系统
public	与出流量相关（tcp）或与 ssh、dhcpv6-client 预定义服务匹配 的流量可传入
work	与出流量相关（tcp）或与 ssh、dhcpv6-client、ipp-client 预定义服务匹配 的流量可传入
home	与出流量相关（tcp）或与 ssh、dhcpv6-client、ipp-client、mdns、Samba-client 预定义服务匹配 的流量可传入
internal	同 home
external	与出流量相关（tcp）或与 ssh 预定义服务匹配 的流量可传入
dmz	与出流量相关（tcp）或与 ssh 预定义服务匹配 的流量可传入
block	与出流量相关（tcp） 的流量可传入
drop	与出流量相关（tcp） 的流量可传入

网卡的**默认加入区域为 public**

firewalld 命令

②区域管理：

```
#firewall-cmd   --get-zones                          //显示预定义的区域
#firewall-cmd   --get-default-zone                   //显示默认区域
#firewall-cmd   --list-all                           //显示默认区域的所有规则
#firewall-cmd   --set-default-zone=区域名              //设置接口的默认区域
#firewall-cmd   --get-active-zones                   //显示已激活的所有区域
#firewall-cmd   --get-zone-of-interface=ens33        //显示指定接口对应的区域
#firewall-cmd   --zone=work   --change-interface=ens37  //更改接口 ens37 对应的区域
#firewall-cmd   --zone=work   --add-interface=ens33     //为指定区域添加接口
#firewall-cmd   --zone=work   --remove-interface=ens33  //为指定区域移除接口
```

③管理区域中的服务（放行策略）

```
#firewall-cmd   --get-services                          //显示预定义的服务
#firewall-cmd   --zone=work   --list-services          //显示指定区域内允许访问的服务
#firewall-cmd   --zone=work   --add-service=服务名   //添加允许访问的服务
#firewall-cmd   --zone=work   --remove-service=服务名   //删除允许访问的服务
#firewall-cmd   --zone=work   --list-ports             //显示指定区域内允许访问的端口号
#firewall-cmd   --zone=work   --add-port=端口号/tcp    //添加允许访问的端口号，要写连接
                                                       类型：tcp 或 udp
#firewall-cmd   --zone=work   --remove-port=端口号/tcp   //删除允许访问的端口号
#firewall-cmd   --zone=work   --list-icmp-blocks   //显示指定区域内拒绝访问的 icmp 类型
#firewall-cmd   --zone=work   --add-icmp-block=类型    //添加拒绝访问的 icmp 类型
#firewall-cmd   --zone=work   --remove-icmp-block=类型  //删除拒绝访问的 icmp 类型
```

如：**拒绝别人 ping 我**（阻止 icmp-echo-request 包入站）

```
#firewall-cmd  --zone=work  --add-icmp-block=echo-request
```

④IP 伪装与端口转发（PAT）

只对入站包的目的端口进行匹配

```
#firewall-cmd  --zone=work  --add-masquerade      //为指定区域开启 IP 伪装
#firewall-cmd  --zone=work  --query-masquerade    //查看指定区域是否开启 IP 伪装
#firewall-cmd  --zone=work  --add-forward-port=port=8080:proto=tcp:toport=80
                                  //把入站访问 8080 的 tcp 包的目的端口转换为 80
#firewall-cmd --zone=work  --add-forward-port=port=33:proto=tcp:toport=80:toaddr=10.1.1.1
                      //把入站访问 33 端口的 tcp 包的目的端口转换为 80 目的 IP 改为 10.1.1.1
#firewall-cmd  --zone=work  --list-forward-ports      //显示指定区域的端口转发配置
#firewall-cmd  --zone=work  --remove-forward-port=port=8080:proto=tcp:toport=80
                                   //删除该端口转发
```

以上的配置只是运行时的模式，不是永久的，要**永久配置**得写入配置文件中

配置文件为 /etc/firewalld/zones/区域名.xml

命令：

```
#firewall-cmd  --runtime-to-permanet
```

也可以先配置为永久的，再重新加载配置

```
#firewall-cmd  xxxxxxxxxx  --permanent
#firewall-cmd  --reload
```

⑤复杂规则 Rich rule

```
#firewall-cmd  --add-rich-rule='rule family="ipv4"  source address="192.168.1.0/24" service
    name="mysql"  accept'
```

⑥恐慌模式 Panic（会丢弃所有的入站和出站流量）

```
#firewall-cmd  --panic-on       //开启恐慌模式
#firewall-cmd  --panic-off      //关闭恐慌模式
#firewall-cmd  --query-panic    //查询是否处于 Panic 模式
```

二十五、Iptables 防火墙

#yum install iptables-services　　//先安装 iptables
#yum install system-config-firewall-base

iptables 与 firewalld 服务是互斥的，要先关闭 firewalld 服务
#systemctl　stop　firewalld
#systemctl　mask　firewalld
#systemctl　start　iptables
#systemctl　enable　iptables

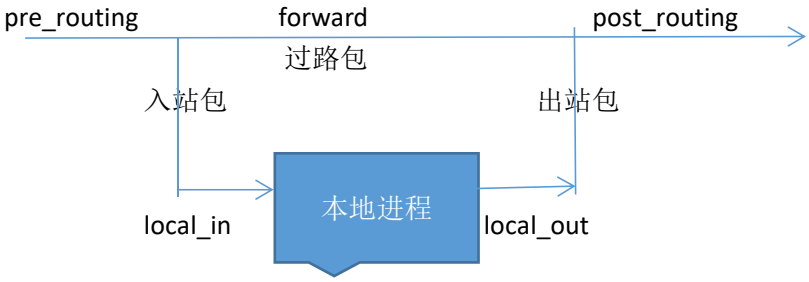
①用 lokkit 工具配置防火墙

#lokkit　--enable
#lokkit　default=server

#lokkit　--list-services　　//查看预定义的服务名
#lokkit　-s　http　-s　https　-s　ftp　-n　//允许访问某些服务，-n 表示不立即启用规则
#lokkit　-p　80:tcp　-p　69:udp　-n　//允许访问某些端口
#lokkit　--update　　//启用已配置但尚未应用的防火墙规则

#lokkit　--list-icmp-types　　//显示预定义的 icmp 类型
#lokkit　--block-icmp=echo-request　　//阻断指定的 icmp 类型入站

②Netfilter 框架：



iptables 工具可操作 Netfilter 中的表模块、链、目标：

表名 table	链名 Chain	说明	目标 target
filter 用于包过滤	INPUT FORWARD OUTPUT	过滤进入系统的包 过滤穿越系统的包 过滤由系统生成的包	ACCEPT REJECT DROP
nat 地址转换	PREROUTING OUTPUT POSTROUTING	地址转换发生在路由之前 DNAT 转换由系统生成的包 地址转换发生在路由之后 SNAT	DNAT REDIRECT SNAT
security		.....	

③iptables 命令工具

iptables 语法：

iptables -t filter　操作　链名　匹配规则　-j 目标

操作：

-A　　在所选链 链尾加入一条规则
-I　3　　以给出的规则号在所选链中插入一条规则
-R　3　　替换
-D　3　　删除

匹配规则：

-s　192.168.1.0/24　　匹配源 ip
-d　192.168.2.0/24　　匹配目的 ip
-i　ens33　　匹配入端口
-o　ens33　　匹配出端口
-p　tcp　　匹配传输层协议
-m　　扩展匹配

例：

1.拒绝外部主动联机的包

#iptables　-A　INPUT　-i　ens33　-p　tcp　--syn　-j　DROP

或：　iptables　-A　INPUT　-i　ens33　-m　conntrack　--ctstate　NEW,INVALID　-j　DROP

2.允许已建立连接或有关联的数据包通过

#iptables -A INPUT -i　ens33　-m conntrack　--ctstate RELATED,ESTABLISHED　-j ACCEPT

3.允许访问特定端口

#iptables　-A INPUT -i ens33 -p tcp　--dport　22 -m conntrack --ctstate　NEW　-j　ACCEPT
-m muliport --dports　20,21,22　-m conntrack　.....

4.允许、限制拒绝 icmp

#iptables　-A INPUT -p　icmp　--icmp-type　echo-request　-j　DROP
-j　ACCEPT
-m limit　--limit 5/s　-i ens33 -j ACCEPT

5.允许数据包转发（Forward）

#iptables　-A　FORWARD　-i　ens33　-s　192.168.1.0/24　-o　ens37　-j　ACCEPT
-d

6.拒绝特定 mac 包的数据访问

#iptables　-A　INPUT　-i　ens33　-m　mac　--mac-source　00:04:0d:33:33:21　-j DROP

SNAT

#iptables -t nat -A POSTROUTING -o ens37 -s 192.68.1.0/24 -j SNAT --to 200.1.1.2

或者转换为出接口的 IP（pppoe 拨号时）

#iptables -t nat -A POSTROUTING -o ens37 -s 192.68.1.0/24 -j　MASQUERADE

DNAT

#iptables -t nat -A PREROUTING -i ens33 -d 200.1.1.2 -p tcp -dport 8080 -j DNAT --tp 10.1.1.1:80

二十六、任务计划

在 CentOS7 中安排一次性的任务可用 at 命令，安排周期性的任务可用 crond 服务

①atd 服务

```
#yum install at                 //安装 at 服务
#systemctl start atd            //开启 atd 服务

#at 时间点     //创建一个任务，在指定的时间点执行，输入 at 加时间点后，
               //可以输入到点后要执行的任务命令
```

例：

```
#at   14:31   02/09/2019          //创建一个计划时间，月/日/年，
                                  //如果不写后面的月日年则默认认为是当天的时间
at> ping -c 3 192.168.33.10       //输入到点后要执行的任务
at> Ctrl+D                        //按 Ctrl+D 退出
```

```
[cof@Cof-Lee ~]$ at 14:31 02/09/2019
at> ping -c 3 192.168.33.10
at> <EOT>
job 10 at Sat Feb  9 14:31:00 2019
```

查看已安排的任务：

```
#at   -l 或 atq
```

```
[cof@Cof-Lee ~]$ atq
11      Sat Feb  9 14:39:00 2019 a cof
[cof@Cof-Lee ~]$ at -l
11      Sat Feb  9 14:39:00 2019 a cof
```

↑任务号　　↑状态

状态为 a 表示还未执行，=表示正在执行

atq 查询如下出现如下提示，表明任务已执行，执行的结果已发到/var/spool/main/cof 文件中，可以用 cat 查看

```
[cof@Cof-Lee ~]$ atq
You have new mail in /var/spool/mail/cof
```

删除任务：

```
#atrm   11      //atrm 加任务号，或者 at  -d  加任务号
```

```
[cof@Cof-Lee ~]$ at -l
11      Sat Feb  9 14:39:00 2019 a cof
[cof@Cof-Lee ~]$ atrm 11
```

②crond 服务

用户可以使用 crontab 命令安排自己的 crond 周期性任务

crontab 任务安排命令格式：

例：　**5　0　*　*　*　　rm　-rf　~/temp/***

字段：1　2　3　4　5　6

字段说明

1: 表示一小时中的哪一分钟（0～59）

2：表示一天中的哪一小时（0～23）*星号表示不限具体的小时，即每个小时

3：表示一月中的哪一天（1～31）*星号表示不限具体的天数，即每一天

4：表示一年中的哪一个月（1～12）*星号表示不限具体的月份，即每个月

5：表示一周中的哪一天（0～7）0 和 7 都表示周日，*星号表示不限星期几，即每天

6：表示任务具体的命令（rm　-rf　~/tmp/*）

本例中的含义：在每天的 0 时 5 分执行 rm -rf ~/temp/*，即清空用户的临时文件

创建用户自己的周期性任务

#crontab -e　　//输入此命令后，以 Vi 工具进行编辑任务格式

比如：

15　1　*　*　*　rm　-rf　~/temp/*

一行表示一个任务，可以写多行

然后保存退出即可。

查看用户自己的周期性任务：

#crontab　-l

```
[cof@Cof-Lee spool]$ crontab -l
27 15 * * * echo hello
```

#crontab　-u　cof　-l　　//查看指定用户的 crontab 任务

删除用户自己的周期性任务：

#crontab　-r　　//不管任务有几行（几个）都会被删除

```
[cof@Cof-Lee spool]$ crontab -l
27 15 * * * echo hello

You have new mail in /var/spool/mail/cof
```

如果任务执行一次，则系统会发一份邮件给执行任务的用户

可以查看/var/spool/mail/cof 文件

二十七、磁盘限额

磁盘限额就是限制每个用户的磁盘使用空间，防止个别用户使用过多的磁盘空间而影响了系统的正常运行和其他用户的使用。
对于 ext3、ext4 文件系统，磁盘限额的配置工具由 quota 软件包提供
对于 xfs 文件系统，磁盘限额的配置工具由 xfs_progs 软件包提供
限制用户的磁盘使用空间可以从 2 方面控制，一是限制用户可以使用的磁盘大小，一是限制用户可以拥有的文件数

②EXT3、EXT4 文件系统磁盘限额配置

首先挂载磁盘时就要做磁盘限额的检查
比如要对 sdb2 这个磁盘分区进行限制，要修改/etc/fstab 文件的挂载项

```
/dev/sdb2          /home        ext4          defaults,usrquota,grpquota      0   0
```

保存退出后要重新挂载文件系统：

```
#mount   -o   remount   /home

#quotacheck   -cmvug      /home
#quotaon   -avug
```

然后就可以对指定用户进行限制了：

```
#setquota   -u   cof   200M   300M   2000   2500   /home
//设置 cof 用户可以使用 200MB 磁盘空间（/home 目录），最大 300M，可以拥有的文件数 2000，最大 2500

#setquota   -u   cof   200M   300M   0   0   /home       //文件数限制为 0 0  表示不限制
#quota   -u   cof        //查看 cof 用户的磁盘限额
#setquota   -u   -p   cof   Dick   /home          //以用户 cof 为参考对 Dick 用户进行限额配置
```

对用户组（整个组）进行磁盘限额配置

```
#setquota   -g   staff   1G   2G   20K   25K   /home
#setquota   -g   -p   staff   student    /home          //以 staff 组为参考对 studnet 组进行配置
#quota   -gv   staff                //查看 staff 组的限额配置

#repquota   -augv                   //查看磁盘限额报告
```

②xfs 文件系统磁盘限额配置

```
#vi /etc/fstab                  //编辑为：
/dev/sdb2          /www      xfs      defaults,uquota,gquota     0   0
```

保存退出后，先卸载文件系统再重新挂载

```
#umount   /www
#mount   /www
```

对用户限制：

#xfs_quota -x -c 'limit -u bsoft=200M bhard=300M isoft=2000 ihard=2500 cof' /www
#xfs_quota -x -c 'limit -u bsoft=200M bhard=300M cof' /www
查看用户限额配置：
#xfs_quota -c 'quota -uv cof' /www
#xfs_quota -c 'quota -i -uv cof' /www

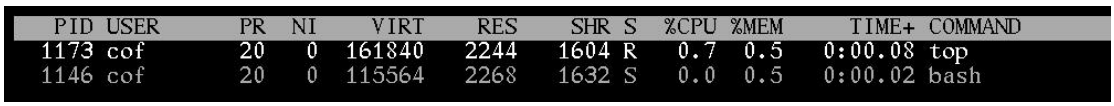
对组限制：
#xfs_quota -x -c 'limit -g bsoft=200M bhard=300M isoft=2000 ihard=2500 staff' /www
查看组限额配置：
#xfs_quota -c 'quota -gv staff' /www
#xfs_quota -c 'quota -i -gv staff' /www

查看用户、组的限额报告
#xfs_quota -x -c 'report -ug' /www
#xfs_quota -x -c 'report -i -ug' /www

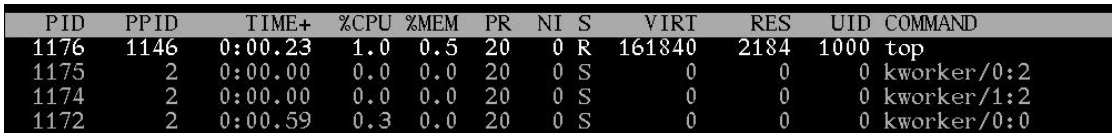
二十八、系统性能查看

①TOP

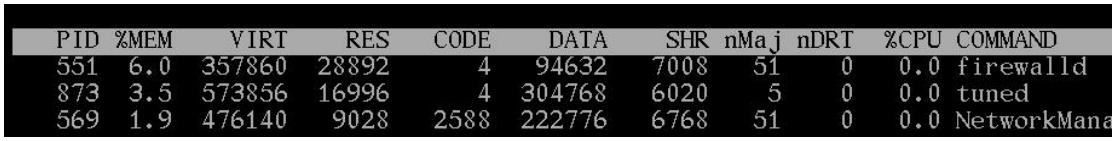
输入 top 命令可以显示当前正在运行的进程及其相关信息，g1 至 g4 的风格分别如下：



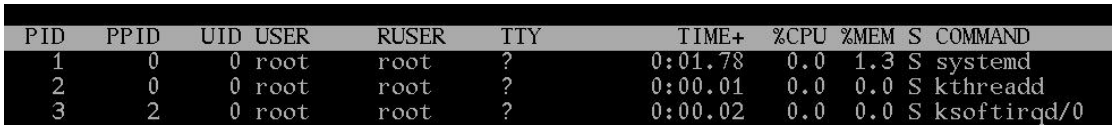
```
  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
 1173 cof       20   0  161840   2244   1604 R   0.7   0.5   0:00.08 top
 1146 cof       20   0  115564   2268   1632 S   0.0   0.5   0:00.02 bash
```



```
  PID   PPID     TIME+  %CPU  %MEM  PR  NI S    VIRT    RES   UID COMMAND
 1176   1146   0:00.23   1.0   0.5  20   0 R  161840   2184  1000 top
 1175      2   0:00.00   0.0   0.0  20   0 S       0      0     0 kworker/0:2
 1174      2   0:00.00   0.0   0.0  20   0 S       0      0     0 kworker/1:2
 1172      2   0:00.59   0.3   0.0  20   0 S       0      0     0 kworker/0:0
```



```
  PID %MEM     VIRT    RES    CODE    DATA    SHR nMaj nDRT  %CPU COMMAND
  551  6.0   357860  28892       4   94632   7008   51    0   0.0 firewalld
  873  3.5   573856  16996       4  304768   6020    5    0   0.0 tuned
  569  1.9   476140   9028    2588  222776   6768   51    0   0.0 NetworkMana
```



```
  PID   PPID   UID USER     RUSER    TTY          TIME+  %CPU %MEM S COMMAND
    1      0     0 root     root     ?        0:01.78   0.0  1.3 S systemd
    2      0     0 root     root     ?        0:00.01   0.0  0.0 S kthreadd
    3      2     0 root     root     ?        0:00.02   0.0  0.0 S ksoftirqd/0
```

输入 top 命令后，即进入 top 的交互模式，在交互模式下可以输入字符命令进行其他操作：

命令（不需回车）	说明
g1 至 g4	切换显示风格，共有 4 种 g1 g2 g3 g4
u	输入 u 然后提示输入用户名，即可查看指定用户的进程
k	输入 k 然后提示输入进程的 pid，即可结束该进程
d 或 s	修改刷新显示的时间间隔，默认为 3 秒更新一次
q	退出 top 程序
y	切换开关，对正在运行的进程加亮显示或者不加亮
i	切换开关，显示闲置进程和僵死进程 或者不显示
M	按%MEM 字段进行排序，降序
N	按 PID 字段排序，降序
P	按%CPU 排序，降序
T	按 TIME+排序，降序

Top 输出项含义：

字段	含义
PID	进程 id
USER	进程所有者 id
PR	进程优先执行的顺序，值越小越优先
NI	进程 nice 值，正值为低优先级，负值为高优先级
VIRT	进程使用的虚拟内存总量，单位 KB，virt=swap+res
RES	进程使用的物理内存大小，单位 KB
SHR	SHR 共享内存大小，单位 KB
S	进程状态，D：不可中断睡眠，R：运行，S：睡眠，T：跟踪/停止 Z：僵尸进程

%CPU	进程在本次刷新时间间隔内的 CPU 时间占用百分比
%MEM	物理内存使用百分比
TIME+	使用 CPU 的总时间，单位为 0.01 秒
COMMAND	使用的命令
PPID	父进程 pid
RUSER	实际用户名
UID	用户 ID
TTY	控制终端
SWAP	交换区使用大小，单位 KB

②vmstat

vmstat 命令显示进程队列、内存交换区、块 I/O 和 CPU 活动信息

#vmstat -a　　//显示所有的（活跃和非活跃内存）

#vmstat -a -S M　　//-S M 表示以指定的单位显示，M 为 MB，**默认为 KB**

```
[cof@Cof-Lee ~]$ vmstat
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 1  0      0 218516   2072 142036    0    0    24     2   43   51  0  0 100  0  0
```

```
[cof@Cof-Lee ~]$ vmstat -a
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free  inact active   si   so    bi    bo   in   cs us sy id wa st
 0  0      0 218500  65260  78764    0    0    23     2   43   51  0  0 100  0  0
```

vmstat 输出项含义：

输出项		说明
procs	r	在运行队列中等待运行的进程数
	b	等待 I/O 的进程数
memory	swpd	当前使用的交换空间
	free	当前空闲的物理内存
	buff	用作缓冲的内存大小
	cache	用作缓存的内存大小
	inact	非活跃的内存大小
	active	活跃的内存大小
swap	si	每秒从交换区写到内存的大小
	so	每秒从内存写到交换区的大小
io	bi	每秒读取块设备的块数
	bo	每秒写入块设备的块数
system	in	每秒的中断数，包括时钟中断
	cs	每秒的环境（上下文）切换次数
CPU	us	用户进程执行时间的百分比
	sy	系统进程执行时间的百分比
	id	空闲时间　百分比
	wa	等待 I/O 时间　百分比
	st	管理程序为另一个虚拟进程提供服务而　等待虚拟 CPU 的百分比

③iostat （yum install sysstat　　//要先安装 sysStat）

iostat 命令用于输出 CPU 与磁盘 I/O 相关的统计信息

```
[cof@Cof-Lee ~]$ iostat
Linux 3.10.0-862.el7.x86_64 (Cof-Lee)   02/09/2019      _x86_64_        (2 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.06    0.00    0.27    0.02    0.00   99.65

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda               1.47        57.20         2.75     178158       8559
scd0              0.10         0.96         0.00       2978          0
```

iostat 输出项含义：

%user	在用户级别运行所使用的 cpu 百分比
%nice	高优先级进程（nice 小于 0）使用的 cpu 百分比
%system	在核心级别（kernel）运行所使用的 cpu 百分比
%iowate	CPU 等待硬件 I/O 所占用 cpu 百分比
%steal	当管理程序为另一个虚拟进程提供服务而等待虚拟 cpu 的百分比
%idle	CPU 空闲时间的百分比
Devices	硬盘设备
tps	每秒钟物理设备的 I/O 传输总量
kB_read/s	每秒从驱动器读入的数据量，单位为　块/s
kB_wrtn/s	每秒向驱动器写入的数据量，单位为　块/s
kB_read	读入的数据总量，单位为 KB
kB_wrtn	写入的数据总量，单位为 KB

④mpstat （yum install sysstat）

mpstat 命令报告与 cpu 相关的统计信息

```
[cof@Cof-Lee ~]$ mpstat
Linux 3.10.0-862.el7.x86_64 (Cof-Lee)   02/09/2019      _x86_64_        (2 CPU)

11:32:00 AM  CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest  %gnice   %idle
11:32:00 AM  all    0.06    0.00    0.26    0.02    0.00    0.01    0.00    0.00    0.00   99.66
```

mpstat 输出项含义：

%usr	在用户级别运行所使用的 cpu 时间百分比
%nice	高优先级进程（nice 小于 0）使用的 cpu 时间百分比
%sys	在核心级别（kernel）运行所使用的 cpu 时间百分比
%iowate	CPU 等待硬件 I/O 所占用 cpu 时间百分比
%irq	中断操作占用 cpu 时间百分比
%soft	softirq 操作占用 cpu 时间百分比
%steal	管理程序为另一个虚拟进程提供服务而等待虚拟 cpu 的百分比
%idle	显示 cpu 在空闲状态占用 cpu 总时间的百分比
intr/s	cpu 每秒接收到的中断数

二十九、网络诊断工具

①Ping 工具

命令格式：　（包大小为字节数，超时单位为毫秒）

#ping　-c 次数　-s　包大小　-W　超时　-I 出接口　目标 ip

例：

```
[cof@Cof-Lee ~]$ ping -c 4 -s 1400 -W 500 -I ens33  192.168.33.10
PING 192.168.33.10 (192.168.33.10) from 192.168.33.1 ens33: 1400(1428) bytes
1408 bytes from 192.168.33.10: icmp_seq=1 ttl=128 time=0.696 ms
1408 bytes from 192.168.33.10: icmp_seq=2 ttl=128 time=0.540 ms
1408 bytes from 192.168.33.10: icmp_seq=3 ttl=128 time=0.542 ms
1408 bytes from 192.168.33.10: icmp_seq=4 ttl=128 time=0.544 ms

--- 192.168.33.10 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3009ms
rtt min/avg/max/mdev = 0.540/0.580/0.696/0.070 ms
```

②Tracepath

命令格式：

#tracepath　-n　-p 出接口 目标 ip　　//-n 为不解析主机名

例：

```
[cof@Cof-Lee ~]$ tracepath -n -p ens33 192.168.33.10
 1?: [LOCALHOST]                                         pmtu 1500
 1:  192.168.33.10                                         0.291ms reached
 1:  192.168.33.10                                         0.369ms reached
     Resume: pmtu 1500 hops 1 back 1
```

③SS　（Socket Statistics）选项可组合

#ss　-a　　//显示所有的套接字

#ss　-t　　//显示 tcp 连接

#ss　-u　　//显示 udp 连接

#ss　-n　　//端口号以数字形式显示

#ss　-p　　//显示出使用该套接字的进程名

例：

```
[cof@Cof-Lee ~]$ ss -tpn
State      Recv-Q Send-Q   Local Address:Port                 Peer Address:Port
ESTAB      0      0          192.168.33.1:22                 192.168.33.10:1322
[cof@Cof-Lee ~]$ ss -tp
State      Recv-Q Send-Q Local Address:Port                 Peer Address:Port
ESTAB      0      0        192.168.33.1:ssh                192.168.33.10:novation
```

④netstat　（yum　install　net-tools）

#netstat　-a　　//显示所有的套接字

#netstat　-t　　//显示 tcp 连接

#netstat　-u　　//显示 udp 连接

#netstat　-n　　//端口号以数字形式显示

#netstat -p //显示出使用该套接字的进程名

例：

```
[cof@Cof-Lee ~]$ netstat -t
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0     52 Cof-Lee:ssh             192.168.33.10:novation  ESTABLISHED
[cof@Cof-Lee ~]$ netstat -t -n
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0     52 192.168.33.1:22         192.168.33.10:1322      ESTABLISHED
```

```
[cof@Cof-Lee ~]$ sudo netstat -t -n -p
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0     52 192.168.33.1:22         192.168.33.10:1322      ESTABLISHED 1307/sshd: cof [p
```

⑤lsof （需要赋予管理员权限,sudo）yum install lsof

lsof 的第一个字母为小写的 L，不是大写的 i

#lsof -i :21 //查看指定端口号运行的程序

#lsof -i @192.168.22.11 //查看指定 ip 使用的端口（该 ip 为本地主机的 ip）

#lsof -n -i TCP@192.168.1.1 //查看指定 ip 使用的 TCP 端口

#lsof -n -i UDP@192.168.1.1 //查看指定 ip 使用的 UDP 端口

```
[cof@Cof-Lee ~]$ sudo lsof -i :22
COMMAND  PID USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
sshd     858 root    3u  IPv4  18217      0t0  TCP *:ssh (LISTEN)
sshd     858 root    4u  IPv6  18234      0t0  TCP *:ssh (LISTEN)
sshd    1307 root    3u  IPv4  23990      0t0  TCP Cof-Lee:ssh->192.168.33.10:novation (ESTABLIS
HED)
sshd    1312  cof    3u  IPv4  23990      0t0  TCP Cof-Lee:ssh->192.168.33.10:novation (ESTABLIS
HED)
```

三十、文件传输工具

①Tftp 客户端工具　yum　install　tftp

#tftp　192.168.33.10　　//登录 tftp 服务器，进入交互模式
进入交互模式后：
> mode　binary　　//切换传输模式为二进制传输模式，netascii 为文本传输模式
> status　　//查看当前模式、状态
> get　srcfile　dstfile　　//下载远地文件 srcfile，保存在本地并命名为 dstfile
> put　srcfile　dstfile　　//上传本地文件 srcfile，保存在服务器上并命名为 dstfile

②FTP 客户端工具（lftp）　yum　install　lftp

登录 FTP 服务器：
1.以匿名身份登录
#lftp　192.168.1.1
2.以用户名，密码登录
#lftp　-u　cof,passwd　192.168.1.1
3.先登录，后输入用户名和密码
#lftp　192.168.1.1
> user　cof
passwd: xxxxxx
登录后的交互操作：
> get　xxx.txt　-o　dst.txt　//下载单个文件，并命名为 dst.txt
　　//如果不写 -o dst.txt 则默认用原文件名
> mget　*.txt　　//使用通配符下载多个文件
> get　-c　xxx.iso　　//下载单个大文件，-c 参数表示支持续传
> pget　-c　-n　3　xxx.iso　//使用多线程下载文件，-n 3 表示使用 3 个线程
> put　src.txt　-o　dst.txt　//上传文件，把 src.txt 上传到服务器，并命名为 dst.txt
> mirror　-c　远地目录名　本地目录名　//镜像远地目录到本地，即下载整个目录
> mirror　-R　本地目录名　//镜像本地目录到服务器，即上传整个目录
>rm　-r　-f 文件名　　//删除服务器上的某个文件

③Wget 下载工具

#wget　http://xxxx/pub/xxx.txt　　//下载网站的上某个文件
#wget　-r　-np　-nd　http://xxxx/pub　　//下载网站的某个目录里的所有文件
　　// -np 表示不遍历父目录，-nd 表示不在本地重新构建目录结构

三十一、输出重定向

文件描述符：（可以看作是设备描述符，在 Linux 里一切皆文件）

0　　表示标准输入，如键盘

1　　表示标准输出，如屏幕，console

2　　表示标准错误，也是输出到屏幕或 console

输出重定向：

>　　表示把输出覆盖到文件中

>>　　表示把输出追加到文件后面

例：

```
#echo   "xxx xxxx" > test.txt          //把"xxx xxx"字符串追加到 test.txt 文件后
#echo   "str" >> test.txt             //把"str"字符串覆盖到 test.txt 文件
```

我们在执行某个命令时，会有输出，如果没有错误，则正常的输出到屏幕上的文字为标准输出，如果出现错误则为标准错误，从键盘向屏幕中输入字符为标准输入。

```
#CMD   > xx.txt    //表示在输入某个命令后，再把标准输出追加到 xx.txt 文件中
                   //这时在屏幕上看不到正常的输出了，因为被重定向到 xx.txt 文件中了
                   //  在大于号>之前没有数字的话默认表示 1（标准输出）

#CMD   2> err.txt          //表示输入某个命令后，如果出错，就把标准错误追加到 err.txt
                           //文件中，大于号>前的 2 表示标准错误

#CMD   >xx.txt   2>&1          //表示把标准错误也重定向到标准输出中，再把标准输出
                               //重定向到 xx.txt 文件中
```

三十二、环境变量

环境变量说得简单点就是 shell 程序在解释我们输入的命令时，会去几个指定的路径下查找目标命令，那些路径就是环境变量。

#echo $PATH //查看当前的环境变量

从上图可见，当前用户（root 用户）默认的环境变量为以上 5 个目录，目录之间用冒号隔开。

/usr/local/sbin

/usr/local/bin

/usr/sbin

/usr/bin

/root/bin

Windows 系统在查找命令时，还会在当前所处的目录下查找，而 Linux 不会在当前目录下查找，所以就算目标程序就在当前目录下，直接输入命令时，还是会提示找不到目标命令。所以要输入./ 这个./表示当前目录的意思。

我们可以把目标程序所在目录也放到环境变量里

#vi /etc/profile //在该文件最后添加如下 2 行

PATH="$PATH:/mypath" //表示在原来的 path 基础上再加一个目录/mypath

export PATH

保存

#source /etc/profile //保存后，再重新加载一下 profile

说到 profile，这里还得再讲一点儿

profile 或 bashrc 之类的文件是 shell 程序的配置文件，当 shell 程序启动后，加载这些文件，以配置一些用户个人或全局的额外的变量或参数。这些文件一共有 5 个（或种）

这些文件的加载顺序在 CentOS 中就是从上往下的顺序。

/etc/profile.d/* //表示在/etc/profile.d/目录下的所有文件

/etc/profile //文件

/etc/bashrc //文件

~/.bashrc //表示在当前用户家目录下的.bashrc 文件

~/.bash_profile //表示在当前用户家目录下的.bash_profile 文件

三十三、sudo 用户权限分配

普通用户能够执行的命令有限，要想执行更多的命令（以 root 身份执行）时，可以把该用户或用户组设置为 sudo 用户，然后指定其能够执行的命令即可。

由 root 用户去进行以下操作：

```
#visudo              //编辑 sudo 配置文件，操作同 vi
                     //按下 i 键即可输入配置文本，随便在配置文件的某个地方输入都行
coflee      ALL=(root)      CMD
//用户名     以 root 的身份    能执行的命令

Tom     ALL=(root)      ALL          //表示 Tom 用户能以 root 的身份执行所有的命令

//如果不想赋予用户所有的命令，可以自定义命令集合，然后放到用户 CMD 那里

Cmnd_Alias      VIHTTP  =  /usr/bin/vi  /etc/httpd/*
Cmnd_Alias      SYSTEMCTL  =  /bin/systemctl  start  *,  /bin/systemctl  stop  *,
/bin/systemctl restart *,  /bin/systemctl status *

//表示定义 2 个别名，代理后面的那几个命令，再放到用户的 CMD 那里，表示该用户能以 root 身份执行这些命令。每个命令后都要有一个逗号，再接一个空格。最后一个命令可以不用逗号

// 命令要写完全路径，命令后可以跟目录，表示该命令只能操作该目录下的文件
//命令后可以接参数，表示只能执行这些参数，未列出的参数不能执行
coflee   ALL=(root)   VIHTTP,  SYSTEMCTL
%wheel   ALL=(root)   ALL
//用户组前有%百分号，用户前没有百分号
```

配置完后，保存即可。

```
#sudo  -l            //列出当前用户能执行的 sudo 命令
```

三十四、Service 命令

Centos7 已经不用 service 命令了，这里只是为了兼容低版本的系统才讲解一下。

```
#service   --status-all                  //查看所有服务状态
#service   服务名   reload               //重新加载配置
#service   服务名   try-reload           //仅当服务运行时，才重新加载配置
#service   服务名   start                //启动该服务
#service   服务名   restart              //重启该服务
#service   服务名   stop                 //停止该服务
```

```
#chkconfig   --list              //查看每个服务的运行级别，就是说该服务在每个 runlevel 下是自启动还是 非自启动
```

```
[root@Centos7 ~]#chkconfig --list

Note: This output shows SysV services only and does not include native
      systemd services. SysV configuration data might be overridden by n
      systemd configuration.

      If you want to list systemd services use 'systemctl list-unit-file
      To see services enabled on particular target use
      'systemctl list-dependencies [target]'.

netconsole      0:off   1:off   2:off   3:off   4:off   5:off   6:off
network         0:off   1:off   2:on    3:on    4:on    5:on    6:off
```

看上图最后一行，表示 network 这个服务，在 runlevel 为 0,1,6 时关闭，在 runlevel 为 2,3,4,5 时开启

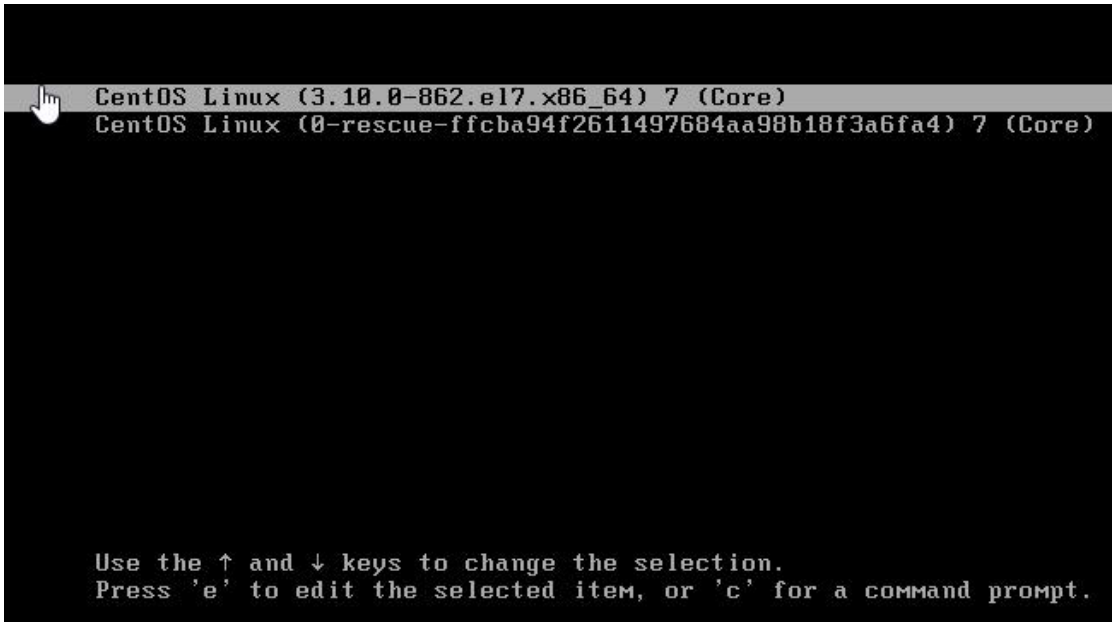
```
#chkconfig   服务名   on              //表示设置该服务在当前运行级别时，自启动
#chkconfig   服务名   off             //表示设置该服务在当前运行级别时，不自启动
#chkconfig   --level   35   服务名   on      //表示设置该服务在 runlevel3,5 时自启动
```

chkconfig 设置服务是否随开机自启动的做法是：把自启的服务配置为一个 shell 脚本文件，放在/etc/rc.d/目录下（该目录下每个 rc*目录对应一个 runlevel），开机时加载并运行相应的 runlevel 的脚本文件，然后脚本文件再去启动目标服务。具体的不多讲了

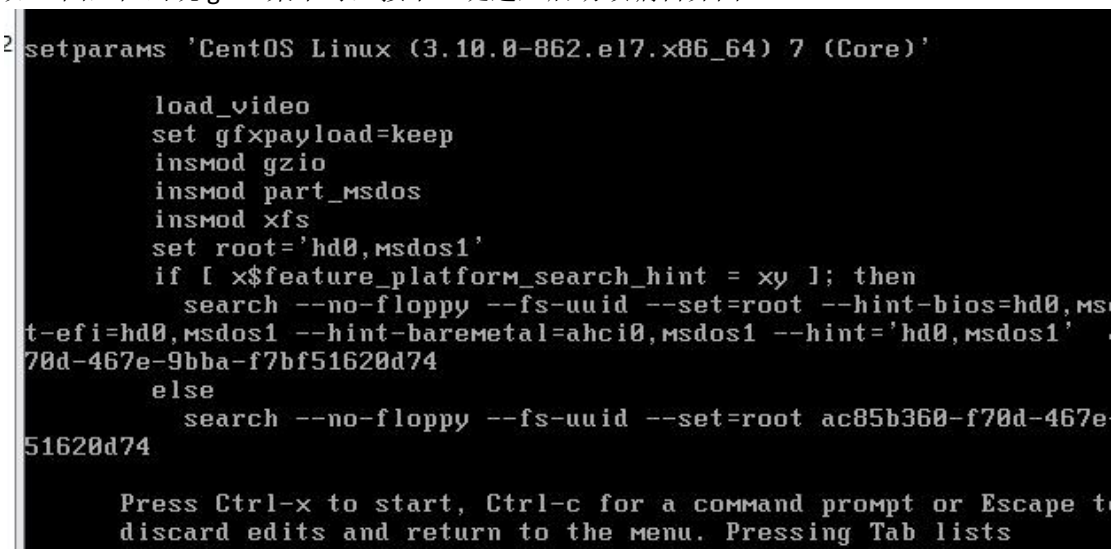
```
[root@Centos7 ~]#ll /etc/rc.d/
total 4.0K
drwxr-xr-x. 2 root root  70 Dec 10 18:21 init.d
drwxr-xr-x. 2 root root  45 Dec 10 18:21 rc0.d
drwxr-xr-x. 2 root root  45 Dec 10 18:21 rc1.d
drwxr-xr-x. 2 root root  45 Dec 10 18:21 rc2.d
drwxr-xr-x. 2 root root  45 Dec 10 18:21 rc3.d
drwxr-xr-x. 2 root root  45 Dec 10 18:21 rc4.d
drwxr-xr-x. 2 root root  45 Dec 10 18:21 rc5.d
drwxr-xr-x. 2 root root  45 Dec 10 18:21 rc6.d
-rwxr-xr-x. 1 root root 491 Dec 10 19:59 rc.local
```

三十五、重置 root 密码

先重启系统（可以用其他帐号登录，再重启，如果没用户能登录时，可以按下 Ctrl+Alt+Del 组合键 重启）



如上图，在出现 grub 菜单时，按下 e 键进入启动项编辑界面



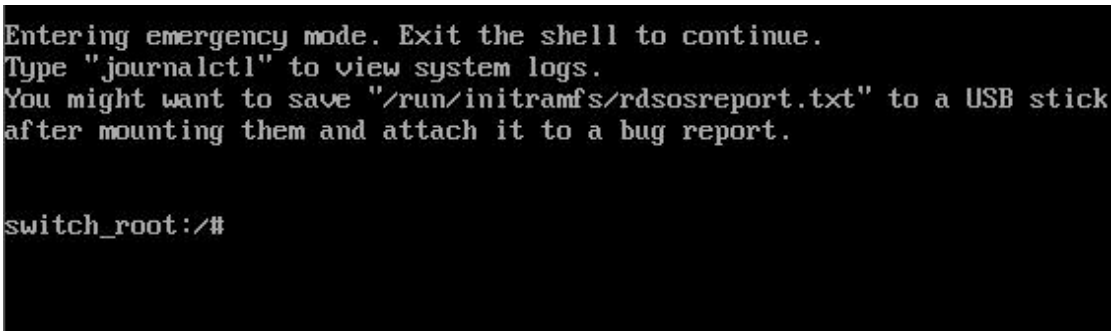
按下 e 键后，如上图，按箭头键往下翻，找到 linux16 /vmlinuz-3.10.0-xxxxx 这一行，就是我们第 1 章设置屏幕分辨率时要找的那一行，在那行的末尾添加 rd.break console=tty0

一行显示不了一个单词时，会自动添加一个反斜杠\，这个不要管它。

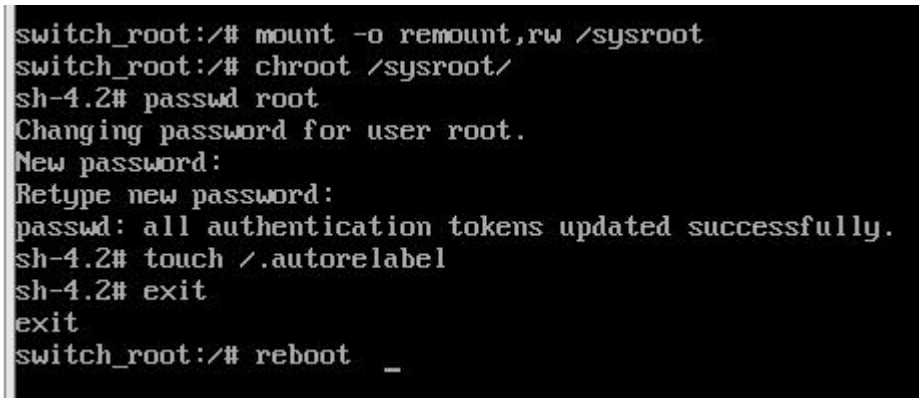
添加完后，按下 Ctrl+X 进入恢复模式

添加的 rd.break console=tty0 表示在内核初始化后中断系统 systemd 的执行，并提供一个无须 root 口令登录的调试 shell 界面。

按下 Ctrl + X 后，进入出下界面



此时按以下步骤操作：



#mount -o remount,rw /sysroot //重挂载/sysroot 目录

#chroot /sysroot/ //切换 root 文件系统

#passwd root //重置 root 用户的口令

…

… //输入新的口令

#touch /.autorelabel //重新标记 SELinux 上下文

#exit

#reboot //重启系统

等系统重启后，就可以使用新的口令登录了。

三十六、开启 SNMP 服务

#yum install net-snmp
#systemctl enable snmpd
#systemctl start snmpd

#snmpd -v //查看版本号

#vi /etc/snmp/snmpd.conf //修改配置
//找到 com2sec notConfigUser default public 这一行，修改为：
com2sec notConfiguser 10.2.2.251 pubxxxxxx
//版本为 v2c，允许连接的管理站为 10.2.2.251，团体字为 pubxxxxxx
view systemview included .1 //表示允许查询的 oid 为.1 开头的所有节点
保存，重启服务

#systemctl restart snmpd
#firewall-cmd --add-port=161/udp --permanent //防火墙放行 161 端口
#firewall-cmd --reload

三十七、Centos7 加入 Windows 域

```
#yum  install  sssd  realmd  oddjob  oddjob-mkhomedir  adcli  samba-common
     samba-common-tools  krb5-workstation  openldap-clients  policycoreutils-pythen  -y

#systemctl  enable  sssd
#systemctl  start  sssd

#vi  /etc/resolv.conf                //编辑 dns 解析
 search  xxx.com
 nameserver  10.1.1.250              //查询 xxx.com 时使用指定的 dns，指向域里的 dns

#realm  join  --user=domainUser  xxx.com  -v              //加域
...              //输入用户名和密码，加域成功


#realm  list              //查看加域情况


#id  user@xxx.com              //查看域用户

//默认情况下使用域用户时，还要加上 xxx.com 的后缀，若不想加后缀，可进行如下配置
#vi  /etc/sssd/sssd.conf
 use_fully_qualified_name=False          //不使用完全名称
 fallback_homedir=/home/%u
保存，重启 sssd


//给相应的 domain 组分配权限，这个组是在 Windows 域控上存在的组
#vi  /etc/sudoers.d/sudoers
%组名@xxx.com        ALL=(root)        自定义权限


//退域
#realm  leave  xxx.com
```

三十八、自动记录历史命令

每当用户退出登录时，就会记录其本次登录进行的操作命令，并保存在单一的文件里，文件名可自定义，比如 user@IP_login_time

#vi /etc/profile //在文件尾部添加以下脚本：

```
#set auto logging user history
history
USER=`whoami`
USER_IP=`who -u am i 2>/dev/null| awk '{print $NF}'|sed -e 's/[()]//g'`
if [ "$USER_IP" = "" ]; then
    USER_IP=`hostname`
fi
if [ ! -d /var/log/history ]; then
    mkdir /var/log/history
    chmod 777 /var/log/history
fi
if [ ! -d /var/log/history/${LOGNAME} ]; then
    mkdir /var/log/history/${LOGNAME}
    chmod 700 /var/log/history/${LOGNAME}
fi
export HISTSIZE=4096
DT=`date +"%Y%m%d_%H:%M:%S"`
export HISTFILE="/var/log/history/${LOGNAME}/${USER}@${DT}_${USER_IP}"
```

保存，

#source /etc/profile

文件名的时间为登录的时间，IP 为客户端的公网 IP

三十九、SeLinux

SELinux（Security Enhanced Linux）是一个强制访问控制系统，是从 Linux2.6 内核开始提供的基于“域-类型”模型的强制访问控制安全系统，开启了 SELinux 后，系统进程对文件资源的访问又多了一层访问权限控制。

SELinux 的控制思路是：对进程进行分类，对资源进行分类，然后指定某类进程只能访问某类资源，这样即使进程是以 root 身份运行的，它也只能访问指定的那类资源。如果不开启 selinux，当以 root 身份运行某个进程时，该进程就能访问所有的资源，这是非常危险的。

①启用 selinux

```
#cat   /etc/sysconfig/selinux        //查看 selinux 的主配置文件，这是个链接文件，实际指向
                                     //   /etc/selinux/config 这个文件，只有 2 个参数要配置
SELINUX = enforcing                  //selinux 的启用状态
.....
SELINUXTYPE = targeted               //selinux 的类型
```

参数说明：

```
SELINUX = disabled          //不启用 selinux
          permissive        //启用，违反策略时不会限制访问，只会记录日志
          enforcing         //启用，违反策略时会阻止访问目标文件
SELINUXTYPE = targeted      //只对主要的网络服务进程访问资源时  进行限制
              strict        //对整个系统进程  都进行访问资源的限制
```

如果不开启 Selinux，那么设置 SELINUX = disabled 就行了

如果要开启 Selinux，那么推荐设置为 SELINUX = enforcing， SELINUXTYPE = targeted

```
#getenforce                  //查看是否启用 selinux
#setenforce   permissive     //临时设置为 permissive 的状态
```

Selinux 对所有的文件（资源）都赋予一个叫 type 的文件类型标签

对所有的进程也都赋予一个叫 domain 的标签

然后，进程对所有资源的访问都是可以基于策略去设置的。

```
#ls   -Z 或 ls   --context          //查看文件的标签
```

重新对文件系统赋予标签（谨慎操作）

```
#/sbin/fixfiles   relabel
```

或

```
#touch   /.autorelabel
```

然后 重启系统

②查看安全上下文

selinux 对资源分配的标签也叫 安全上下文（Security Context）

安全上下文（标签）由 5 个元素组成：

user: //表示能够访问它的用户

role: //表示该资源的角色

type: //表示该资源的类型

sensitivity:

category:

#ls -Z //查看文件的安全上下文

#ps -Z //查看进程的安全上下文

#id -Z //查看当前用户的安全上下文

例：ll -Z //普通文件只显示前 4 个元素

-rw-r--r--. cof cof system_u:object_r:user_home_t:s0 Centos7.pptx

↑user ↑role ↑type ↑sensitivity

③端口标签（传输层的端口）

#semanage port -l //列出所有预定义的端口资源标签

#semanage port -l | grep http //查看与 http 相关的端口资源标签

#semanage port -a -t http_port_t -p tcp 8080 //添加端口标签

//表示给 8080 端口打上 http_port_t 的标签，这样能访问该类标签资源的进程就能访问该 8080 端口了

#semanage port -d -t http_port_t -p tcp 8080 //删除端口标签

④文件标签

#semanage fcontext -l | grep "/var" //查看/var 目录及其子目录所有文件的标签

#chcon -t httpd_sys_content_t 文件名 //修改文件标签中的 type 类型

//表示把该文件的 type 类型改为 httpd_sys_content_t 标签，这样之后，能访问该类型资源的进程也就能访问该文件了

#chcon -t xxxxx_t 目录 //如果是修改目录标签，则要用-R 参数表示递归

#semanage fcontext -a -t xxxx_t '/var/www/html(/.*)?' //修改目录的默认

//标签，-a 表示永久生效

#restorecon -Rv /var/www/html //恢复该目录下所有文件的标签为该目录的

//默认标签

⑤Sebool 策略

表示网络服务进程开启的功能或能够访问的资源

#getsebool -a | grep httpd //查看 selinux 布尔值

#semanage boolean -l | grep http //查看 selinux 布尔值

#semanage boolean -l -C //查看修改过的布尔值

#semanage -P ftpd_use_nfs=1 //设置 selinux 布尔值，1 为开，0 为关

或者 semanage -P ftpd_use_nfs on //设置为开，on 为开，off 为关

当我们 mv 移动文件到新的目录下时，其 selinux 标签不变，只有 copy 时才会生成新的标签

⑥查看标签权限

#yum install setools-console

#sesearch --allow -t httpd_sys_content_t //查看该类资源可被哪些进程类 所访问

```
root @ ~ #  sesearch --allow -t httpd_sys_content_t |more
Found 759 semantic av rules:
   allow sectoolm_t file_type : dir { ioctl read getattr lock
   allow mailman_domain file_type : filesystem getattr ;
   allow avahi_t file_type : filesystem getattr ;
   allow mdadm_t file_type : filesystem getattr ;
   allow telepathy_domain file_type : filesystem getattr ;
   allow mock_t file_type : filesystem getattr ;
   allow privoxy_t file_type : filesystem getattr ;
```

#sesearch --allow -s httpd_t //查看该类进程可以访问哪些类型的资源

```
root @ ~ # sesearch  --allow -s  httpd_t |more
Found 1867 semantic av rules:
   allow sepgsql_client_type sepgsql_sysobj_t : db_tuple { use select
   allow httpd_t httpd_tmp_t : dir { ioctl read write create getattr s
k unlink link rename add_name remove_name reparent search rmdir open }
   allow httpd_t mailman_data_t : lnk_file { read getattr } ;
   allow syslog_client_type device_t : dir { ioctl read getattr lock s
 } ;
```

⑦查看 selinux 阻止访问日志

#yum install setroubleshoot

//setroubleshoot 将 selinux 错误信息写入/var/log/messages 文件中

#tail /var/log/messages | grep setroubleshoot //查看日志

#sealert -a /var/log/audit/audit.log //查看日志