

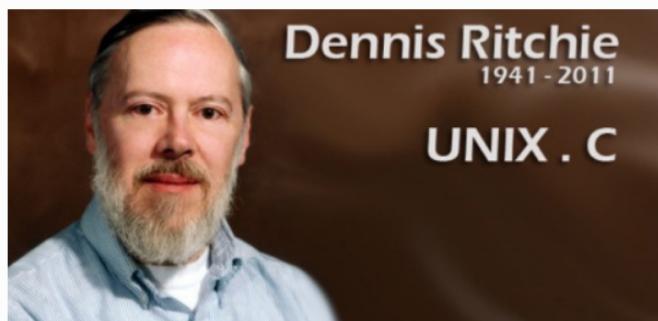
Linux 及 Shell 编程

讲课老师(墨竹)

第一章 Linux 概述

1. 了解 Unix 系统

Unix 是较早被广泛使用的计算机操作系统之一，它的第一版于1969年由 Ken Thompson 在 AT&T 贝尔实验室实现，1973年 Ken Thompson 与 Dennis Ritchie 用C语言重写了 Unix 的第三版内核



- Unix是一个强大的多用户、多任务操作系统。
- UNIX的商标权由国际开放标准组织 (The Open Group) 所拥有。
- UNIX操作系统是商业版，需要收费，价格比Microsoft Windows正版要贵一些。

2. 了解 Linux 发展历史

Linux是一套自由加开放源代码的类Unix操作系统，诞生于1991年10月5日（第一次正式向外公布），由芬兰学生Linus Torvalds和后来陆续加入的众多爱好者共同开发完成。

Linux这个词本身只表示Linux内核，但实际上人们已经习惯了用Linux来形容整个基于Linux内核，并且使用GNU工程各种工具和数据库的操作系统。

GPL协议：

Richard M. Stallman 于1984年创立自由软件体系GNU，拟定普遍公用版权协议（General Public License，简称GPL），今天Linux的成功就得益于GPL协议。

所有GPL协议下的自由软件都遵循着Richard M. Stallman的"Copyleft"(非版权)原则：即自由软件允许用户自由拷贝、修改和销售，但是对其源代码的任何修改都必须向所有用户公开。

GNU 计划

GNU计划和自由软件基金会FSF(the Free Software Foundation)是由Richard M. Stallman 于1984 年一手创办的。旨在开发一个类似UNIX 并且是自由软件的完整操作系统

到上世纪90 年代初，GNU 项目已经开发出许多高质量的免费软件，其中包括有名的emacs 编辑系统、bash shell 程序、gcc 系列编译程序、gdb 调试程序等等。这些软件为Linux 操作系统的开发创造了一个合适的环境。这是Linux 能够诞生的基础之一，以至于目前许多人都将Linux 操作系统称为“GNU/Linux”操作系统。



自由软件之“父”——Richard. M. Stallman

自由软件运动的精神领袖、美国工程院院士，GNU 计划以及自由软件基金会的创立者



林纳斯·本纳第克特·托瓦兹 (Linus Benedict

Torvalds, 1969年12月28日-)，芬兰赫尔辛基人，著名的电脑程序员，Linux内核的发明人及该计划的合作者

Linux 系统特点

开放性（开源）、多用户、多任务、良好的用户界面、优异的性能和稳定性以及多用户多任务的特点

多用户：多个用户，在登陆计算机（操作系统），允许同时登陆多个用户进行操作

多任务：多个任务，允许用户同时进行多个操作任务

注意：Windows 属于单用户多任务，Linux 属于多用户多任务

3. Linux的应用领域

- 服务器系统
 - Web应用服务器、数据库服务器、游戏服务器、接口服务器、DNS、FTP等等；
- 嵌入式系统
 - 路由器、防火墙、手机、PDA、IP 分享器、交换器、家电用品的微电脑控制器等等
- 高性能运算、计算密集型应用
 - Linux有强大的运算能力。IBM的Watson超级计算机就是使用了Linux系统
- 桌面应用系统
 - 很多桌面操作系统的底层也是Linux
- 移动手持系统
 - 安卓系统就是基于Linux

4. Linux版本

Linux 的版本继承了 Unix 的版本定制规则，分为内核版本和发行版本

内核版本：内核就是一个核心，其他软件都基于这个核心，不能直接使用，内核版本统一在 <http://www.linux.org> 发布

发行版本：由各个 Linux 发行商发布，Linux 发行商有权选择 Linux 的内核版本。常见的 Linux 的发行版本：
RedHat、CentOS、Debian、Ubuntu

内核版本分为稳定版和开发版，区分方式是根据次版本的奇偶判定，奇数为开发版，偶数为稳定版



- Debian

Debian运行起来极其稳定，这使得它非常适合用于服务器。

- redhat :

这是第一款面向商业市场的Linux发行版。它有服务器版本，支持众多处理器架构。

全球最大的linux发行厂商，功能全面、稳定。

- ubuntu :

Ubuntu是Debian的一款衍生版，侧重于它在这个市场的应用，在服务器、云计算、甚至一些运行的移动设备上很常见。

- centos :

CentOS是一款企业级Linux发行版，它使用红帽企业级Linux中的免费源代码重新构建而成。这款重构版完全去掉了注册商标以及Binary程序包方面一个非常细微的变化。

- Fedora

Fedora同是一款非常好的发行版，有庞大的用户论坛，软件库中还有为数不少的软件包。Fedora同样使用YUM来管理软件包。

第二章 安装Linux

1. 安装虚拟机软件

- 什么是虚拟

虚拟机（Virtual Machine）指通过软件模拟的具有完整硬件系统功能的、运行在一个完全隔离环境中的完整计算机系统。

- 常用的虚拟机

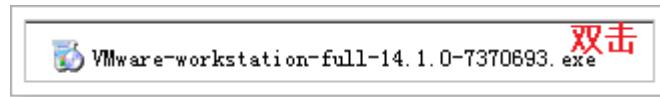
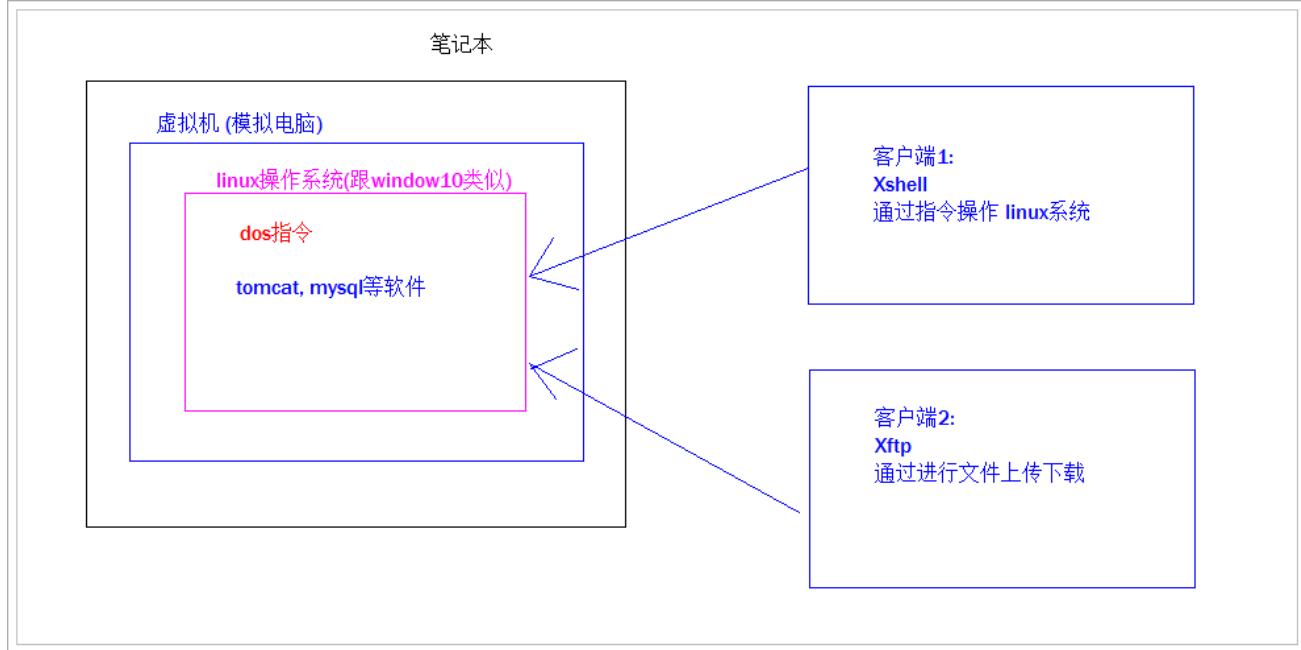
常用的虚拟机软件 主要包括：VMware Workstation、 VirtualBox、 Virtua* PC

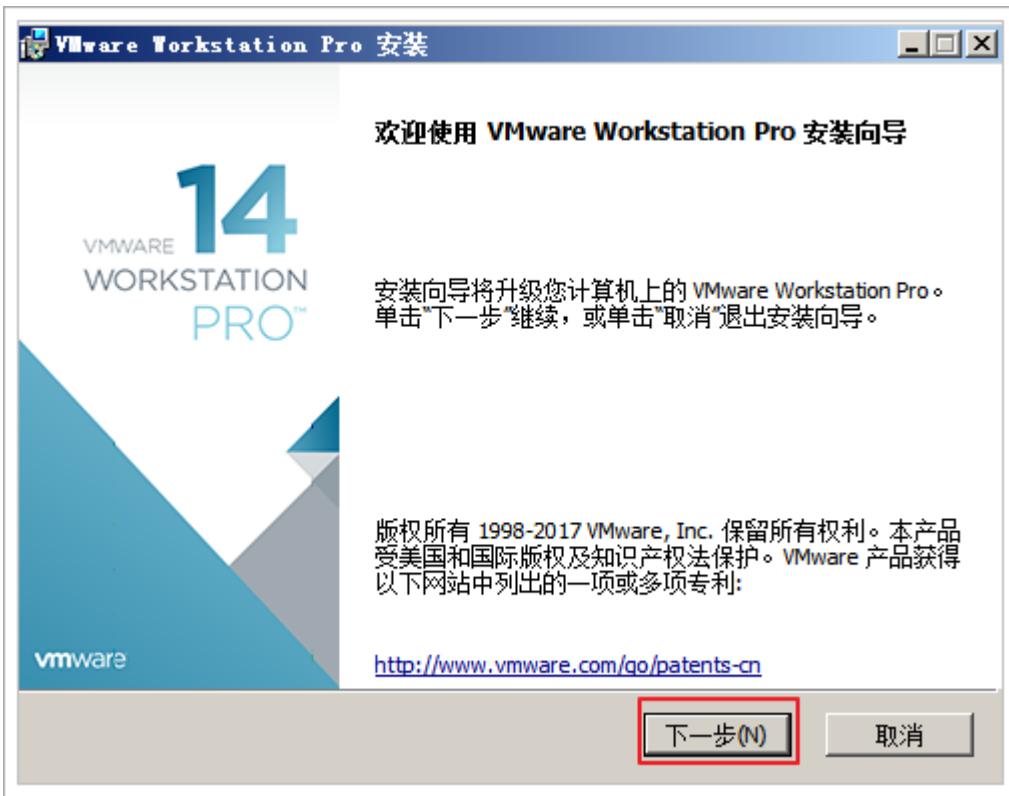
VMware Workstation: 是VMware公司销售的商业软件产品之一。该工作站软件包含一个用于英特尔x86相容电脑的虚拟机套装，其允许用户同时创建和运行多个x86虚拟机

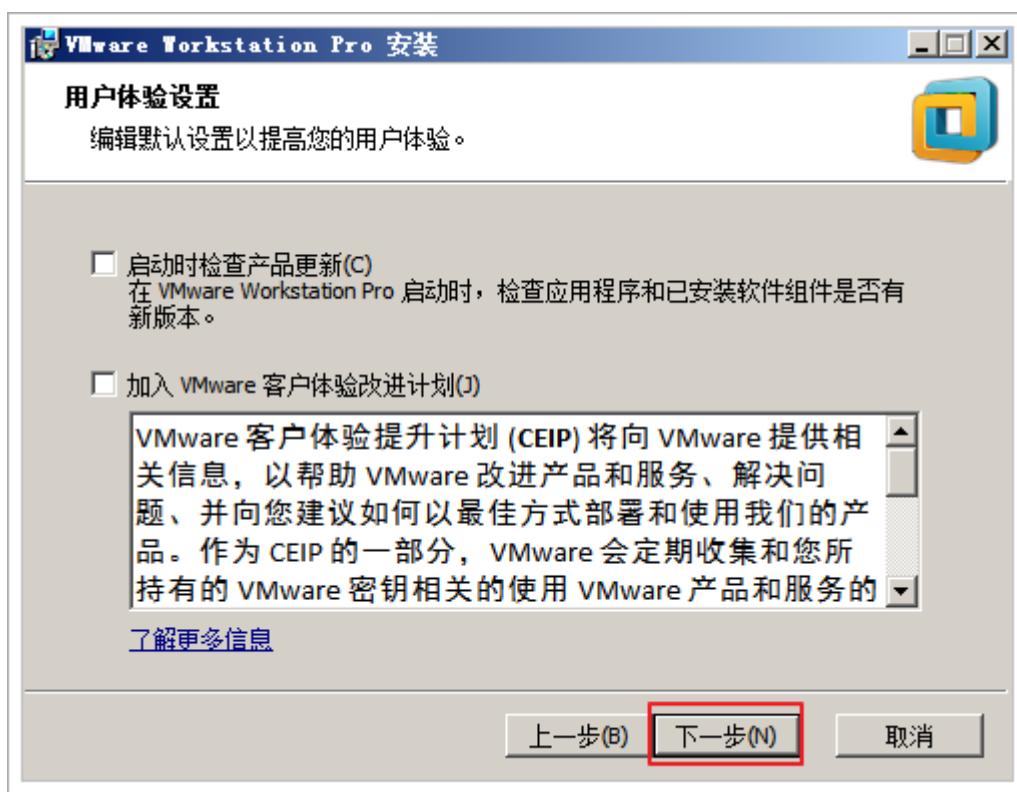
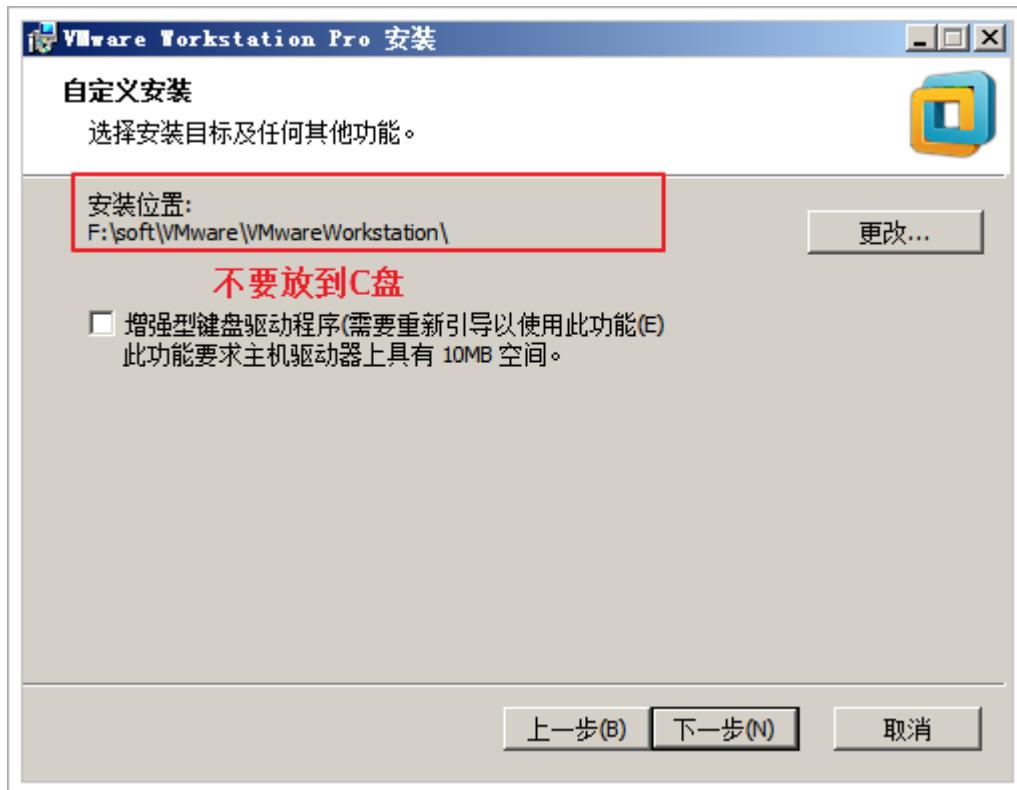
VirtualBox: VirtualBox 是一款开源 **虚拟机软件**。VirtualBox 是由德国 Innotek 公司开发，由Sun Microsystems公司出品的软件，使用 **Qt** 编写，在 Sun 被 **Oracle** 收购后正式更名为 Oracle VM VirtualBox

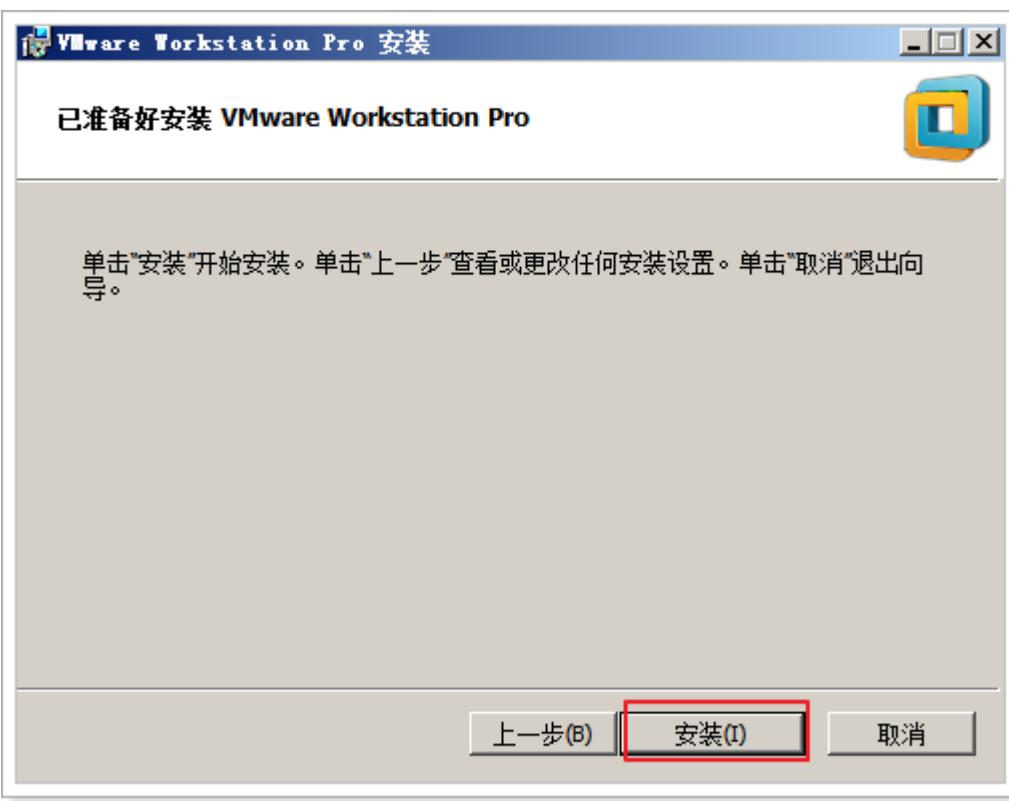
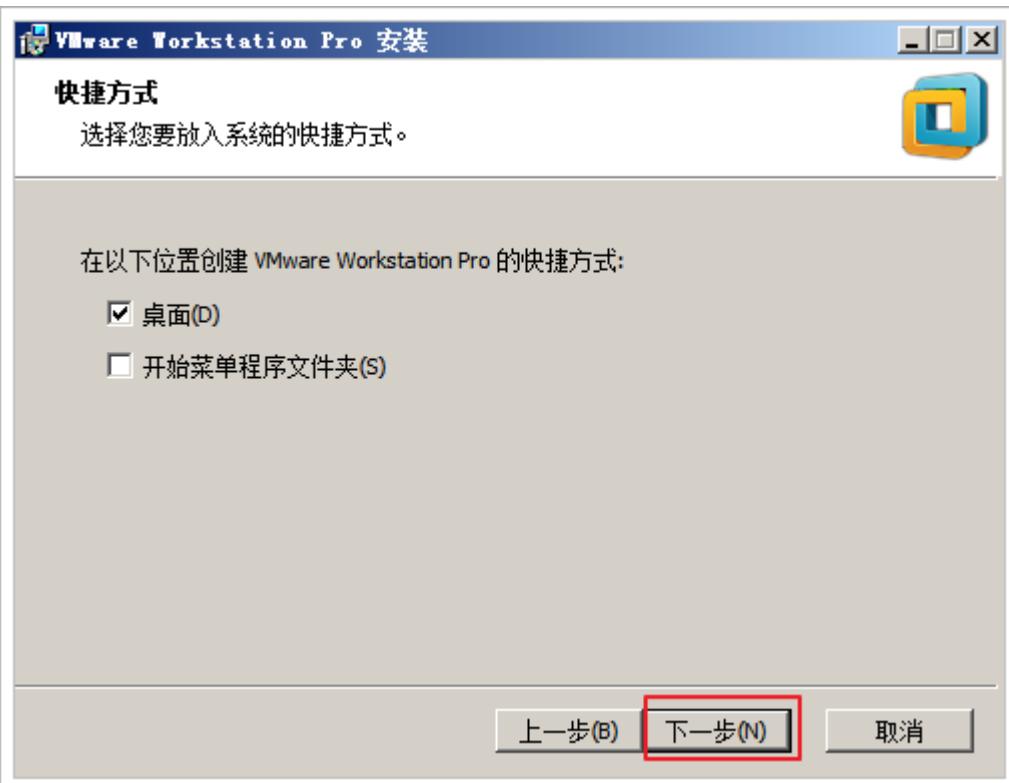
Virtua* PC : 是Microsoft 最新的虚拟化技术。主要适合做微软自己产品的服务

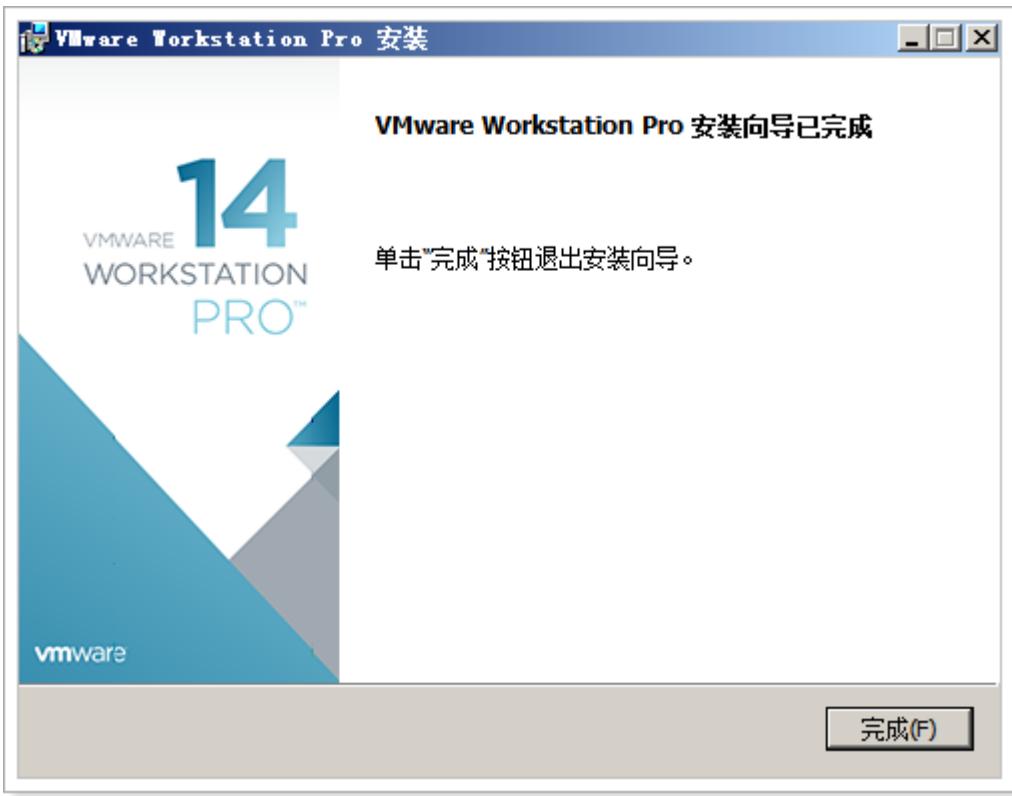
- **我的电脑与虚拟机**





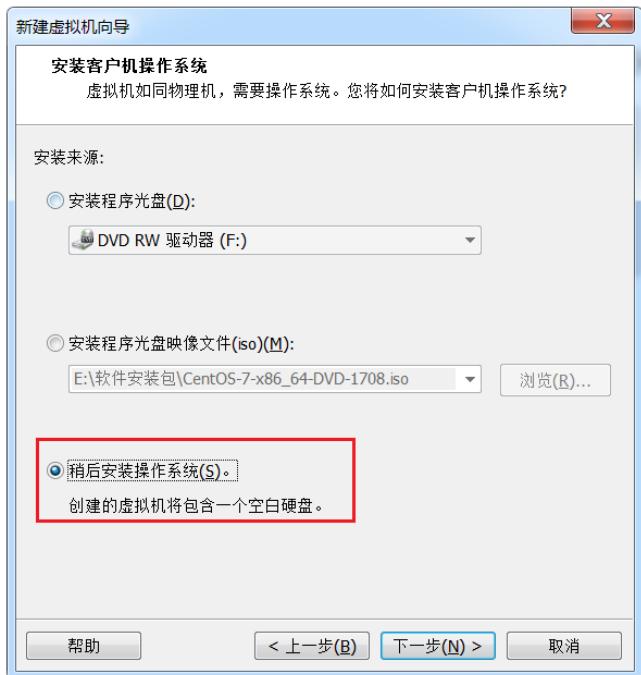


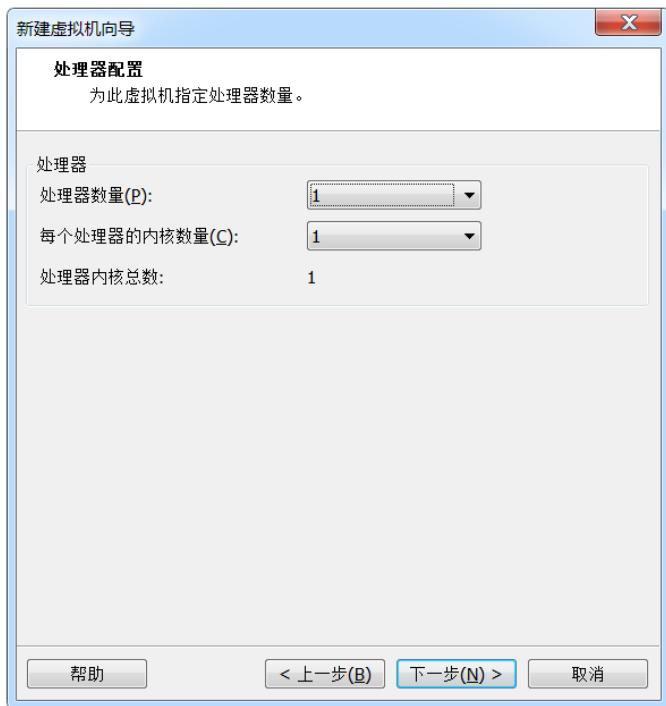


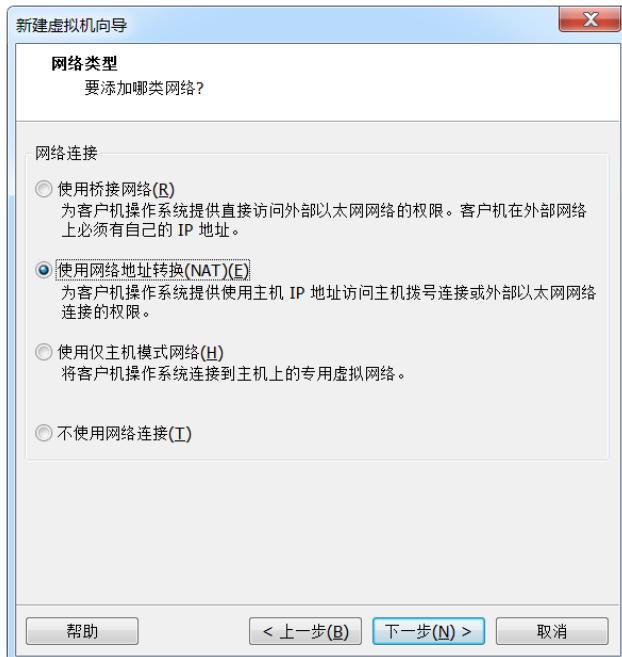
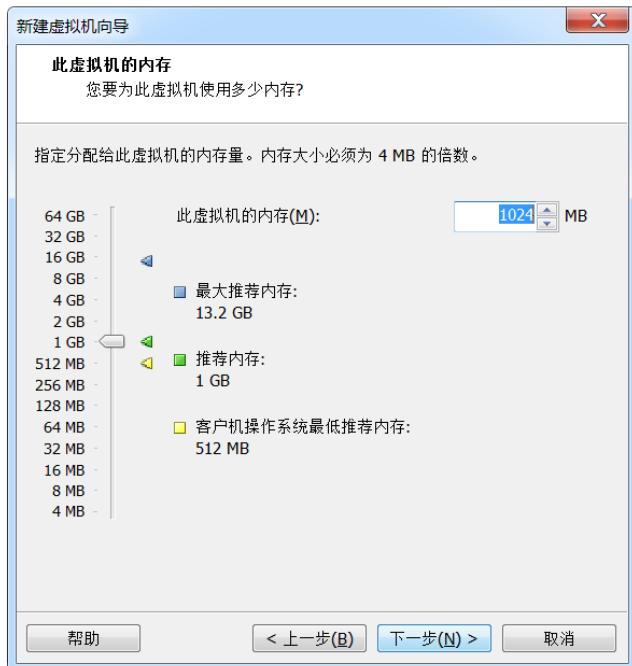


2. 安装Centos 7系统

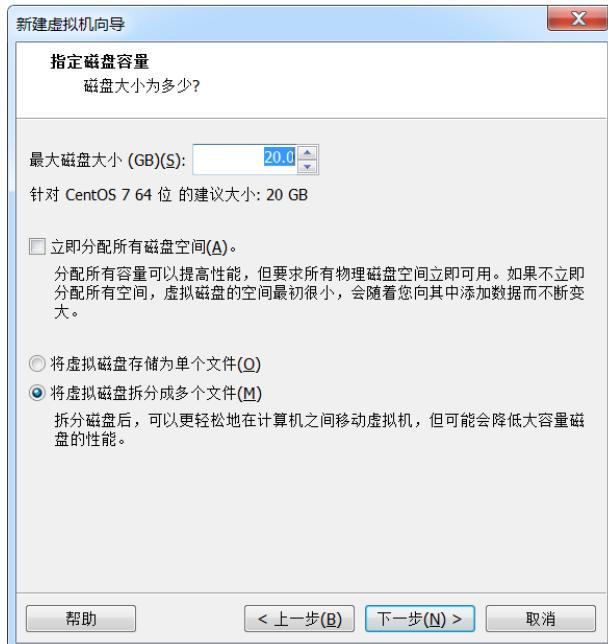
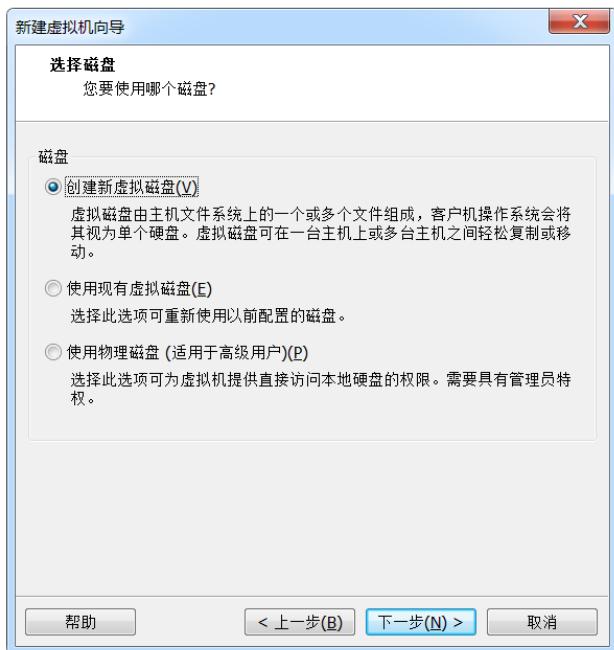


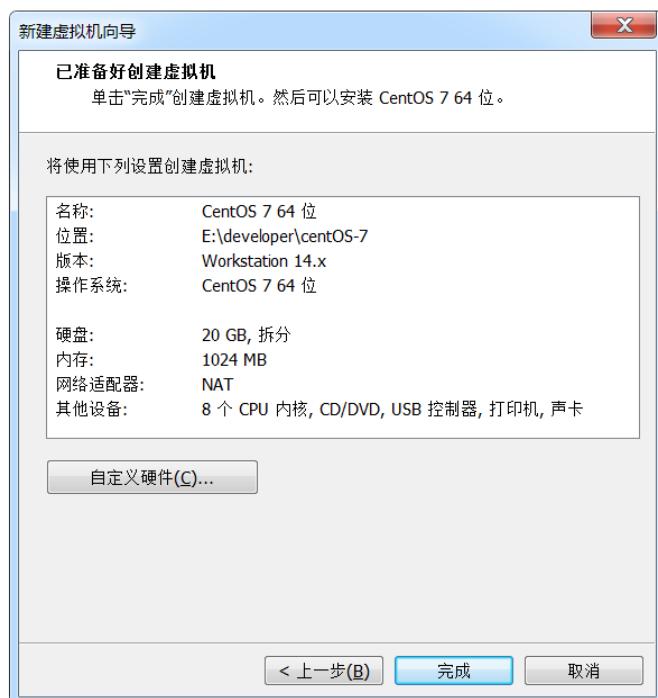
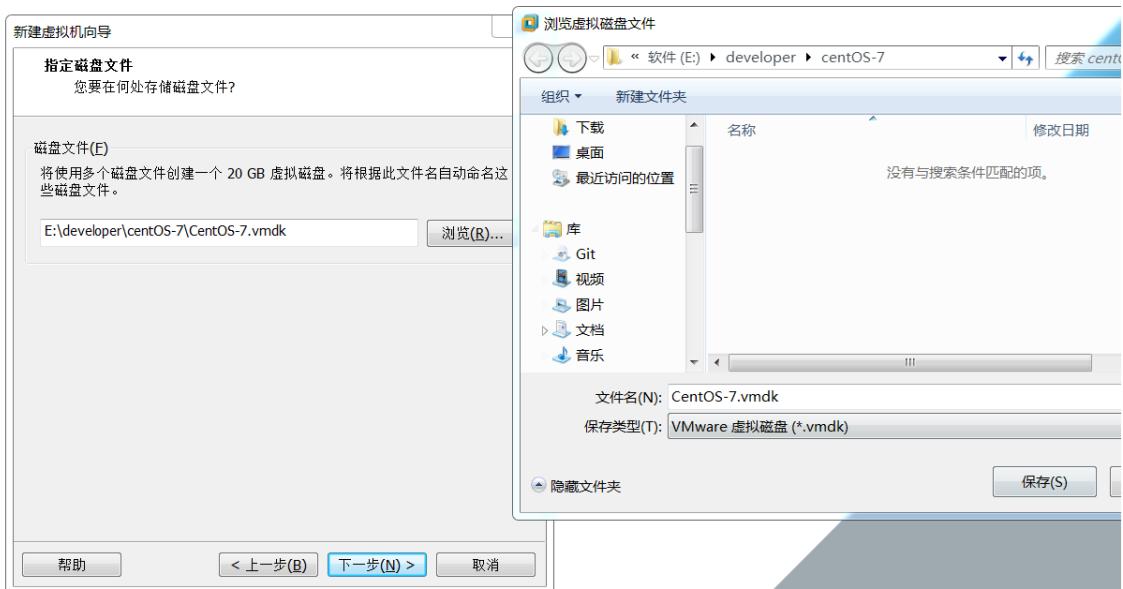


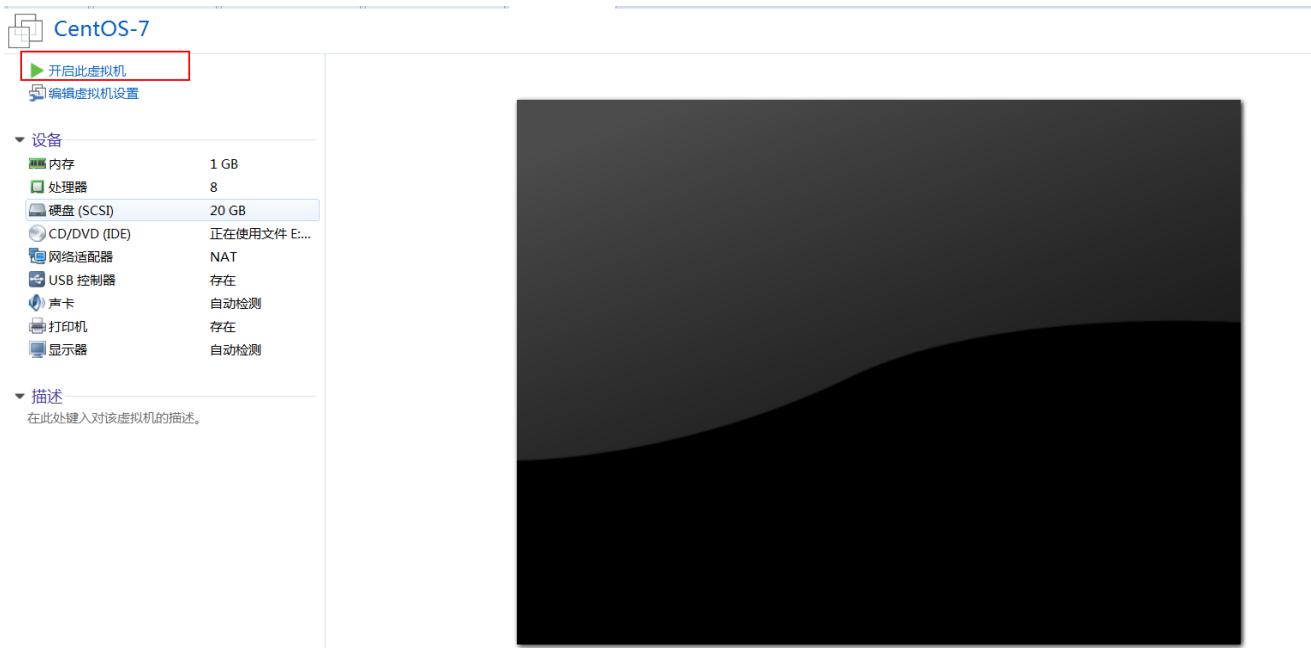
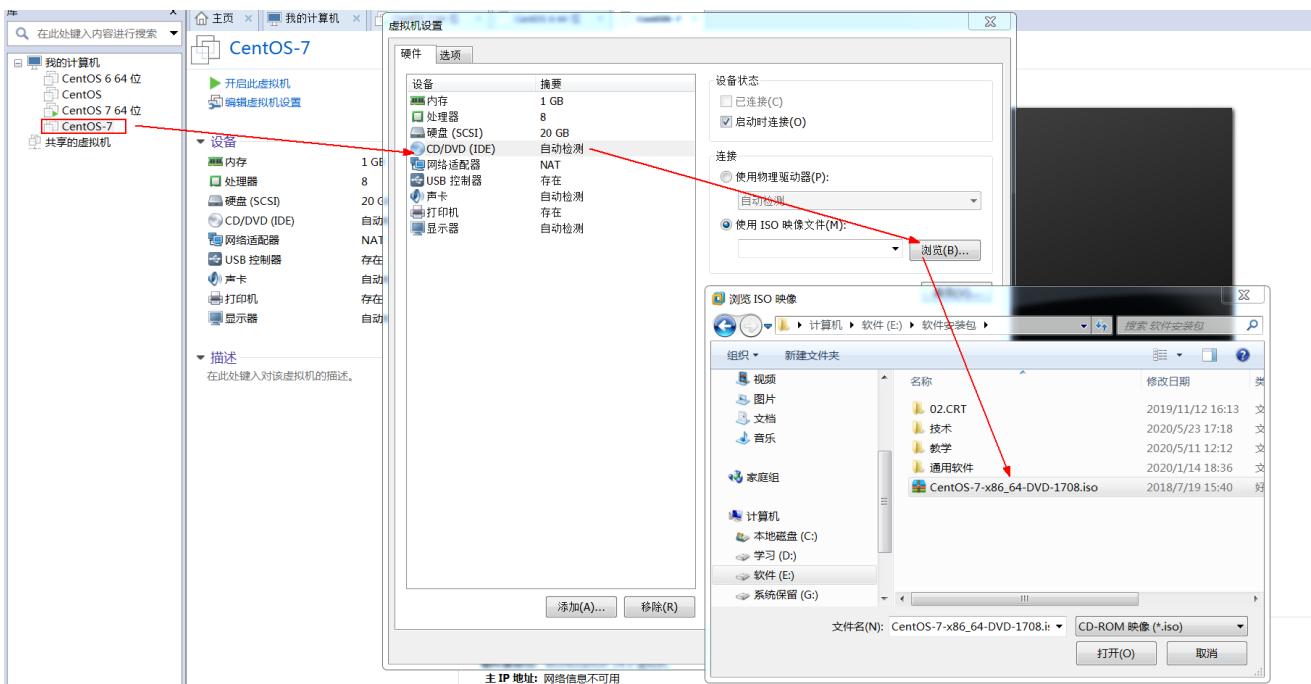


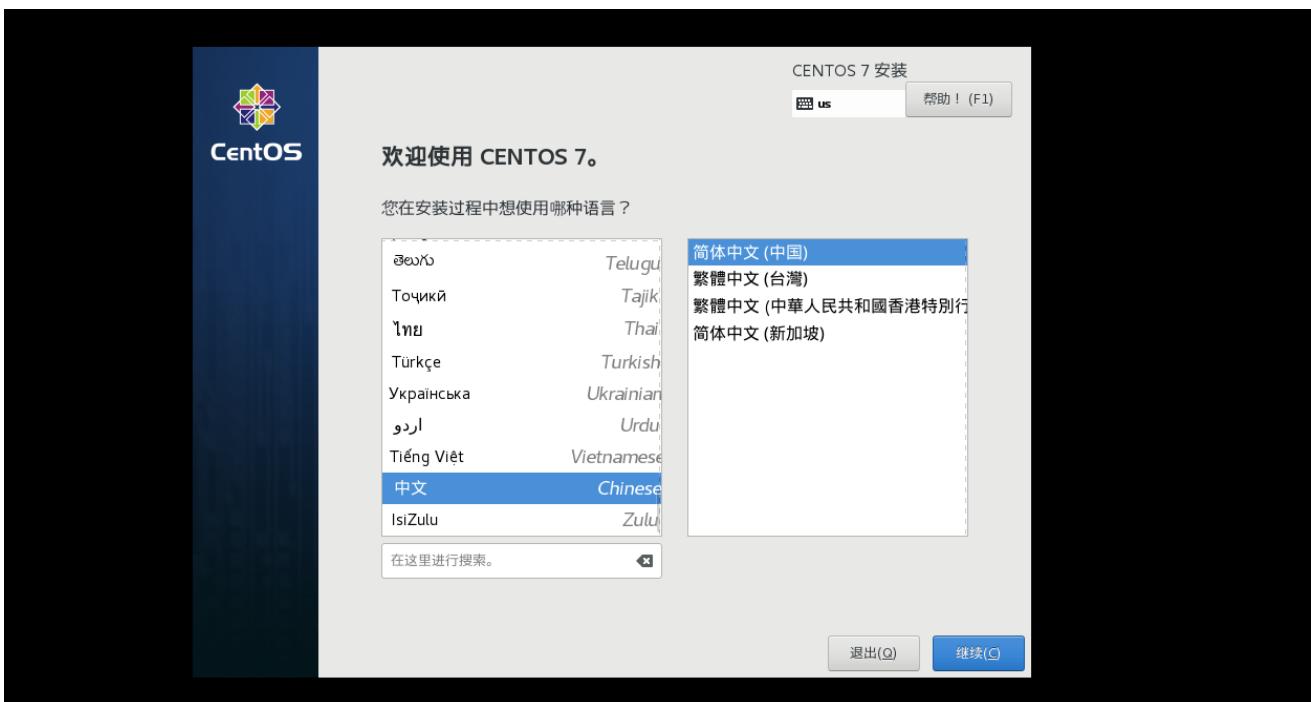
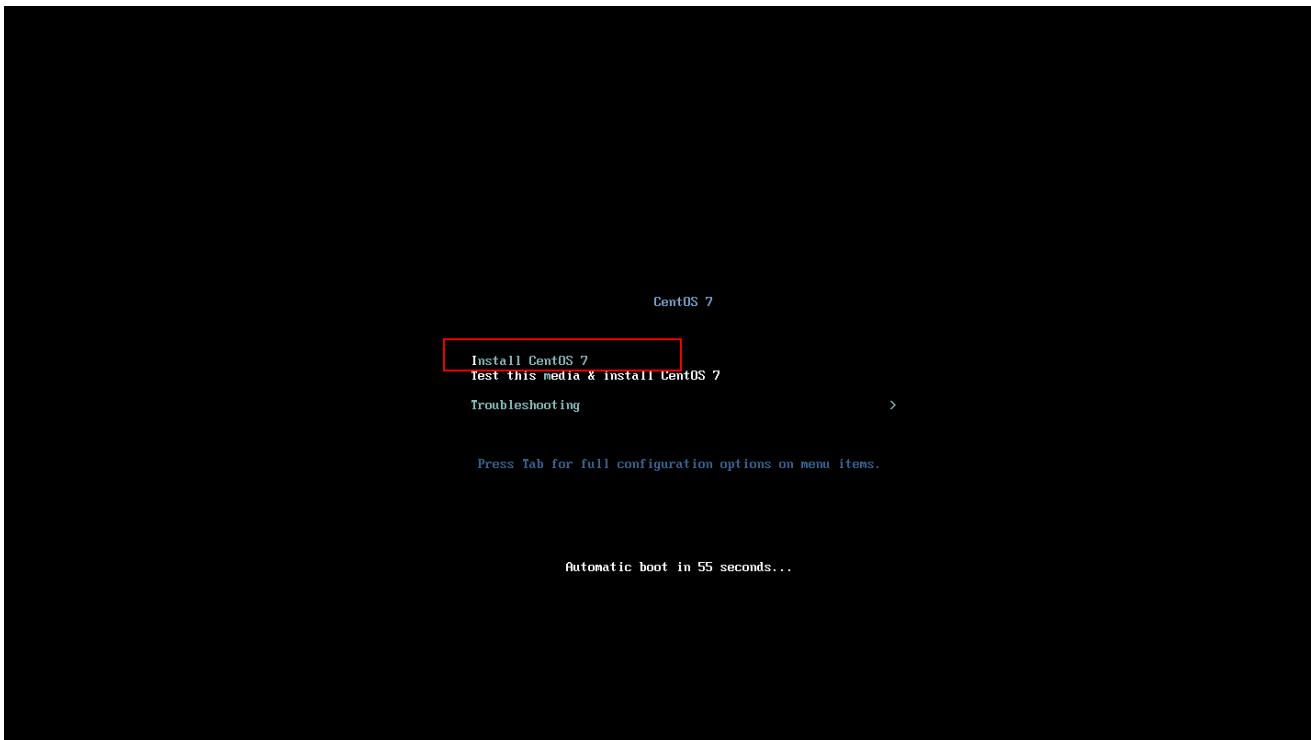


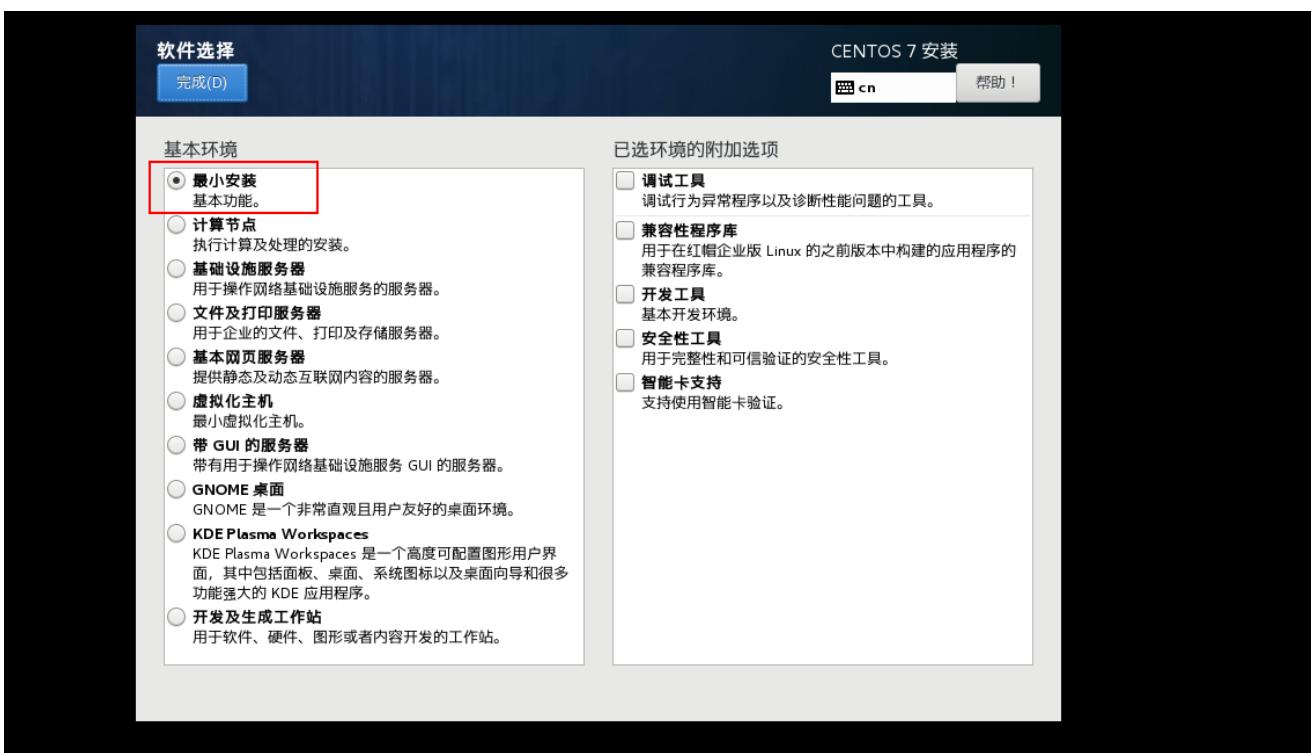




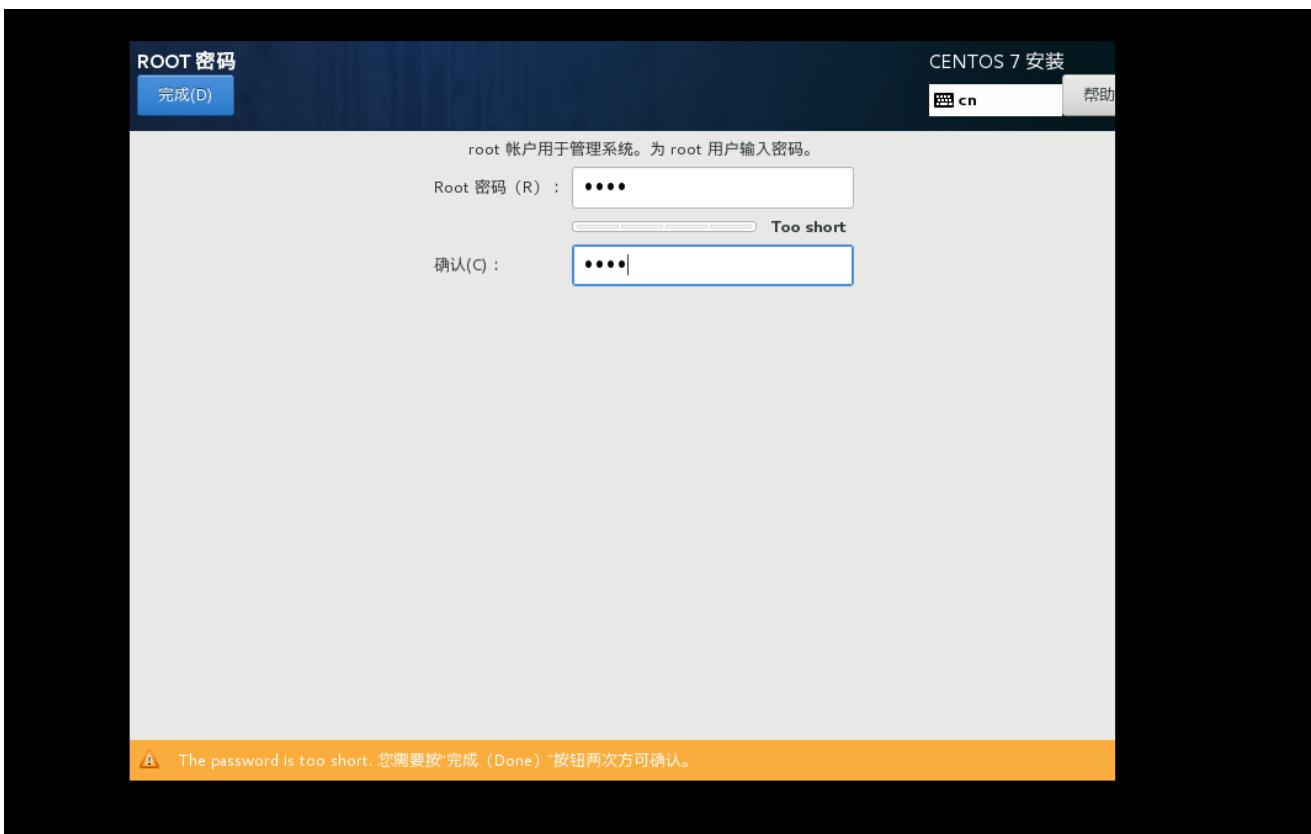


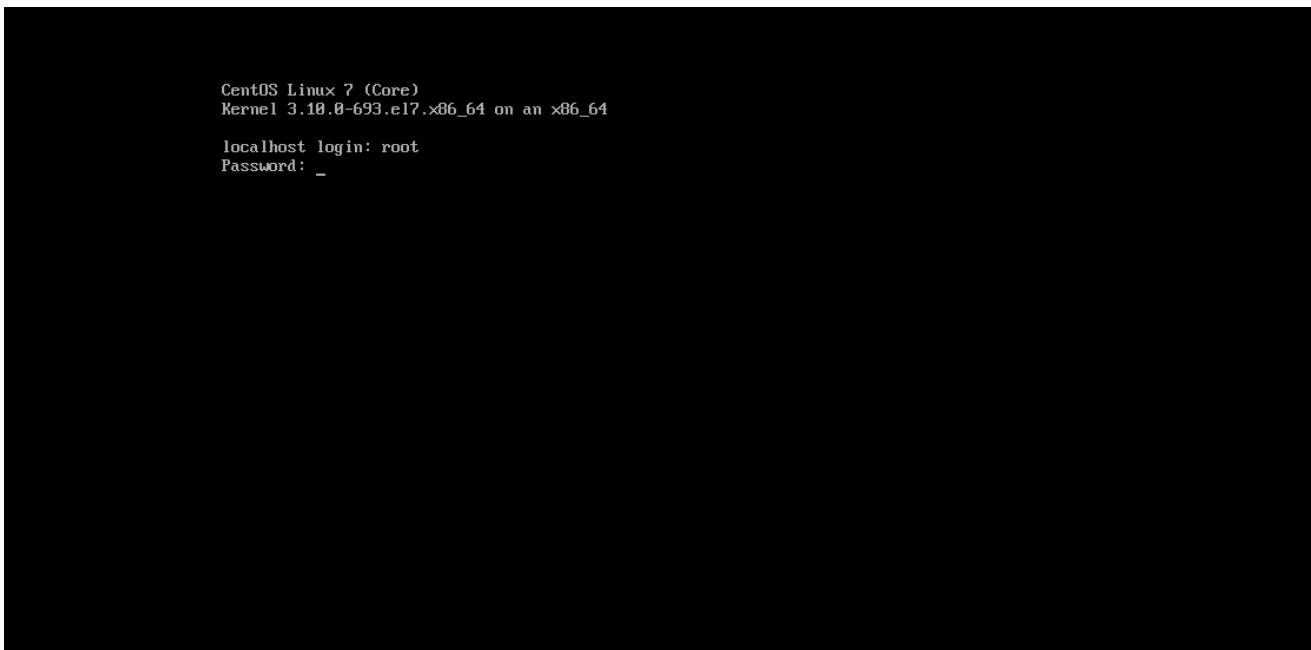
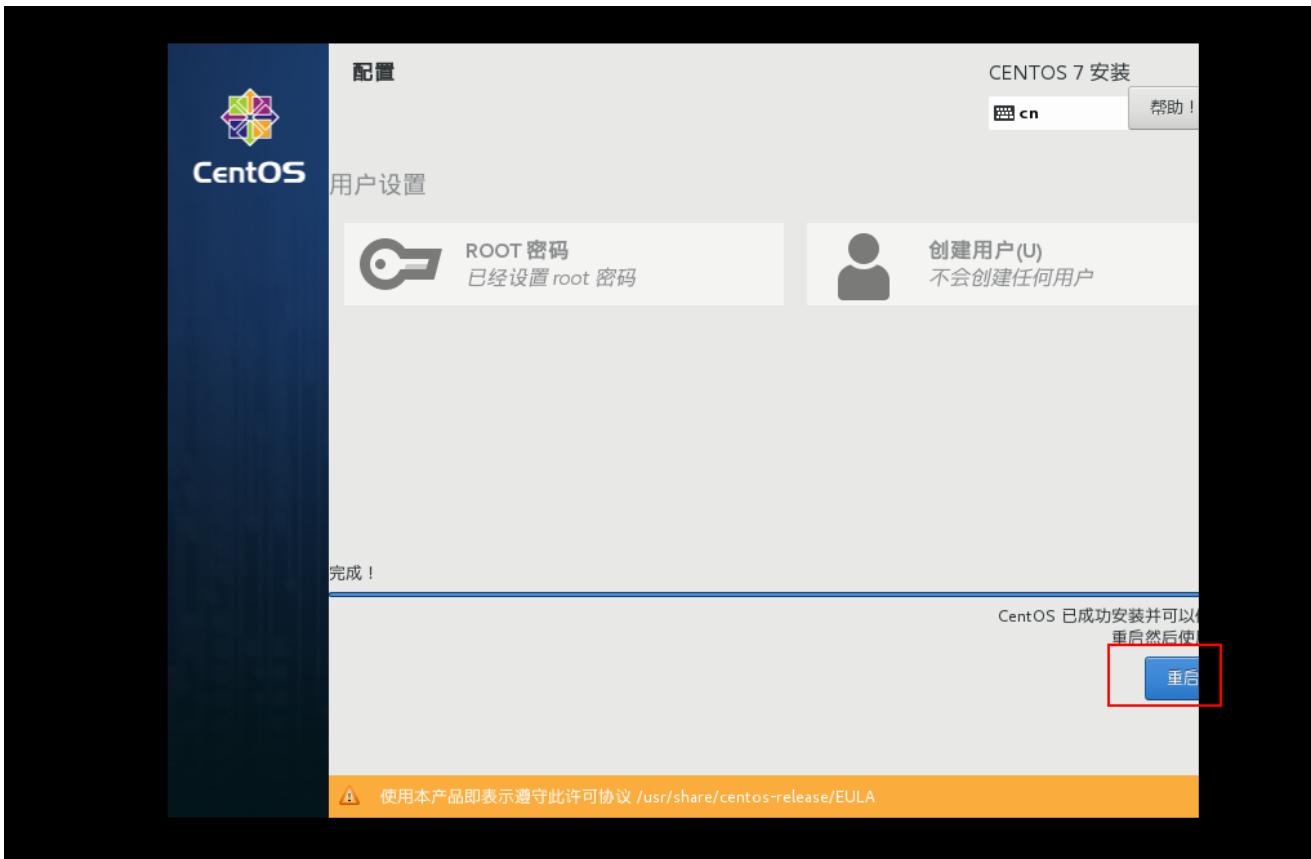










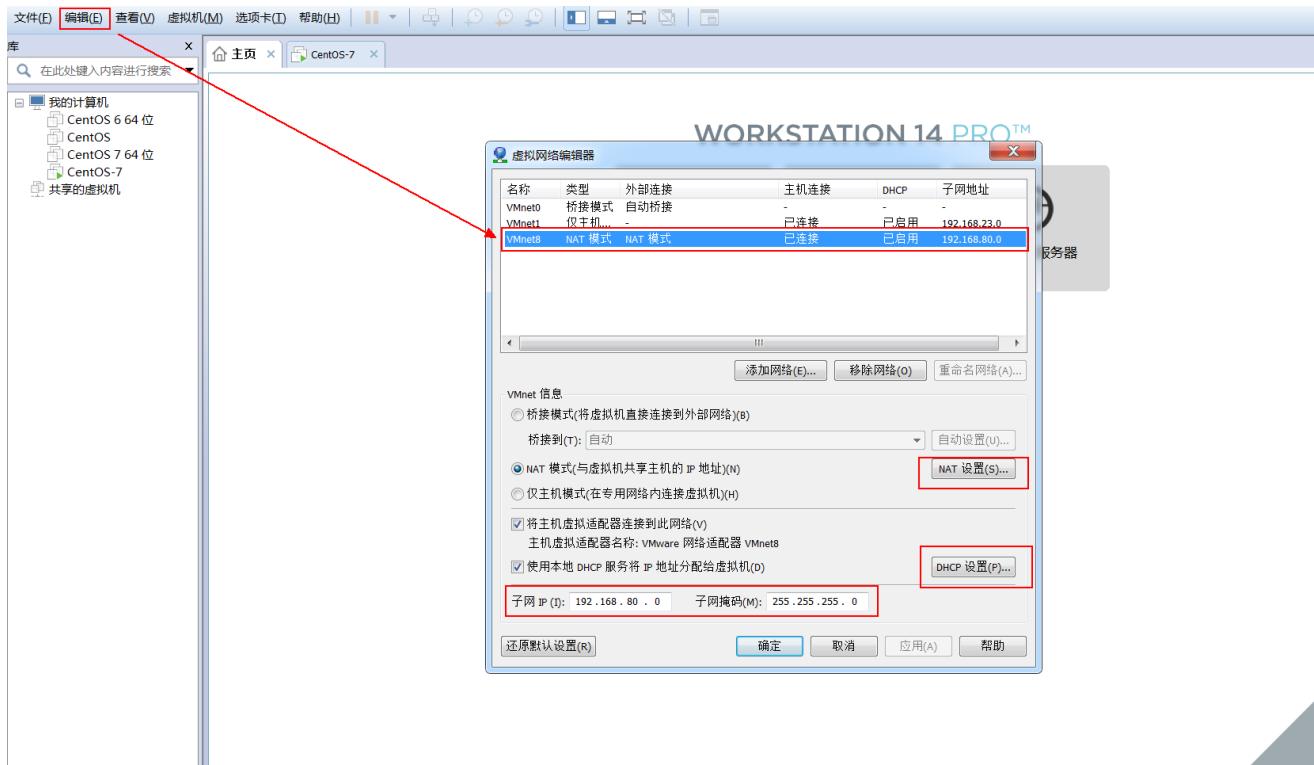


3. 配置静态IP

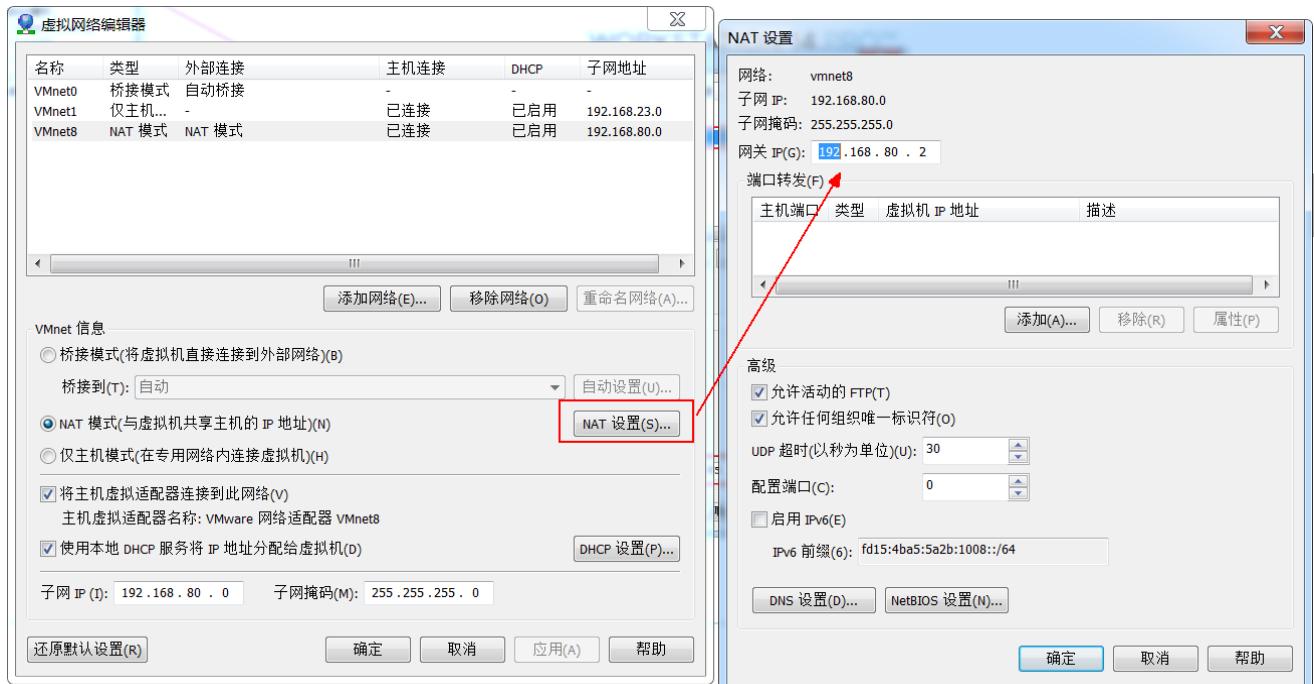
配置网络的目的主要是为了固定虚拟机的内网IP，方便我们在真实的操作系统中使用Linux连接工具软件进行远程连接。

1.关闭虚拟机

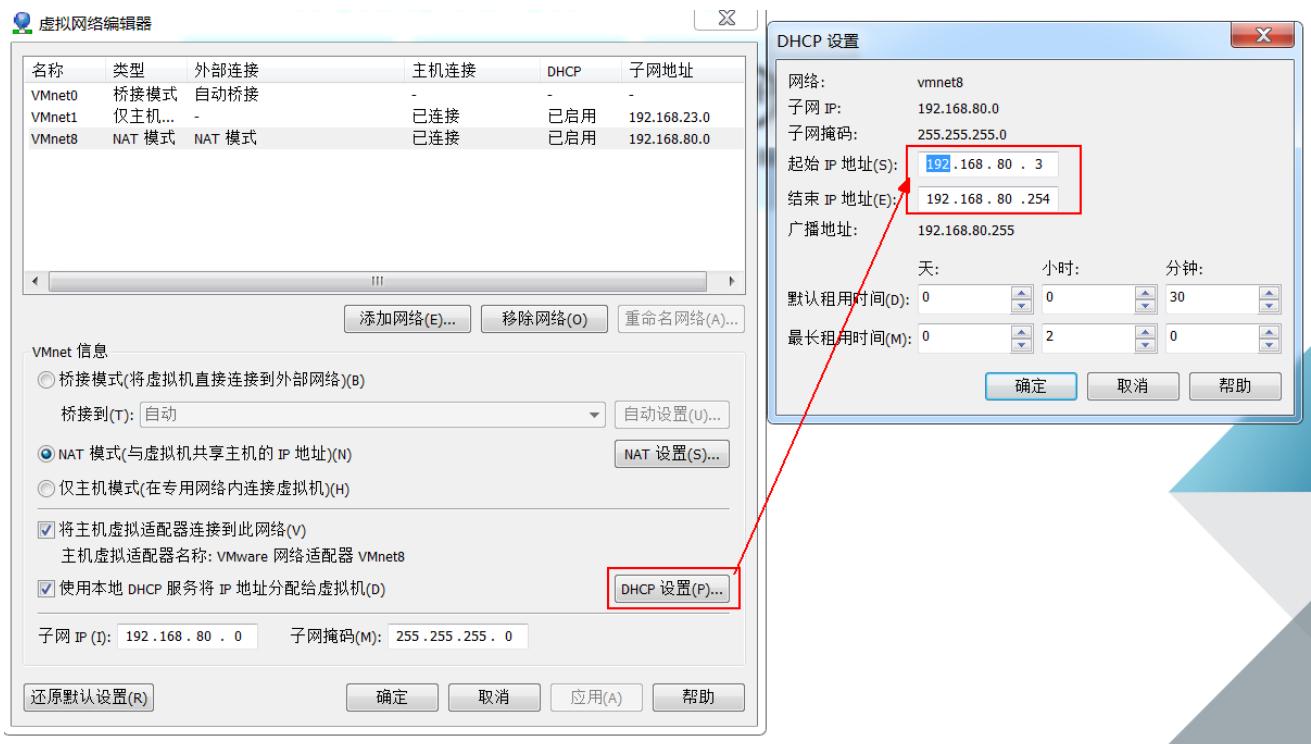
2.点击编辑，选择倒数第二个选项“虚拟网络编辑器”，出现下面的“虚拟网络编辑器”窗口。选择NAT模式，注意子网IP前三位与NAT设置的网关IP、DHCP网段一致。



设置NAT



设置DHCP范围



3.进入操作系统配置网卡信息

执行命令

```
vi /etc/sysconfig/network-scripts/ifcfg-ens33
```

```

TYPE=Ethernet
PROXY_METHOD=none
BROWSER_ONLY=no
BOOTPROTO=static
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=4fb14a41-d80a-47fb-b679096d019f
DEVICE=ens33
ONBOOT=yes
IPADDR=192.168.80.121
NETMASK=255.255.255.0
GATEWAY=192.168.80.2
DNS1=8.8.8.8
~
```

保存退出

```
esc退出
:wq 保存退出
```

4.重启执行命令重启网卡服务

```
systemctl restart network
```

5.检查IP是否变更

命令： ip addr

```
[root@localhost ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN qlen 1
    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_fast state UP qlen 1000
    link/ether 00:0c:29:ff:56:8f brd ff:ff:ff:ff:ff:ff
    inet 192.168.80.121/24 brd 192.168.80.255 scope global ens33
        valid_lft forever preferred_lft forever
    inet6 fe80::2be:cf53:a75f:3119/64 scope link
        valid_lft forever preferred_lft forever
[root@localhost ~]# _
```

5.测试是否网络是否连通

```
ping www.baidu.com
```

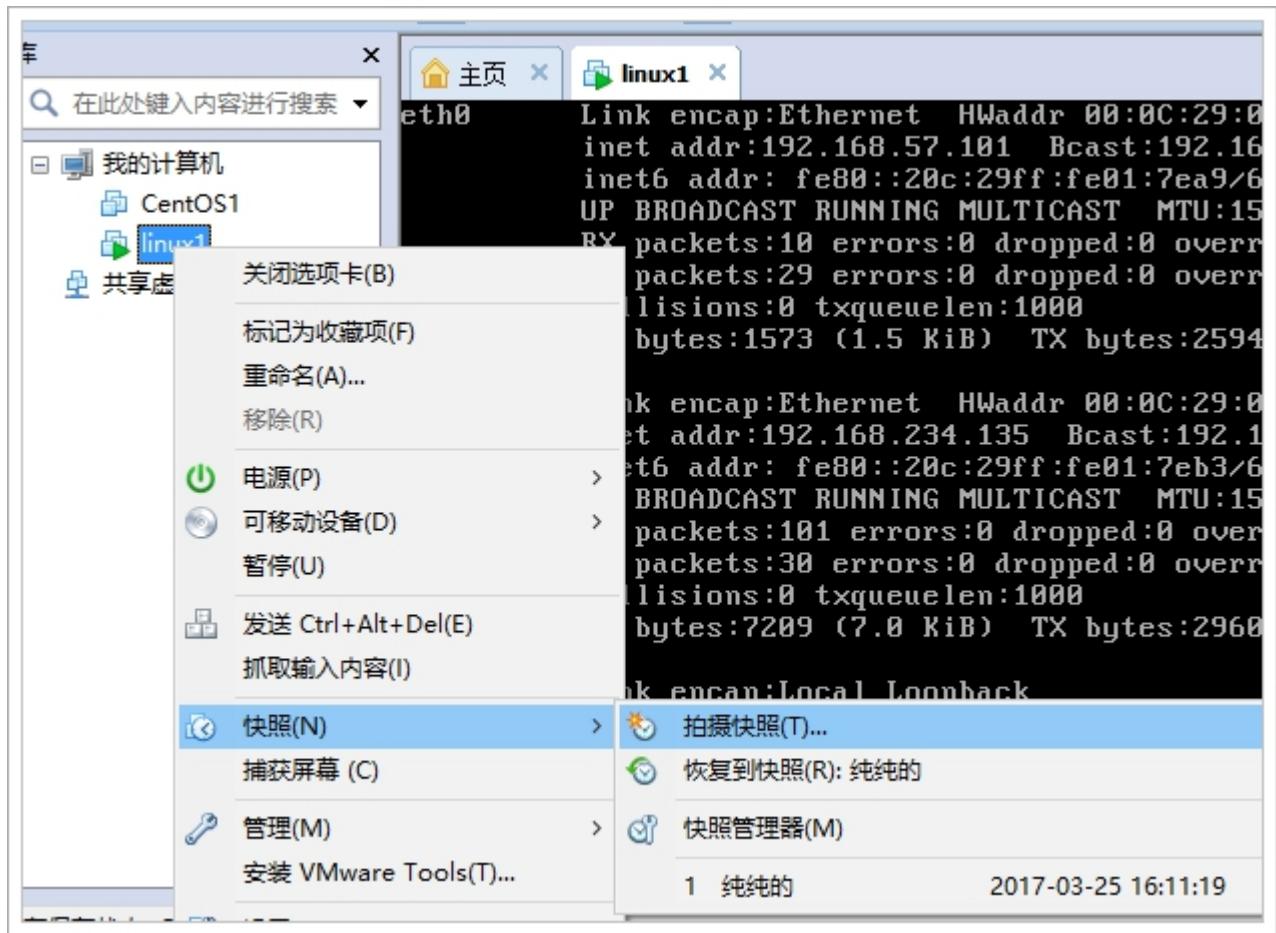
```
[root@localhost ~]# ping www.baidu.com
PING www.a.shifen.com (220.181.38.150) 56(84) bytes of data.
64 bytes from 220.181.38.150 (220.181.38.150): icmp_seq=1 ttl=128 time=6.46 ms
64 bytes from 220.181.38.150 (220.181.38.150): icmp_seq=2 ttl=128 time=5.61 ms
64 bytes from 220.181.38.150 (220.181.38.150): icmp_seq=3 ttl=128 time=5.35 ms
64 bytes from 220.181.38.150 (220.181.38.150): icmp_seq=4 ttl=128 time=6.42 ms
64 bytes from 220.181.38.150 (220.181.38.150): icmp_seq=5 ttl=128 time=8.53 ms
...
--- www.a.shifen.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 13025ms
rtt min/avg/max/mdev = 5.353/6.478/8.538/1.122 ms
[root@localhost ~]#
```

说明网络已经连通.

4. 给虚拟机进行快照

快照其实就是还原点，我们设置了快照。以后如果被我们玩坏了，也可以快速的还原到以前状态！

VMware快照



5. 安装远程连接工具

1 远程终端命令工具

主要功能是向Linux系统远程发送命令

Xshell：目前最好用

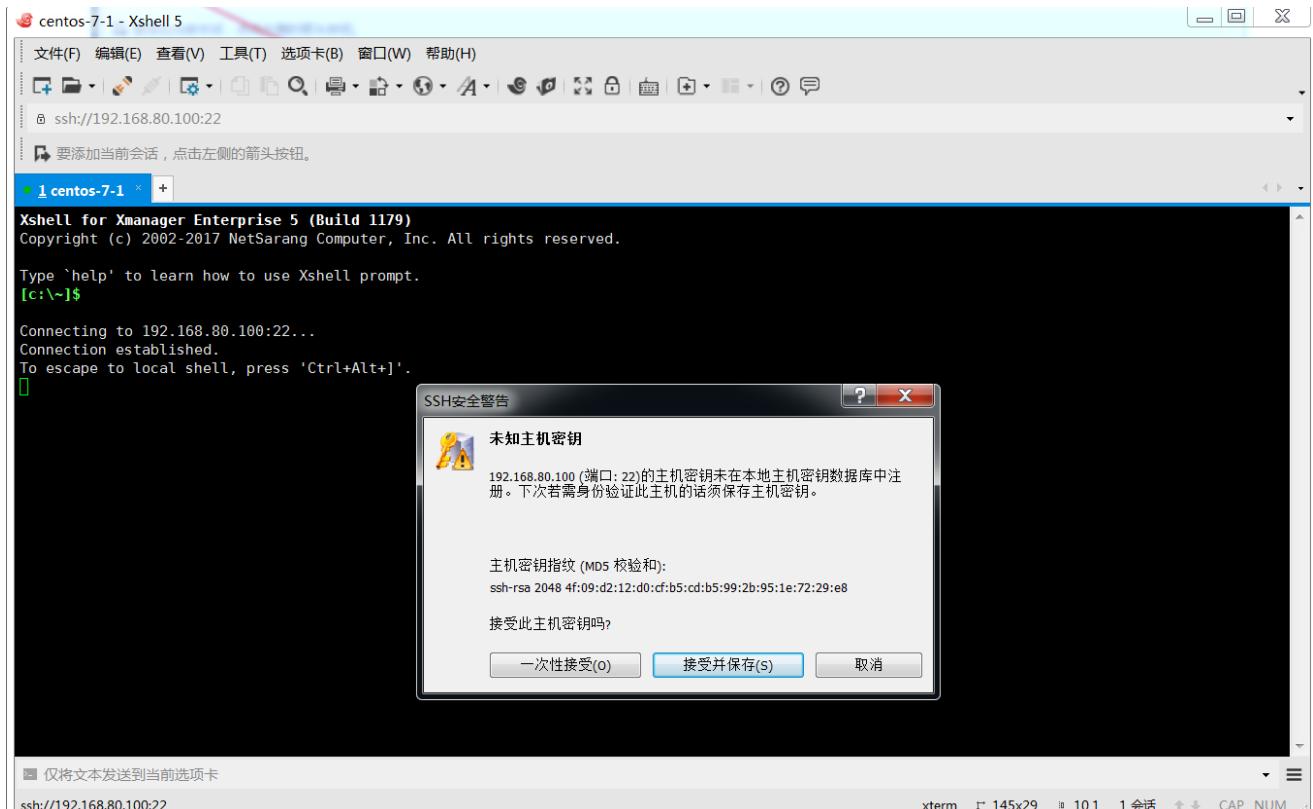
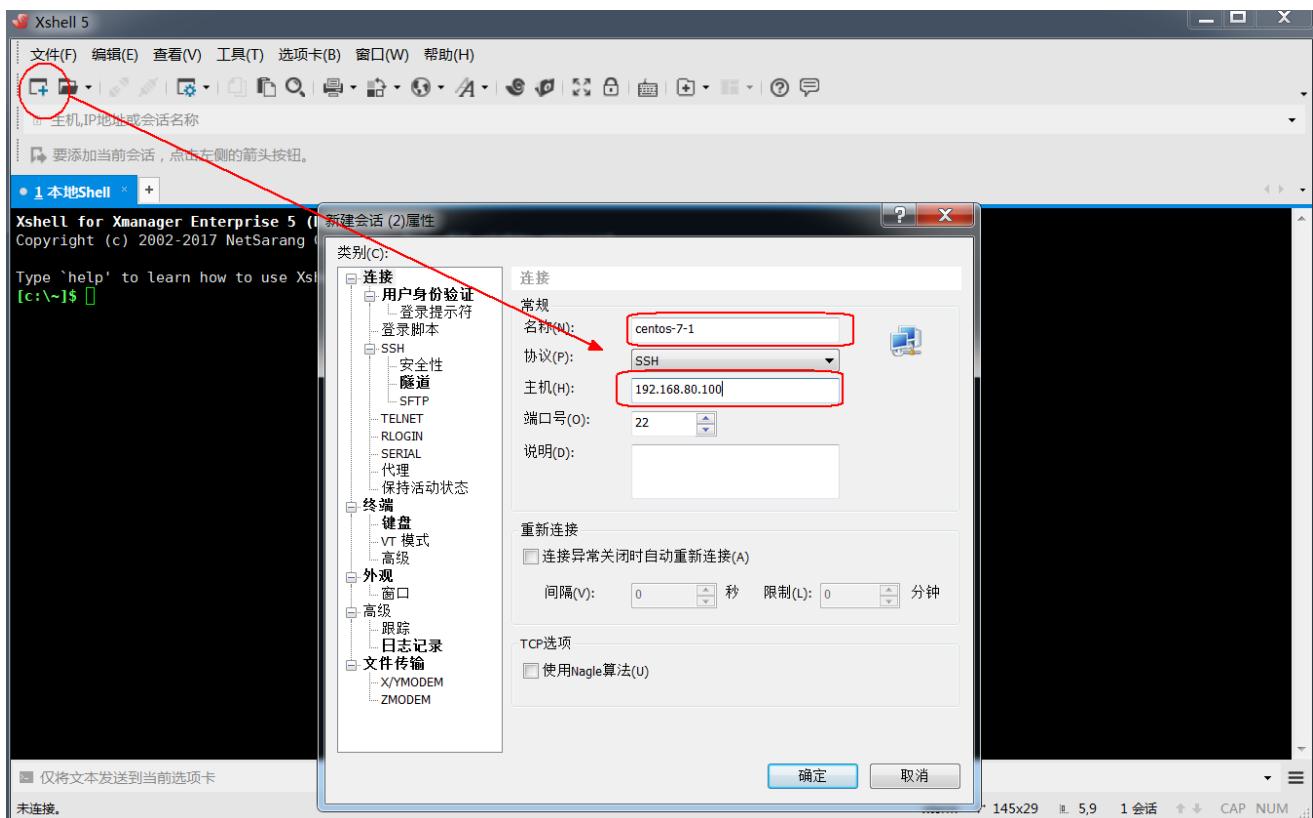
SecureCRT：收费

Putty：早就停止维护了，很多东西支持的很差。但因为习惯依旧很多人支持

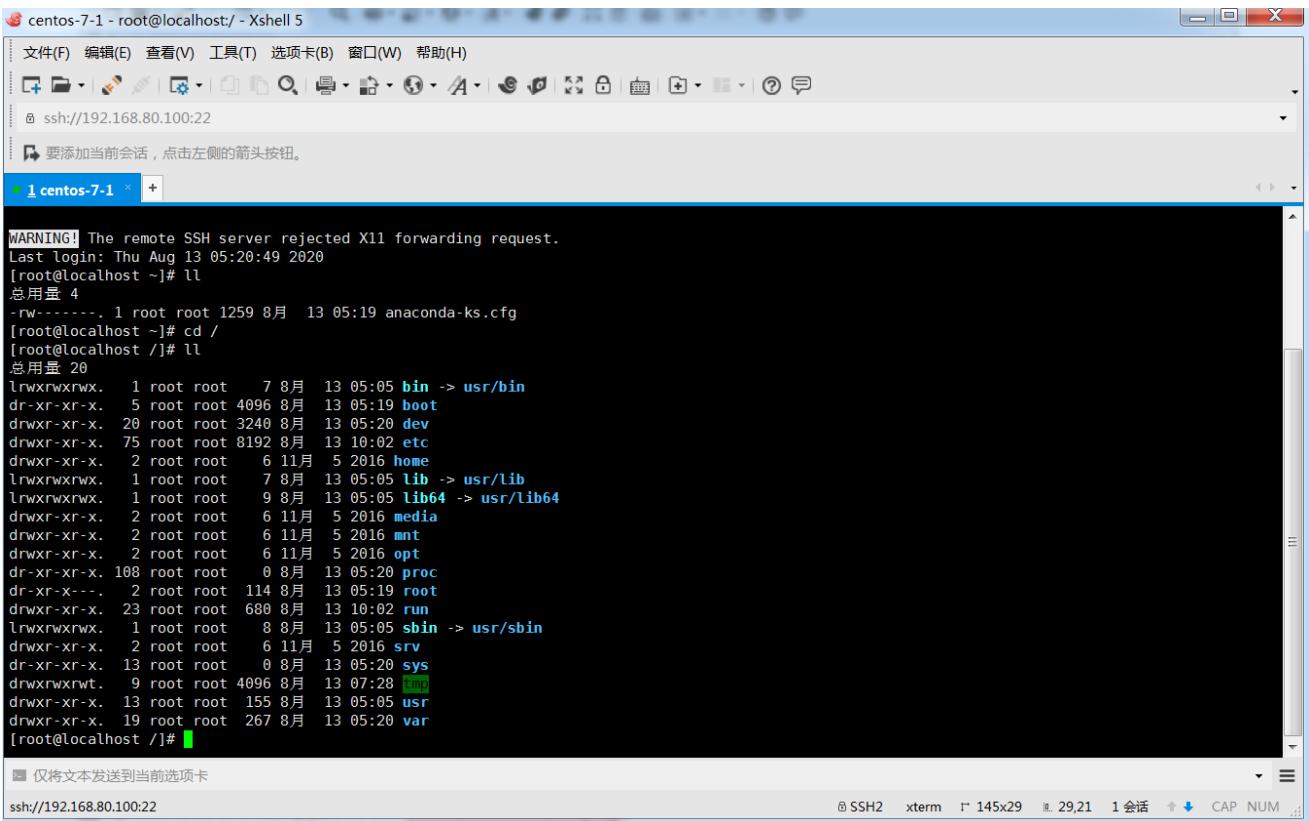
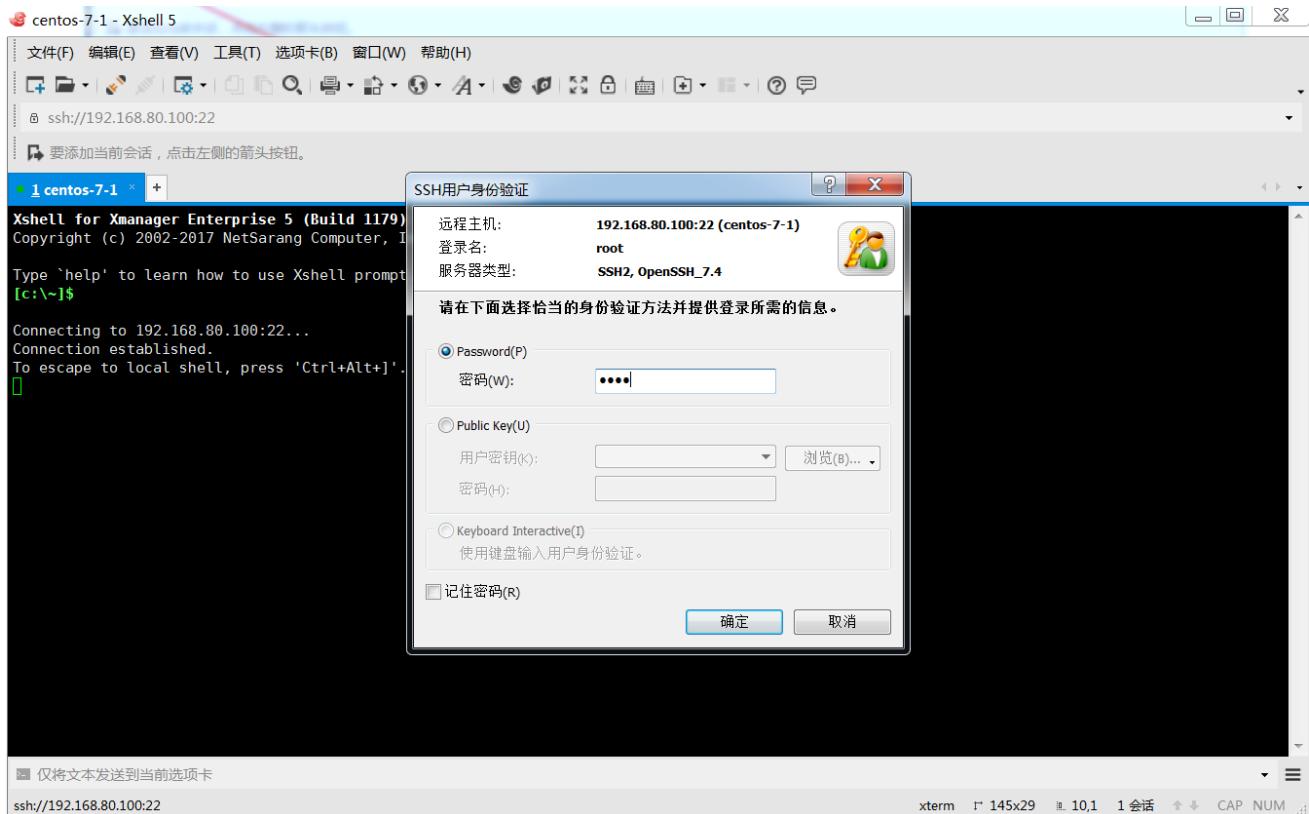
2.XManager工具

1)安装过程比较简单下一步

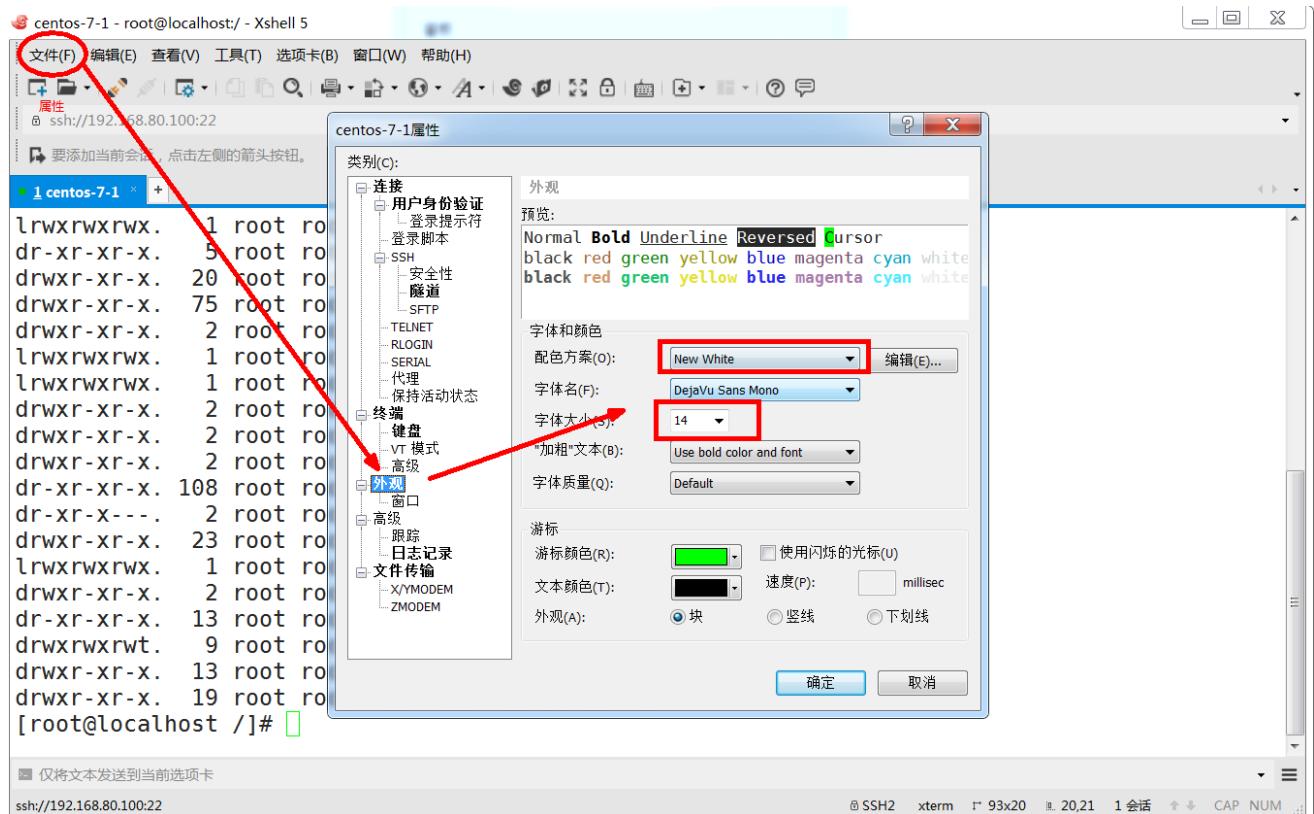
2)配置主机地址



3) 输入用户名和密码

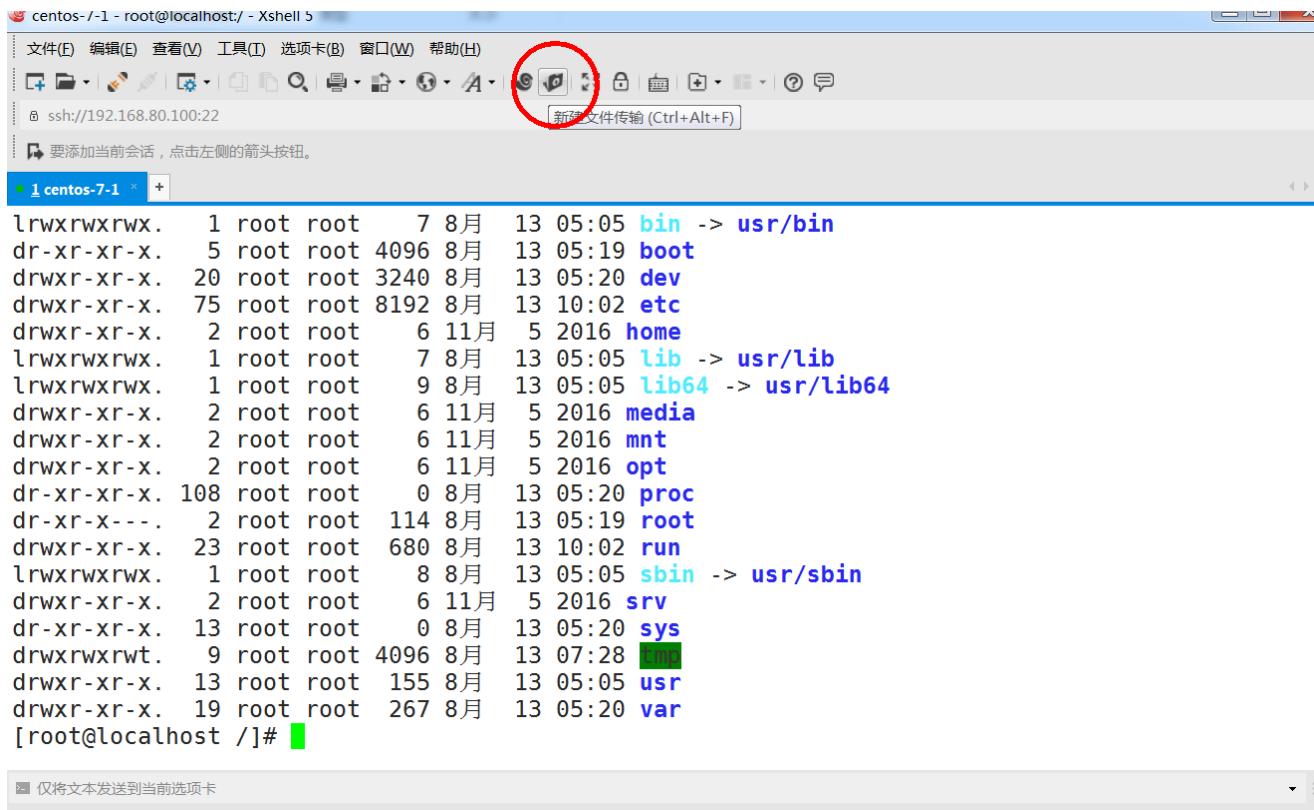


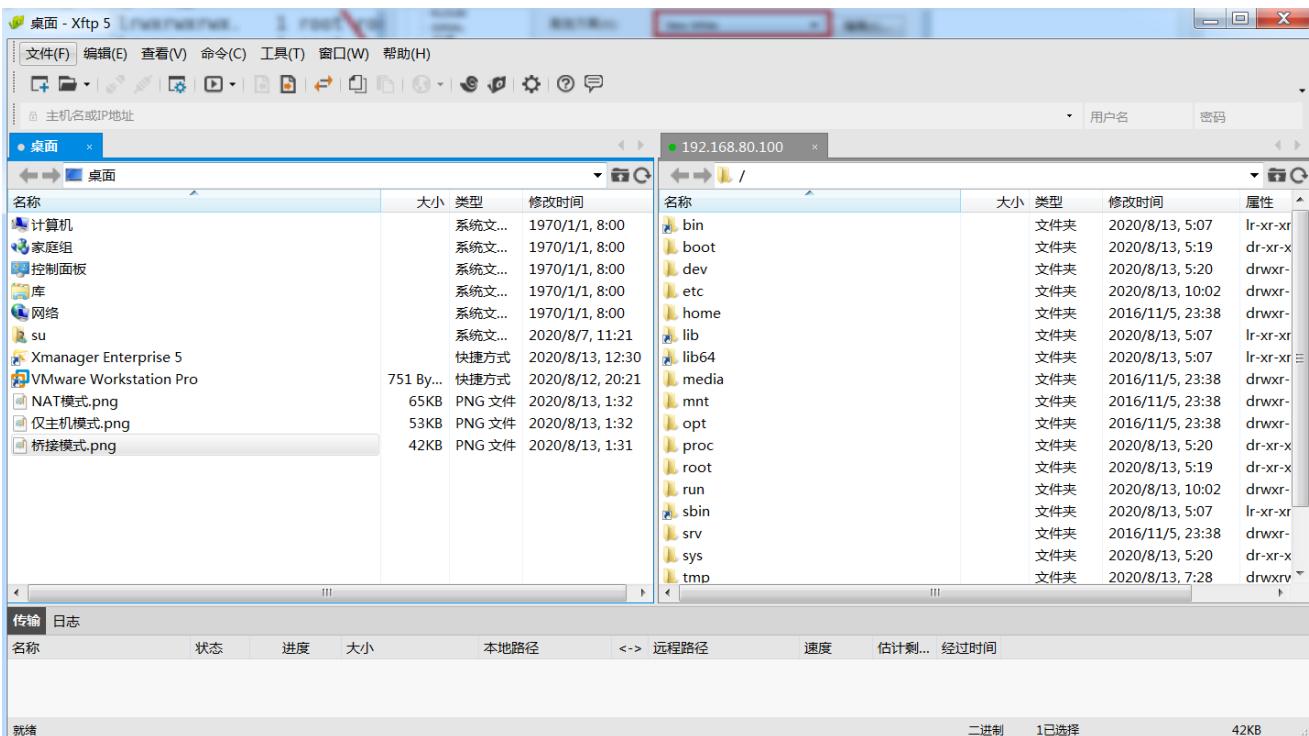
4) xshell工作样式配置



5) xftp 工具使用

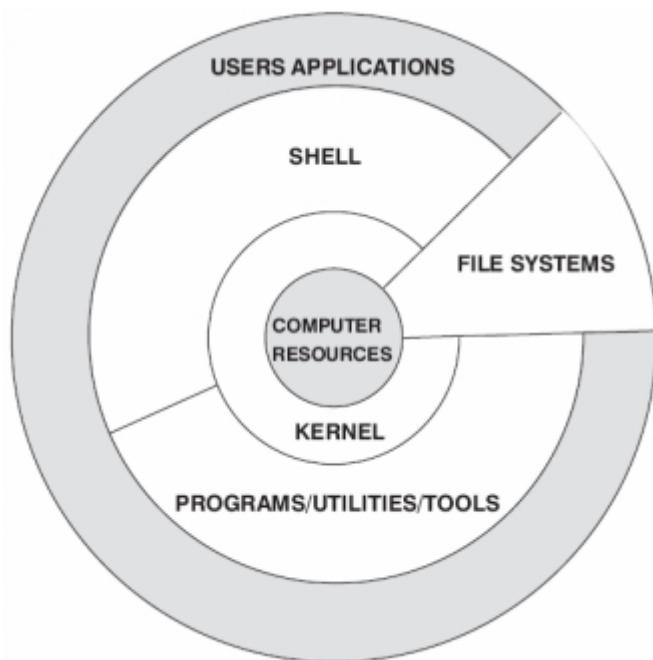
xftp window操作系统和centos 系统传输文件





第三章 Linux的结构

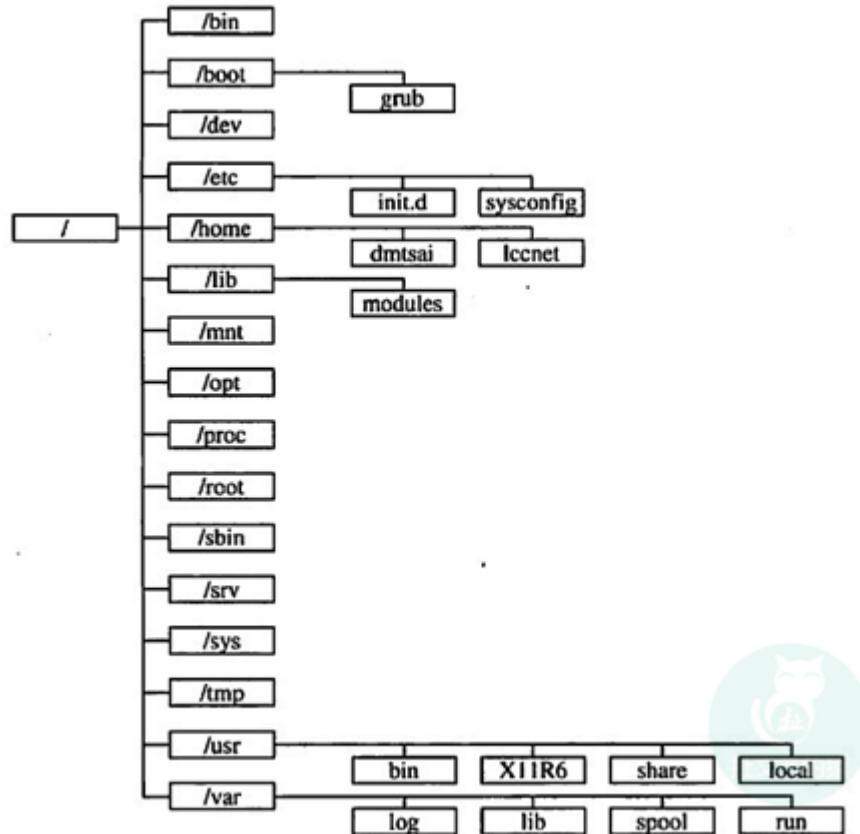
1 Linux组成



内核：是系统的心脏，是运行程序和管理像磁盘和打印机等硬件设备的核心程序。Shell：是系统的用户界面，提供了用户和内核进行交互操作的一种接口。它接收用户输入的命令并把它送入内核去执行，是一个命令解释器。但它不仅是命令解释器，而且还是高级编程语言，shell编程。FILE SYSTEMS(文件系统)：文件系统是文件存放在磁盘等存储设备上的组织方法，Linux支持多种文件系统，如ext3,ext2,NFS,SMB,iso9660等

应用程序：标准的Linux操作系统都会有一套应用程序例如X-Window,Open Office等

2 Linux目录结构



注意：

/boot : 系统引导文件、内核
/bin : 用户的基本命令
/dev : 设备文件
/etc : 配置文件
/home : 用户目录
/root : root用户目录
/sbin : 管理类的基本命令
/tmp : 临时文件存放地
/usr : 共享的只读数据
/mnt : 临时文件系统挂载点
/media : 移动设备挂载点
/opt : 第三方应用程序的安装位置
/srv : 系统运行的服务用到的数据
/var : 变化的数据文件
/proc : 用于输出内核与进程信息相关的虚拟文件系统
/sys : 用于输出当前系统上硬件设备相关信息的虚拟文件系统

第四章 常用Linux命令的基本使用

1 常用快捷键

- 1) tab键 : 命令或者路径提示及补全 ;
- 2) ctrl+c : 放弃当前输入 , 终止当前任务或程序 ;
- 3) ctrl+l : 清屏 ;
- 4) ctrl + insert : 复制 ;
- 5) 鼠标右键 : 粘贴 ;
- 6) alt+c : 断开连接 / ctrl + shfit + R 重新连接
- 7) alt+1/2/3/4/5... : 切换会话窗口
- 8) 上下键 : 查找执行行过的命令 , 或者是history命令

2 终端命令格式

```
command [-options] [parameter]
```

说明:

- **command** : 命令名, 相应功能的英文单词或单词的缩写
- **[-options]** : 选项, 可用来对命令进行控制, 也可以省略
- **[parameter]** : 传给命令的参数, 可以是 零个、一个或者多个

3 帮助命令

因为一个命令有很多可选项, 死记硬背肯定不行, 所以需要借助手册查阅.

- 1) -- help 帮助信息

```
command --help
```

说明:

- 显示 `command` 命令的帮助信息

缺点: 虽然可以查询命令的帮助信息, 但是没有提供 翻页、搜索功能.

2) man 手册

```
man command
```

说明:

- 查询 `command` 命令的使用手册

`man` 时 `manual` 的缩写, 是Linux提供的一个手册, 包含了绝大部分的命令、函数的详细使用说明.

使用 `man` 时的操作键:

操作键	功能
空格键	显示手册的下一屏
Enter键	一次滚动首页也得一行
b	回滚一屏
f	前滚一屏
q (quit)	退出
/word	搜索word字符串
n(next)	搜索下一个
N	搜索上一个

提醒:

- 现阶段只需要 **知道** 通过以下两种方式可以查询命令的帮助信息
- 先学习 **常用命令** 及 **常用参数** 的使用即可, 工作中如果遇到问题可以借助 **网络搜索**

4.其他常用命令

序号	命令	对应英文	作用
01	ls	list	查看当前目录下的内容
02	pwd	print working directory	查看当前所在文件夹
03	cd [目录名]	change directory	切换文件夹
04	touch [文件名]	touch	如果文件不存在, 新建文件
05	mkdir [目录名]	make directory	创建目录
06	rm [文件名]	remove	删除指定的文件名
07	clear	clear	清屏

5. 目录操作命令

Linux中并没有文件夹的概念，应该叫目录。

1) 切换目录

```
cd(change directory) 功能: 切换目录
pwd(print work directory) 功能: 显示当前工作目录
```

目录操作命令

- pwd: 查看当前所在路径
- cd: 切换目录
 - cd .. : 切换到上级目录
 - cd -: 后退到上一次所在目录
 - cd /: 去根目录
 - 绝对路径: /开始的目录, 从根目录开始
 - 相对路径: 直接目录, 从当前目录开始

2) 练习

- 1 查看当前所在目录
- 2 切换到 /usr/local(绝对路径)
- 3 切换到 上一级 /usr
- 4 切换到 /usr/tmp (相对路径)
- 5 切换回 /usr/local
- 6 后退到上一次所在目录

答案

```
pwd  
cd /usr/local  
cd ..  
cd tmp  
cd /usr/local  
cd -
```

6. 查看目录内容

ls(list)	功能：列出目录内容
a(all)	功能：所有
h(human)	功能：人性化的显示(单位：K,G等)

● ls：查看目录下内容

- ls -a：查看全部内容，包含隐藏文件
- ls -l：查看内容的详细信息，效果等同于 ll命令
- ls -lh：以人能读懂的方式显示文件大小

2)ls练习

- 1 查看 /usr内容
- 2 查看所有 /usr内容(既包含隐藏,也包含非隐藏)
- 3 查看 /usr详细内容
- 4 简化 查看 /usr详细内容
- 5 易懂简化版 查看 /usr详细内容

答案

```
[root@localhost /]# ls  
bin boot dev etc home lib lib64 media mnt opt proc root run sbin srv sys tmp
```

```
usr var
[root@localhost /]# ls -a
. .. bin boot dev etc home lib lib64 media mnt opt proc root run sbin srv sys
tmp usr var
[root@localhost /]# ls -l
总用量 20
lrwxrwxrwx.  1 root root    7 8月 13 05:05 bin -> usr/bin
dr-xr-xr-x.  5 root root 4096 8月 13 05:19 boot
drwxr-xr-x. 20 root root 3240 8月 14 01:42 dev
drwxr-xr-x. 75 root root 8192 8月 14 05:55 etc
drwxr-xr-x.  2 root root    6 8月 15 02:56 home
lrwxrwxrwx.  1 root root    7 8月 13 05:05 lib -> usr/lib
lrwxrwxrwx.  1 root root    9 8月 13 05:05 lib64 -> usr/lib64
drwxr-xr-x.  2 root root    6 11月 5 2016 media
drwxr-xr-x.  2 root root    6 11月 5 2016 mnt
drwxr-xr-x.  2 root root    6 11月 5 2016 opt
dr-xr-xr-x. 112 root root    0 8月 14 01:42 proc
dr-xr-x---.  2 root root 151 8月 16 22:37 root
drwxr-xr-x. 23 root root 680 8月 14 05:55 run
lrwxrwxrwx.  1 root root    8 8月 13 05:05 sbin -> usr/sbin
drwxr-xr-x.  2 root root    6 11月 5 2016 srv
dr-xr-xr-x. 13 root root    0 8月 14 01:42 sys
drwxrwxrwt. 11 root root 4096 8月 16 03:15 tmp
drwxr-xr-x. 13 root root 155 8月 13 05:05 usr
drwxr-xr-x. 19 root root 267 8月 13 05:20 var
[root@localhost /]# ll
总用量 20
lrwxrwxrwx.  1 root root    7 8月 13 05:05 bin -> usr/bin
dr-xr-xr-x.  5 root root 4096 8月 13 05:19 boot
drwxr-xr-x. 20 root root 3240 8月 14 01:42 dev
drwxr-xr-x. 75 root root 8192 8月 14 05:55 etc
drwxr-xr-x.  2 root root    6 8月 15 02:56 home
lrwxrwxrwx.  1 root root    7 8月 13 05:05 lib -> usr/lib
lrwxrwxrwx.  1 root root    9 8月 13 05:05 lib64 -> usr/lib64
drwxr-xr-x.  2 root root    6 11月 5 2016 media
drwxr-xr-x.  2 root root    6 11月 5 2016 mnt
drwxr-xr-x.  2 root root    6 11月 5 2016 opt
dr-xr-xr-x. 112 root root    0 8月 14 01:42 proc
dr-xr-x---.  2 root root 151 8月 16 22:37 root
drwxr-xr-x. 23 root root 680 8月 14 05:55 run
lrwxrwxrwx.  1 root root    8 8月 13 05:05 sbin -> usr/sbin
drwxr-xr-x.  2 root root    6 11月 5 2016 srv
dr-xr-xr-x. 13 root root    0 8月 14 01:42 sys
drwxrwxrwt. 11 root root 4096 8月 16 03:15 tmp
drwxr-xr-x. 13 root root 155 8月 13 05:05 usr
drwxr-xr-x. 19 root root 267 8月 13 05:20 var
[root@localhost /]# ls -lh
总用量 20K
lrwxrwxrwx.  1 root root    7 8月 13 05:05 bin -> usr/bin
dr-xr-xr-x.  5 root root 4.0K 8月 13 05:19 boot
drwxr-xr-x. 20 root root 3.2K 8月 14 01:42 dev
drwxr-xr-x. 75 root root 8.0K 8月 14 05:55 etc

drwxr-xr-x.  2 root root    6 8月 15 02:56 home
```

```
lrwxrwxrwx.  1 root root    7 8月 13 05:05 lib -> usr/lib
lrwxrwxrwx.  1 root root    9 8月 13 05:05 lib64 -> usr/lib64
drwxr-xr-x.  2 root root    6 11月  5 2016 media
drwxr-xr-x.  2 root root    6 11月  5 2016 mnt
drwxr-xr-x.  2 root root    6 11月  5 2016 opt
dr-xr-xr-x. 112 root root   0 8月 14 01:42 proc
dr-xr-x---.  2 root root   151 8月 16 22:37 root
drwxr-xr-x.  23 root root   680 8月 14 05:55 run
lrwxrwxrwx.  1 root root   8 8月 13 05:05 sbin -> usr/sbin
drwxr-xr-x.  2 root root   6 11月  5 2016 srv
dr-xr-xr-x. 13 root root   0 8月 14 01:42 sys
drwxrwxrwt. 11 root root  4.0K 8月 16 03:15 tmp
drwxr-xr-x. 13 root root  155 8月 13 05:05 usr
drwxr-xr-x. 19 root root  267 8月 13 05:20 var
```

扩展： linux 如何树结构显示某文件夹下的所有文件（包括子文件夹）？

```
ls -R
```

7. 创建文件

通过 `touch` 命令 创建文件

第一步: 创建一个空白文件

```
touch 不存在的文件
```

第二步: 修改文件的末次访问时间

```
touch 存在的文件
```

2) 案例例

```
[root@linux121 ~]# touch test.txt
```

8. 创建目录命令

通过 `mkdir` 命令 创建目录

基本语法

```
mkdir [-p] 要创建的目录
```

选项	英文	含义
-p	parent	可以递归创建目录

案例

```
[root@linux121 ~]# mkdir test  
[root@linux121 ~]# mkdir -p test/test1
```

注意

通过 `mkdir -p 目录名` 命令 创建目录
注意：新建目录的名称 不能与当前目录中 已有的目录或文件同名

9.rmdir 删除一个空的目录

基本语法：`rmdir 要删除的空目录 , remove 即移除`

案例

```
[root@linux121 ~]# mkdir test2  
[root@linux121 ~]# rmdir test2
```

注意:

如果该目录中存在文件或其他目录是该命令是不能删除的.

10.touch创建文件和rm 删除文件

1)创建文件的命令

`touch 文件名称`

注意事项：
`touch 文件名1 文件2 ..` 可以创建多个文件

2) 删除命令

语法

```
rm [-参数] 文件/目录 (功能描述：递归删除目录中所有内容) 慎用
```

通过 `rm -rf 文件/目录` 命令实现删除文件和目录的功能, `rm` 对应的英文是 `remove` 含义: 删除

参数	英文	含义
-f	force (强制)	强制删除,忽略不存在的文件,无需提示
-r	recursive (递归)	递归地删除目录下的内容, 删除文件夹 时必须加此参数

文件操作命令

- `touch` 文件名: 创建文件
- `rm`: 删除文件或目录
 - `rm` 文件名: 删除一个文件
 - `rm -f` 文件名: 不经确认就删除文件
 - `rm -r` 目录: 递归删除一个目录及目录中的内容
 - `rm -rf` 目录: 递归删除一个目录, 并且不经确认
 - `rm -rf *`: 清空当前文件夹
 - `rm -rf /*`: **自杀行为, 不要尝试**

案例

```
1) 删除空目录  
[root@linux121 test]# rmdir test1  
2) 递归删除目录中所有内容  
[root@linux121 test]# rm -rf test2
```

11. cp 复制拷贝命令

通过 `cp` 实现复制将指定的文件或目录复制到两个文件或目录中

基本语法:

```
(1) cp source dest (功能描述：复制source文件到dest)
(2) cp -r sourceFolder targetFolder (功能描述：递归复制整个文件夹)
```

命令	英文	作用
-r	recursive (递归)	递归复制目标目录的内容

案例：

```
(1) 复制文件
[root@linux121 opt]# cp test.txt test1.txt
(2) 递归复制整个文件夹
[root@linux121 opt]# cp -r abc /tmp
```

12. mv (move)

通过 **mv** 命令可以用来 移动 文件 或 目录, 也可以给 文件或目录重命名

基本语法

```
(1) mv oldNameFile newNameFile (功能描述：重命名)
(2) mv /temp/movefile /targetFolder (功能描述：递归移动文件)
```

2) 案例：

```
1) 重命名
[root@linux121 test]# mv file1 file11 (把file1文件夹改名为file11)

2) 移动文件
[root@linux121 test]# mv file11 test (把file11文件夹放到test文件夹内)
```

13.cat 查看文件内容

查看文件内容，从第一行开始显示。 1) 基本语法

```
cat [选项] 要查看的文件
```

2) 选项

```
-b : 列出行号，仅针对非空白行做行号显示，空白行不标行号！  
-E : 将结尾的断行行节 $ 显示出来；  
-n : 列出行号，连同空白行也会有行号，与 -b 的选项不同；  
-T : 将 [tab] 按键以 ^I 显示出来；  
-v : 列出一些看不出来的特殊字符  
-A : 相当于 -vET 的整合选项，可列列出一些特殊字符而不是空白而已；
```

3) 案例

查看内容

```
[root@localhost usr]# cat abc.txt  
babbaba  
abababab  
ababbabaZZ
```

查看添加-A

```
[root@localhost usr]# cat -A abc.txt  
babbaba$  
abababab$  
ababbabaZZ$
```

查看添加-b

```
[root@localhost usr]# cat -b abc.txt  
1 babbaba  
2 abababab  
3 ababbabaZZ
```

查看添加-E

```
[root@localhost usr]# cat -E abc.txt  
babbaba$  
abababab$  
ababbabaZZ$
```

查看内容添加-n

```
[root@localhost usr]# cat -n abc.txt  
1 babbaba  
2 abababab  
3 ababbabaZZ
```

查看内容添加-T

```
[root@localhost usr]# cat -T abc.txt  
babbaba  
abababab  
ababbabaZZ
```

查看内容添加-v

```
[root@localhost usr]# cat -v abc.txt  
babbaba  
abababab  
ababbabaZZ
```

14. more 查看文件内容

查看文件内容，一页一页的显示文件内容。 1) 基本语法：

```
more 要查看的文件
```

2) 功能使用说明

空格键 (space) : 代表向下翻一页；

Enter : 代表向下翻『一行』；

q 代表立刻离开 more , 不再显示该文件内容。

Ctrl+F 向下滚动一屏

Ctrl+B 返回上一屏

= 输出当前行的行号

3) 案例

```
[root@linux121 test1]# more test1.java
```

15. less 查看文件内容

less 的作用与 more 十分相似，都可以用来浏览文字档案的内容，不同的是 less 允许使用[pageup] [pagedown]往回滚动。 1) 基本语法：

```
less 要查看的文件
```

2) 功能使用说明

空格键 : 向下翻动一页；

[pagedown] : 向下翻动一页；

[pageup] : 向上翻动一页；

/字符串 : 向下搜寻『字符串』的功能；n : 向下查找；N : 向上查找；

q : 离开 less 这个程序；

3) 案例

```
[root@linux121 test1]# less test1.java
```

16.head查看文件内容

查看文件内容，只看头几行，优点：对于大文件不必都加载，只显示头几行即可。

1) 基本语法

head 文件名 : 查看前10行

head -n 3 文件名 : 查看文件的前3行

head -c 3 文件名 : 查看文件的前3个字符

17.tail 查看文件内容

查看文件内容，只看尾巴几行行，优点：可以查看文件实时追加的内容。 1) 基本语法

(1) tail -n 10 文件 (功能描述：查看文件头 (从未尾开始数) 10行行内容，10可以是任意行行数)
(2) tail -f 文件 (功能描述：实时追踪该文档的所有更新)

2) 案例

- 1 查看文件后10行内容
- 2 动态追踪文件内容
- 3 动态追踪 最后10行内容 且 退出

答案

```
tail -10 文件名  
tail -f 文件名  
tail -10f 文件名 (ctrl + c 是退出)
```

18 组合命令

Linux中的命令组合后，可以产生神奇的效果!

append 追加

replace 替换,覆盖

1) 重定向输出>和>>

- > 重定向输出；
- >> 重定向输出，又追加功能；
- 示例：
 - cat /etc/passwd > a.txt 将输出定向到a.txt中
 - cat /etc/passwd >> a.txt 输出并且追加

echo 控制台输出的内容 (类似sout)

需求:

- 1 删除 /usr/tmp/目录下的所有内容
- 2 增加 1.txt文件, 内容: 文本1
- 3 增加 2.txt文件, 内容: 文本2
- 4 将2.txt内容 复制粘贴到 3.txt
- 5 将1.txt内容 复制粘贴到 3.txt(缺点: 产生替换效果)
- 6 将2.txt内容 复制追加粘贴到 3.txt中
- 7 将ifconfig内容 追加到 3.txt中

答案:

- 1 删除 /usr/tmp/目录下的所有内容

```
[root@spark01 tmp]# rm -rf *
```

- 2 增加 1.txt文件, 内容: hello

```
[root@spark01 tmp]# touch 1.txt && ls  
1.txt  
[root@spark01 tmp]# echo hello > 1.txt && cat 1.txt  
hello
```

- 3 增加 2.txt文件, 内容: world

```
[root@spark01 tmp]# touch 2.txt && ls  
1.txt 2.txt  
[root@spark01 tmp]# echo world > 2.txt && cat 2.txt  
world
```

- 4 将2.txt内容 复制粘贴到 3.txt

```
[root@spark01 tmp]# touch 3.txt && ls  
1.txt 2.txt 3.txt  
[root@spark01 tmp]# cat 2.txt > 3.txt && cat 3.txt  
world
```

5 将1.txt内容 复制粘贴到 3.txt(缺点: 产生替换效果)

```
[root@spark01 tmp]# cat 1.txt > 3.txt && cat 3.txt  
hello
```

6 将2.txt内容 复制追加粘贴到 3.txt中

```
[root@spark01 tmp]# cat 2.txt >> 3.txt && cat 3.txt  
hello  
world
```

7 将ifconfig指令内容 追加到 3.txt中

```
[root@spark01 tmp]# ifconfig >> 3.txt  
[root@spark01 tmp]# cat 3.txt  
hello  
world  
eth0      Link encap:Ethernet HWaddr 00:0C:29:0E:3F:05  
          inet addr:192.168.220.200 Bcast:192.168.220.255 Mask:255.255.255.0  
          inet6 addr: fe80::20c:29ff:fe0e:3f05/64 Scope:Link  
            UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1  
            RX packets:12903 errors:0 dropped:0 overruns:0 frame:0  
            TX packets:10765 errors:0 dropped:0 overruns:0 carrier:0  
            collisions:0 txqueuelen:1000  
            RX bytes:1189753 (1.1 MiB) TX bytes:3324044 (3.1 MiB)  
            Interrupt:19 Base address:0x2000  
  
lo        Link encap:Local Loopback  
          inet addr:127.0.0.1 Mask:255.0.0.0  
          inet6 addr: ::1/128 Scope:Host  
            UP LOOPBACK RUNNING MTU:16436 Metric:1  
            RX packets:0 errors:0 dropped:0 overruns:0 frame:0  
            TX packets:0 errors:0 dropped:0 overruns:0 carrier:0  
            collisions:0 txqueuelen:0  
            RX bytes:0 (0.0 b) TX bytes:0 (0.0 b)
```

2) 管道 “|”

管道符号 | 的作用是： 将一个命令的输出作为另一个命令的输入 .

配合使用的命令

```
ps(Process Status)      进程状态 ps -ef  
grep(Global Regular Expression Print) 全局正则表达式版本(搜索)
```

- 管道是Linux命令中重要的一个概念，其作用是将一个命令的输出用作另一个命令的输入。
- 示例
 - `ls --help | more` 分页查询帮助信息
 - `ps -ef | grep java` 查询名称中包含java的进程

需求:

```
1 分页查询 ls的帮助信息(回车键 下一行, 空格键 下一页, ctrl +c 退出)  
2 查询ls帮助信息中包含'递归'的指令
```

答案:

```
ls --help | more  
ls --help | grep '递归'
```

3) 逻辑控制&&

命令间的逻辑控制

- 命令之间使用 **&&** 连接，实现类似逻辑与的功能。
- 只有在 **&&** 左边的命令运行成功时，**&&** 右边的命令才会被执行。
- 只要左边命令运行失败，后面的命令就不会被执行。
- 例如：
 - `cp 1.txt 2.txt && cat 2.txt`

因为 启动软件通常不会打印启动的日志信息, 所以需要再打开对应日志信息查看.

分:

```
tail -100f catalina.out  
./startup.sh
```

使用合并指令:

```
./startup.sh && tail -100f catalina.out
```

缺点: 比较麻烦. 解决: 使用**&&**指令就可以一步到位

这个经常把一些命令组合使用，比如我们在启动tomcat后，再用tail命令查看日志。如果启动失败，则不查看

```
./startup.sh && tail -50f ..//logs/catalina.out
```

需求:

- 1 打印1.txt内容 且 打印2.txt内容
- 2 打印100.txt内容 且 打印2.txt内容(没有)
- 3 启动tomcat 且 打印日志信息

答案:

```
cat 1.txt && cat 2.txt  
cat 100.txt && cat 2.txt  
./startup.sh && tail -50f ..//logs/catalina.out
```

19.history查看所敲命令历史

1) 基本语法 :

```
history
```

2) 案例

```
[root@linux121 test]# history
```

```
[root@localhost usr]# history
 1  clear
 2  ll
 3  clear
 4  cd /
 5  clear
 6  man ls
 7  uit
 8  ll
 9  cd /
10  ll
11  pwd
12  ll
13  cd usr/
14  ll
15  touch abc
```

第五章 打包和解包命令

1.打tar包

- 类似将冬天的衣服放到袋
- 打包之后的大文件需要以 `.tar` 结尾.

`tar` 打包命令格式

```
# 将一系列文件 打包成 一个大文件
tar -cvf 打包名.tar 被打包的目录
tar -cvf 打包名.tar 被打包的文件1 被打包的文件2 被打包的文件3
```

`tar` 选项说明

命令	英文	含义
c	create	生成档案文件, 创建打包文件
v	verbosely(啰嗦的)	像 '唐僧' 一样报告进度
f	file	指定档案的文件名称, f后面一定是 .tar 文件, 所以必须放到左后

练习1：将1.txt、2.txt、3.txt 打包成 123.tar文件

练习2：将有内容的aaa目录 打包成 aaa.tar 文件

2.解tar包

- 类似将冬天的衣服从袋子里取出来

tar 解包命令格式

```
# 将一个打包后的 分解成 一系列小文件, 分解位置为 当前目录
tar -xvf 打包名.tar
```

```
# 将一个打包后的 分解成 一系列小文件, 分解位置为 指定目录
tar -xvf 打包名.tar -C 解包路径位置
```

命令	英文	含义
x	extract (提取)	解包
C (大写C)	directory (目录)	默认保存到当前目录, 通过 -C 更改解压目录, 注意: 解压目录必须存在

练习1：将 123.tar 解压到 当前目录中

练习2：将 aaa.tar 解包到 /export/test/a1/b1/c1/ 目录中

小结

打包：tar -cvf 打包之后的文件名.tar 被打包的目录或文件名

解包：tar -xvf 打包之后的文件名.tar [-C 指定解包位置]

3.gz格式 压缩和解压缩

- 打包 和 压缩 是两件事
- 类似与 先将冬天衣服放到压缩袋, 再抽取里面的空气
- 在 Linux 中, 最常用的压缩文件格式是 xxx.tar.gz
- 在 tar 命令中有一个选项 -z 可以调用 gzip , 从而可以方便的实现压缩和解压缩的功能

命令格式如下

```
# 压缩文件  
tar -zcvf 打包压缩文件名.tar.gz 被压缩的文件/目录  
  
# 解压缩文件  
tar -zxvf 打包文件.tar.gz  
  
# 解压缩到指定路径  
tar -zxvf 打包文件.tar.gz -C 目录路径
```

tar 的选项说明

命令	英文	含义
z	gzip	使用gzip压缩和解压缩
j	bzip2	使用bzip2压缩和解压缩

练习1：将1.txt、2.txt、3.txt 打包压缩成 123.tar.gz文件(gzip压缩格式)

练习2：将有内容的aaa目录 打包成 aaa.tar.gz 文件(gzip压缩格式)

练习3：将 123.tar.gz 解压到 当前目录中(gzip压缩格式)

练习4：将 aaa.tar.gz 解包到 /export/bbb 目录中(gzip压缩格式)

4.bzip2 格式 压缩和解压缩

- bzip 是压缩的第二种方式
- 类似与 先将冬天衣服放到压缩袋, 再抽取里面的空气
- 在 Linux 中, bzip2 压缩文件格式是 xxx.tar.bz2
- 在 tar 命令中有一个选项 -j 可以调用 bzip2 , 从而可以方便的实现压缩和解压缩的功能

命令格式如下

```
# 压缩文件  
tar -jcvf 打包压缩文件名.tar.bz2 被压缩的文件/目录  
  
# 解压缩文件 (绩效潍坊)  
tar -jxvf 打包文件.tar.bz2  
  
# 解压缩到指定路径  
tar -jxvf 打包文件.tar.bz2 -C 目录路径
```

注意事项：如果报错tar (child): bzip2 : 无法 exec: 没有那个文件或目录
要安装bzip2的包
yum install -y bzip2

tar 的选项说明

命令	英文	含义
z	gzip	使用gzip压缩和解压缩
j	bzip2	使用bzip2压缩和解压缩

练习1：将1.txt、2.txt、3.txt 打包压缩成 123.tar.bz2文件(bzip2压缩格式)

练习2：将有内容的aaa目录 打包成 aaa.tar.bz2 文件(bzip2压缩格式)

练习3：将 123.tar.bz2 解压到 当前目录中(bzip2压缩格式)

练习4：将 aaa.tar.bz2 解包到 /export/bbb 目录中(bzip2压缩格式)

小结

打包压缩：tar -jcvf 打包之后的文件名.tar.bz2 被打包压缩的目录或文件名

解包解压缩：tar -jxvf 打包之后的文件名.tar.bz2 [-C 指定解包位置]

第六章 时间日期

1.date 显示当前时间

基本语法：注意命令与参数之间有空格

```
(1) date (功能描述：显示当前时间)
(2) date +%Y (功能描述：显示当前年份)
(3) date +%m (功能描述：显示当前月份)
(4) date +%d (功能描述：显示当前是哪一天)
(5) date +%Y%m%d ... (功能描述：显示当前年月日各种格式)
(6) date "+%Y-%m-%d %H:%M:%S" 或者单引号也可以 (功能描述：显示年年□月□日时分秒)
```

案例

```
[root@linux121 /]# date
[root@linux121 /]# date +"%Y-%m-%d" (注意date后面有个空格再加 )
[root@linux121 /]# date "+%Y-%m-%d %H:%M:%S" (注意date后面有个空格再加 ; 注意%d空格%H)
```

显示的是字符串串描述的时间，不是当前时间。

2.date显示非当前时间

显示的是字符串串描述的时间，不是当前时间。 1) 基本语法：

```
(1) date -d '1 days ago' (功能描述：显示前一天日期)
(2) date -d yesterday +"%Y-%m-%d" (同上)
(3) date -d next-day +"%Y-%m-%d" (功能描述：显示明天日期)
(4) date -d 'next monday' (功能描述：显示下周一时间)
```

2) 案例：

```
[root@linux121 /]# date -d '1 days ago'
```

2020年年 04月 01日 星期三 21:07:22 CST

```
[root@linux121 /]# date -d 'next monday'
```

2020年年 04月 01日 星期三 00:00:00 CST

3.设置系统时间

1) 基本语法：

```
date -s 字符串时间
```

2) 案例

```
[root@hadoop106 /]# date -s "2020-06-20 20:52:18"
```

4. cal查看日历

1) 基本语法 :

```
cal [选项] ( 功能描述 : 不加选项 , 显示本月日历 )
```

选项 :

-3 , 显示系统前一个月 , 当前月 , 下一个月的日历

具体某一年年 , 显示这一年年的日历。

2) 案例 :

```
[root@linux121 /]# cal
```

```
[root@linux121 /]# cal -3
```

```
[root@linux121 /]# cal 2020
```

31
[root@linux121 ~]# cal
June 2020
Su Mo Tu We Th Fr Sa
1 2 3 4 5 6
7 8 9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30

[root@linux121 ~]#

[root@linux121 ~]# cal -3
May 2020 June 2020 July 2020
Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa
1 2 1 2 3 4 5 6 1 2 3 4 5 6 1 2 3 4
3 4 5 6 7 8 9 7 8 9 10 11 12 13 5 6 7 8 9 10 11
10 11 12 13 14 15 16 14 15 16 17 18 19 20 12 13 14 15 16 17 18
17 18 19 20 21 22 23 21 22 23 24 25 26 27 19 20 21 22 23 24 25
24 25 26 27 28 29 30 28 29 30 26 27 28 29 30 31
31
[root@linux121 ~]#

2020											
January				February				March			
Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th
1	2	3	4	5	6	7	1	2	3	4	5
5	6	7	8	9	10	11	2	3	4	5	6
12	13	14	15	16	17	18	9	10	11	12	13
19	20	21	22	23	24	25	11	12	13	14	15
26	27	28	29	30	31		12	13	14	15	16
							13	14	15	16	17
April				May				June			
Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th
1	2	3	4	5	6	7	1	2	3	4	5
5	6	7	8	9	10	11	3	4	5	6	7
12	13	14	15	16	17	18	10	11	12	13	14
19	20	21	22	23	24	25	11	12	13	14	15
26	27	28	29	30			12	13	14	15	16
							13	14	15	16	17
July				August				September			
Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th
1	2	3	4	5	6	7	1	2	3	4	5
5	6	7	8	9	10	11	2	3	4	5	6
12	13	14	15	16	17	18	10	11	12	13	14
19	20	21	22	23	24	25	11	12	13	14	15
26	27	28	29	30	31		12	13	14	15	16
							13	14	15	16	17
October				November				December			
Su	Mo	Tu	We	Th	Fr	Sa	Su	Mo	Tu	We	Th
1	2	3	4	5	6	7	1	2	3	4	5
4	5	6	7	8	9	10	8	9	10	11	12
11	12	13	14	15	16	17	15	16	17	18	19
18	19	20	21	22	23	24	16	17	18	19	20
25	26	27	28	29	30	31	22	23	24	25	26
							23	24	25	26	27
							29	30	31	27	28

第七章 搜索查找

1.find 查找文件或者目录

find命令是根据**文件的属性**进行查找，如文件名，文件大小，所有者，所属组，是否为空，访问时间，修改时间等。

基本格式：**find path [options]**

1.按照文件名查找

```
(1)find /etc -name yum.conf    #在/etc目录下文件yum.conf
(2)find /etc -name 'yum'      #使用通配符*(0或者任意多个)。表示在/etc目录下查找文件名中含有字符串' yum'的文件
(3)find . -name 'yum*'       #表示当前目录下查找文件名开头是字符串' yum'的文件
```

2.按照文件特征查找

```
(1)find / -atime -2      # 查找在系统中最后48小时访问的文件 (Access Time , 文件读取访问时间)
(2)find / -empty         # 查找在系统中为空的文件或者文件夹
(3)find / -group susan   # 查找在系统中属于group为susan的文件
(4)find / -mtime -1      #查找在系统中最后24小时里修改过的文件 (modify time)
(5)find / -user susan    #查找在系统中属于susan这个用户的文件
(6)find / -size +10000c   #查找大于10000000字节的文件(c:字节 , w:双字 , k:KB , M:MB , G:GB)
(7)find / -size -1000k    #查找小于1000KB的文件
```

3. 使用混合查找方式查找文件

参数有： ! , -and(-) , -or(-o)。

```
(1)find /tmp -size +10c -and -mtime +2      #在/tmp目录下查找大于10字节并在最后2分钟内修改的文件
(2)find / -user root -or -user susan          #在/目录下查找用户是root或者susan的文件文件
(3)find /tmp ! -user susan                   #在/tmp目录中查找所有不属于susan用户的文件
```

2 grep 过滤查找

grep是根据文件的内容进行查找，会对文件的每一行按照给定的模式(pattern)进行匹配查找。

基本格式：grep [options] 范围

1. 主要参数

[options]主要参数：

- c : 只输出匹配行的计数。
- i : 不区分大小写
- n : 显示匹配行及行号。
- w:显示整个单词
- r:递归查询

实例练习：

- 1) 在applicationContext.xml文件中查找name
- 2) 查找name统计行号
- 3) 统计jdbc的个数
- 4) 查找dataSource 并忽略大小写
- 5) 查找mapper单词
- 6) 递归查询/usr目录下 含有name的字段

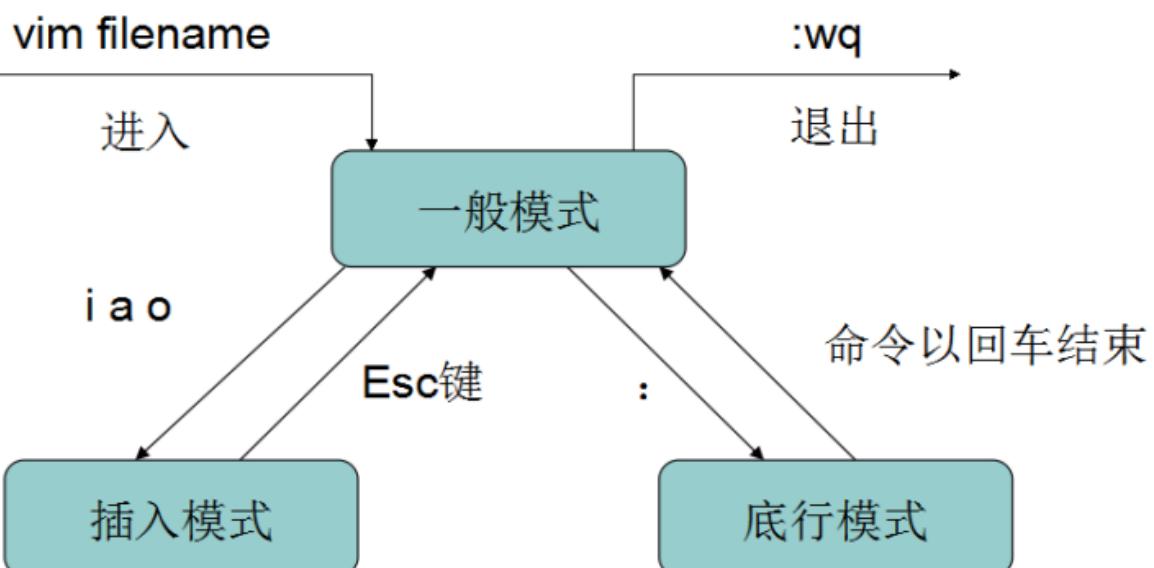
第八章 vi编辑器

- 在Linux下一般使用vi编辑器来编辑文件。

- vi既可以查看文件也可以编辑文件。
- 而vim是vi的升级版本，具备更多的功能。
- vi如果目标文件不存在，会创建新的文件。但是如果新文件没做编辑，退出后还会消失。

1. vi的三种模式介绍

- 三种模式（状态）：编辑、底行、命令模式。
 - 切换到编辑模式：按 i、o、a 键；
 - 切换到底行模式：按 :（冒号）；
 - 切换到命令行模式：按 Esc 键；
- 更多详细用法，查询文档 \相关资料\文档《Vim命令合集.docx》和《vi使用方法详细介绍.docx》



编辑模式(插入模式)：对文本进行输入和修改

底行模式：退出vim或者查找、替换功能

命令模式(一般模式)：通过快捷命令操作数据，打开vi默认就是命令模式

如果vim命令不能使用需要安装：

```
yum -y install vim-enhanced
```

2 编辑模式

命令模式按下 : i、o、a 进入编辑模式：

i：光标不动

o：另起一行

a：光标到下一个字符

按 ESC 退出编辑模式，进入命令模式

3 底行模式

命令模式下，按 ‘:’ 或者 ‘/’ 进入底行模式，可以输入命令

1) 退出 vim:(重点重点重点)

:q 未编辑时退出 vim

:q! 编辑后，退出并且不保存

:wq 编辑后，退出且保存

:x 编译后保存

2) 撤销上次操作(扩展---一般模式下)

u 撤销上一次操作 (ctrl + z windows 操作)

ctrl + r 恢复上一次被撤销的操作 (ctrl + y windows 操作)

3) 设置行号(了解) 底行模式

:set nu 显示行号

:set nonu 不显示行号

4) 替换文本(了解)

:s/old/new/ 用 new 替换 old，替换当前行的第一个匹配

:s/old/new/g 用 new 替换 old，替换当前行的所有匹配

:%s/old/new/ 用 new 替换 old，替换所有行的第一个匹配

:%s/old/new/g 用 new 替换 old，替换整个文件的所有匹配

5) 查找(一般模式)

/文本 搜索指定文本，高亮显示，按 n 显示下一个，按 N 显示前一个

:整数 快捷跳转到指定行

4 命令模式(一般模式)

p(pause) 将之前dd或yy的数据粘贴到光标位置

yy 复制光标所在行

5yy 复制光标及下面共5行

dd 剪切当前行

5dd 剪切光标及下面共5行

第九章 用户及组管理

1 useradd 添加新用户

(注意:当前用户必须有添加用户的权限)

1) 基本语法 :

```
useradd 用户名 ( 功能描述 : 添加新用户 )
```

2) 案例 :

```
[root@linux121 ~]# useradd hadoop
```

2 passwd 设置用户密码

1) 基本语法 :

```
passwd 用户名 ( 功能描述 : 设置用户密码 )
```

2) 案例

```
[root@linux121 ~]# passwd hadoop
```

```
[root@linux121 ~]# useradd hadoop
[root@linux121 ~]# passwd hadoop
Changing password for user hadoop.
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: all authentication tokens updated successfully.
[root@linux121 ~]#
```

3 id 判断用户是否存在

1) 基本语法 :

```
id 用户名
```

2) 案例 :

```
[root@linux121 ~]#id hadoop
```

```
[root@linux121 ~]# id hadoop
uid=1000(hadoop) gid=1000(hadoop) groups=1000(hadoop)
[root@linux121 ~]#
```

4 su 切换用户

1) 基本语法 :

```
su 用户名称 ( 功能描述 : 切换用户 , 只能获得用户的执行权限 , 不能获得环境变量 )
```

```
su - 用户名称 ( 功能描述 : 切换到用户并获得该用户的环境变量及执行权限 )
```

2) 案例

```
[root@linux121 ~]#su hadoop
```

```
[root@linux121 ~]#su - hadoop
```

5 userdel 删除用户

1) 基本语法 : 注意区别

```
( 1 ) userdel 用户名 ( 功能描述 : 删除用户但保存用户主目录 )
```

```
( 2 ) userdel -r 用户名 ( 功能描述 : 用户和用户主目录 , 都删除 )
```

2) 案例 :

```
( 1 ) 删除用户但保存用户主目录
```

```
[root@linux121 ~]#userdel hadoop
```

```
( 2 ) 删除用户和用户主目录 , 都删除
```

```
[root@linux121 ~]#userdel -r hadoop
```

6 who 查看登录用户信息

1) 基本语法

```
( 1 ) whoami ( 功能描述 : 显示自身用户名称 )  
( 2 ) who am i ( 功能描述 : 显示登录用户的用户名 )  
( 3 ) who ( 功能描述 : 看当前有哪些用户登录到了本台机器器上 )
```

2) 案例

```
[root@linux121 ~]# whoami  
[root@linux121 ~]# who am i  
[root@linux121 ~]# who
```

7 设置Linux普通用户具有root权限即sudo的使用

1) sudo命令

sudo是linux系统管理指令，是允许 **系统管理员** 让普通用户执行一些或者全部的root命令的一个工具，如halt，reboot，su等等。这样不仅减少了root用户的登录和管理时间，同样也提高了安全性。

2) 修改配置文件 修改 /etc/sudoers 文件，找到下面一行，在root下面添加一行，如下所示：

vim /etc/sudoers

```
## Allow root to run any commands anywhere  
root ALL=(ALL) ALL  
tom ALL=(ALL) ALL
```

3) 使用tom用户登录,操作管理员命令

本质:使用临时管理员权限

```
#不切换root用户,也可以完成添加用户的功能  
sudo useradd lisi  
sudo passwd lisi
```

8 cat /etc/passwd 查看创建了哪些用户

```
cat /etc/passwd
```

9 用户组管理命令

每个用户都有一个用户组，系统可以对一个用户组中的所有用户进行集中管理。不同Linux系统对用户组的规定有所不同，如Linux下的用户属于与它同名的用户组，这个用户组在创建用户时同时创建。用户组的管理涉及用户组的添加、删除和修改。组的增加、删除和修改实际上就是对/etc/group文件的更新

groupadd 新增组

1) 基本语法

```
groupadd 组名
```

2) 案例：添加一个hadoop组

```
[root@linux121 ~]# groupadd hadoop
```

groupdel 删除组

1) 基本语法：

```
groupdel 组名
```

2) 案例例

```
[root@linux121 ~]# groupdel hadoop
```

groupmod 修改组

1) 基本语法：

```
groupmod -n 新组名 老组名
```

2) 案例 修改hadoop组名称为hadoop1

```
[root@linux121 hadoop]# groupmod -n hadoop1 hadoop
```

cat /etc/group 查看创建了哪些组

```
cat /etc/group
```

10 usermod修改用户

1) 基本语法 :

```
usermod -g 用户组 用户名
```

2) 案例 : 将用户hadoop加入mygroup用户组

```
[root@linux121 ~]#usermod -g mygroup hadoop
```

第十章 文件权限

Linux系统是一种典型的多用户系统 , 不同的用户处于不同的地位 , 拥有不同的权限。为了保护系统的安全性 , Linux系统对不同的用户访问同一文件 (包括目录文件) 的权限做了不同的规定。在Linux中我们可以使用ll或者ls -l命令来显示一个文件的属性以及文件所属的用户和组。

1. 文件权限

The diagram shows the output of the 'll' command on a Linux system. The output is as follows:

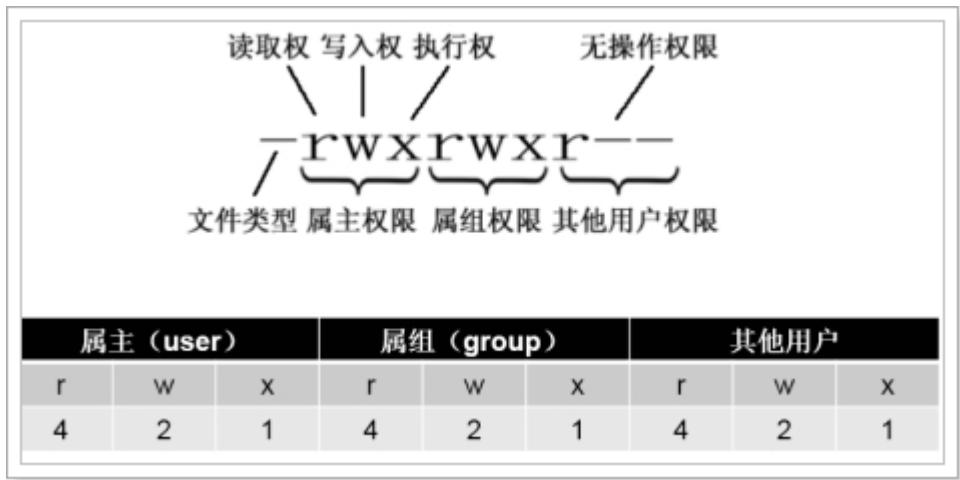
总用量 8							
-rwxrwxrwx	1	zhangsan	dev	0	5月 22 17:37	01.txt	
-rwxrwxrwx	1	zhangsan	dev	0	5月 22 17:37	02.txt	
-rwxrwxrwx	1	zhangsan	dev	0	5月 22 17:37	03.txt	
drwxr-xr-x	2	zhangsan	dev	4096	5月 22 17:40	jkd1.8	
drwxr-xr-x	2	zhangsan	dev	4096	5月 22 17:40	soft	

Annotations explain the columns:

- 文件类型 (File Type): Points to the first column.
- 权限 (Permissions): Points to the second column.
- 所属用户 (Owner): Points to the third column.
- 所属组 (Group): Points to the fourth column.
- 大小 (Size): Points to the fifth column.
- 末次访问时间 (Last Accessed): Points to the sixth column.
- 文件名 (File Name): Points to the seventh column.

Legend for file types:

- 文件 (File)
- d 目录 (Directory)
- l 链接 (Link)



权限分段，每一段代表不同的用户：

属主权限：当前文件所属用户的权限，在Linux中每一个文件都有所属的用户

属组权限：当前文件的用户所在组的其它成员的权限

其它用户权限：跟文件所属用户不在同组的其它用户的权限

每一段中的内容都是一样的，分别限定：读 (r=4)、写 (w=2)、执行 (x=1)

rwx	7	具备所有权限(421)
rw-	6	具备读写权限(420)
r-x	5	具备读和执行权限(401)
r--	4	具备只读权限(400)
-wx	3	具备写和执行权限(021)
-w-	2	具备写权限(020)
--x	1	具备执行权限(001)
---	0	000

问题:

- 1 777 什么意思?
- 2 644 什么意思?
- 3 755 什么意思?

答案

777 所属用户具有 读写执行权限，所属同组用户具有 读写执行权限，其他人具有 读写执行权限
 644 所属用户具有 读写权限，所属同组用户具有 读权限，其他人具有 读权限
 755 所属用户具有 读写执行权限，所属同组用户具有 读执行权限，其他人具有 读执行权限

2 文件权限管理

修改文件权限

文件权限

● chmod修改文件权限

- `chmod 755 a.txt`
- `chmod u=rwx,g=rx,o=rx a.txt`
- 上面的两种方式是等效的。将[a.txt](#)这个文件修改为：
 - 所属用户具备所有权限：`rwx = 7`
 - 本组用户具备读和执行权限：`rx = 5`
 - 其它用户具备读和执行权限：`rx = 5`

需求：

- 1 新增 /usr/tmp/1.txt 文件 且 查看 权限
- 2 修改权限为 777(方式一)
- 3 修改权限为 755(方式二)

答案

```
[root@itheima16 tmp]# touch /usr/tmp/1.txt
[root@itheima16 tmp]# ll
总用量 0
-rw-r--r--. 1 root root 0 7月 24 05:15 1.txt
[root@itheima16 tmp]# chmod 777 1.txt
[root@itheima16 tmp]# ll
总用量 0
-rwxrwxrwx. 1 root root 0 7月 24 05:15 1.txt
[root@itheima16 tmp]# chmod u=rwx,g=rx,o=rx 1.txt
[root@itheima16 tmp]# ll
总用量 0
-rwxr-xr-x. 1 root root 0 7月 24 05:15 1.txt
```

3 chown改变所有者

1) 基本语法：

```
chown 最终用户 ( 功能描述：改变文件或者目录的所有者 )
```

2) 案例

```
[root@linux121 test]# chown hadoop test1.txt  
[root@linux121 test]# ls -al
```

4 chgrp改变所属组

1) 基本语法 :

```
chgrp 最终用户组 (功能描述 : 改变文件或者目录的所属组)
```

2) 案例

```
[root@linux121 test]# chgrp hadoop test1.java  
[root@linux121 test]# ls -al  
-rwxr-xr-x. 1 root hadoop 551 4月 23 13:02 test1.java
```

第十一章 系统管理与进程管理

1.磁盘信息

序号	命令	作用
1	df -h	disk free 显示磁盘剩余空间
2	du -h [目录名称]	disk usage 显示目录下的目录大小

- 选项说明

参数	含义
-h	以人性化的方式显示文件大小

2 ps查看系统中所有进程

进程是正在执行的一个程序或命令，每一个进程都是一个运行的实体，都有自己的地址空间，并占用一定的系统资源。

1) 基本语法 :

```
ps -aux ( 功能描述 : 查看系统中所有进程 )
```

选项	含义
a	显示终端上的所有进程, 包括其他用户的进程
u	显示进程的详细状态
x	显示没有控制终端的进程

注意:

ps aux 和 ps -ef 两者的输出结果差别不大, 但展示风格不同。aux是BSD风格, -ef是System V风格

2) 功能说明

USER : 该进程是由哪个用户产生的

PID : 进程的ID号

%CPU : 该进程占用CPU资源的百分比, 占用越高, 进程越耗费资源;

%MEM : 该进程占用物理内存的百分比, 占用越高, 进程越耗费资源;

VSZ : 该进程占用虚拟内存的大小, 单位KB;

RSS : 该进程占用实际物理内存的大小, 单位KB;

TTY : 该进程是在哪个终端中运行的。其中tty1-tty7代表本地控制台终端, tty1-tty6是本地的字符界面终端, tty7是图形终端。pts/0-255代表虚拟终端。

STAT : 进程状态。常见的状态有: R : 运行、 S : 睡眠、 T : 停止状态、 s : 包含子进程、 + : 位于后台

START : 该进程的启动时间

TIME : 该进程占用CPU的运算时间, 注意不是系统时间

COMMAND : 产生此进程的命令名

3 top查看系统健康状态

1) 基本命令

```
top [选项]
```

(1) 选项 :

-d 秒数 : 指定top命令每隔几秒更新。默认是3秒。

-i : 使top不显示任何闲置或者僵死进程。

-p : 通过指定监控进程ID来仅仅监控某个进程的状态。

(2) 操作选项 , 即在执行top命令之后 , 与top命令进行交互 :

P : 以CPU使用率排序 , 默认就是此项

M : 以内存的使用率排序

N : 以PID排序

q : 退出top

(3) 查询结果字段解释

top - 02:15:17 up 2:54, 1 user, load average: 0.00, 0.01, 0.05 系统运行的时间
Tasks: 93 total, 2 running, 91 sleeping, 0 stopped, 0 zombie 运行进程相关
%Cpu(s): 0.3 us, 0.3 sy, 0.0 ni, 99.3 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st CPU相关
KiB Mem : 999696 total, 79000 free, 146640 used, 774056 buff/cache 内存相关
KiB Swap: 2097148 total, 2097148 free, 0 used. 629296 avail Mem 交换分区相关

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
660	root	20	0	305296	6312	4928	S	0.3	0.6	0:17.92	vmtoolsd
6597	root	20	0	157584	2140	1516	R	0.3	0.2	0:02.14	top
1	root	20	0	128164	6820	4060	S	0.0	0.7	0:06.33	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.01	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:00.54	ksoftirqd/0
5	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	kworker/0:0H
7	root	rt	0	0	0	0	S	0.0	0.0	0:00.00	migration/0
8	root	20	0	0	0	0	S	0.0	0.0	0:00.00	rcu_bh
9	root	20	0	0	0	0	R	0.0	0.0	0:01.91	rcu_sched
10	root	rt	0	0	0	0	S	0.0	0.0	0:00.10	watchdog/0
12	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kdevtmpfs
13	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	netns
14	root	20	0	0	0	0	S	0.0	0.0	0:00.00	khungtaskd
15	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	writeback
16	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	kintegrityd
17	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	bioset

第一行信息为任务队列信息

内容	说明
13:35:15	系统当前时间
up 5:25	系统的运行时间, 本机已经运行5小时25分钟
1 users	当前登录了一个用户
load average: 0.00, 0.01, 0.05	系统在之前1分钟, 5分钟, 15分钟的平均负载。一般认为小于1时, 负载较小。如果大于1, 系统已经超出负荷。

第二行为进程信息

Tasks: 95 total	系统中的进程总数
2 running	正在运行的进程数
92 sleeping	睡眠的进程
0 stopped	正在停止的进程
0 zombie	僵尸进程。如果不是0，需要手工检查僵尸进程

第三行为CPU信息

Cpu(s): 0.1%us	用户模式占用的CPU百分比
0.3%sy	系统模式占用的CPU百分比
0.0%ni	改变过优先级的用户进程占用的CPU百分比
99.7%id	空闲CPU的CPU百分比
0.0%wa	等待输入/输出的进程的占用CPU百分比
0.0%hi	硬中断请求服务占用的CPU百分比
0.0%si	软中断请求服务占用的CPU百分比
0.0%st	st (Steal time) 虚拟时间百分比。就是当有虚拟机时，虚拟CPU等待实际CPU的时间百分比。

第四行为物理内存信息

Mem: 995896k total	物理内存的总量, 单位KB
92580k used	已经使用的物理内存数量
638392k free	空闲的物理内存数量
264924k buffers	作为缓冲的内存数量

第五行为交换分区 (swap) 信息

Swap: 2097148k total	交换分区（虚拟内存）的总大小
0k used	已经使用的交互分区的大小
2097148k free	空闲交换分区的大小
719608 avail Mem	可用交换区总量

2) 案例

```
[root@linux121 hadoop]# top -d 1  
[root@linux121 hadoop]# top -i  
[root@linux121 hadoop]# top -p 2575
```

执行上述命令后，可以按P、M、N对查询出的进程结果进行排序

4 kill终止进程

1) 基本语法：

```
kill -9 pid进程号
```

选项 -9 表示强迫进程立即停止

举例:

```
[root@linux121 ~]# kill -9 5102
```

5 netstat显示网络统计信息

1) netstat作用

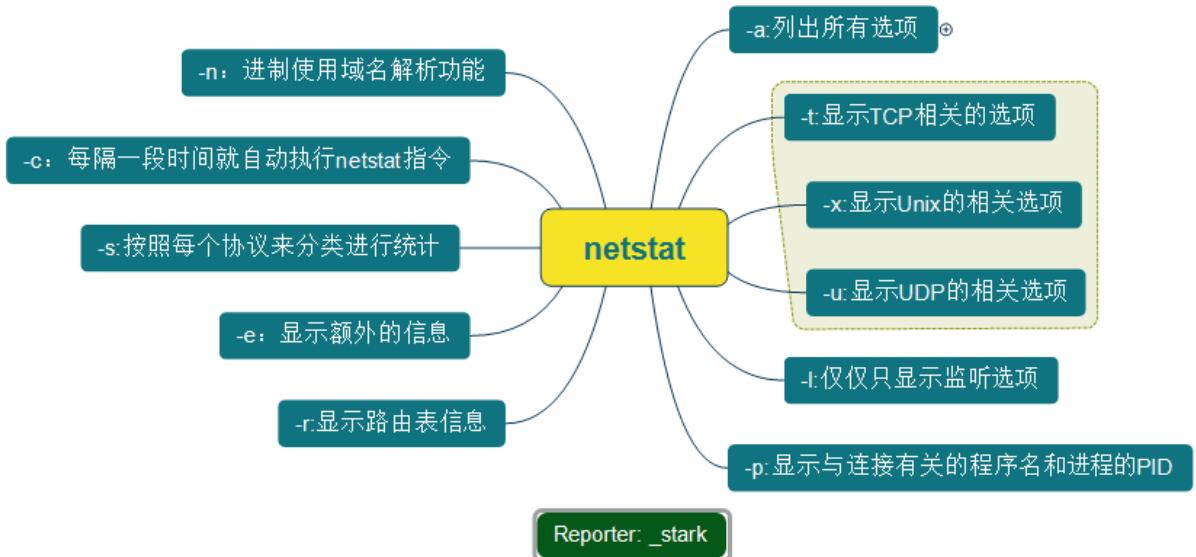
netstat命令用于显示与IP、TCP、UDP和ICMP协议相关的统计数据，一般用于检验本机各端口的网络连接情况

2) 安装命令

```
yum install net-tools
```

3) 基本语法

选项：



```
NETSTAT [-a] [-b] [-e] [-n] [-o] [-p proto] [-r] [-s] [-v] [interval]
-a          显示所有连接和监听端口。
-e          显示以太网统计信息。它列出的项目包括传送的数据报的总字节数、错误数、删除数、数据报的数量和广播的数量
-n          以数字形式显示地址和端口号。
-o          显示与每个连接相关的所属进程 ID 。
-p proto    显示 proto 指定的协议的连接；proto 可以是下列协议之一：TCP、UDP、TCPv6 或 UDPv6。
```

常用选项

`netstat -e` 本选项用于显示关于以太网的统计数据。

`netstat -r` 本选项可以显示关于路由表的信息，类似于后面所讲使用`route print` 命令时看到的 信息。除了显示有效路由外，还显示当前有效的连接。

`netstat -a` 本选项显示一个所有的有效连接信息列表，包括已建立的连接（ESTABLISHED），也包括监听连接请求（LISTENING）的那些连接，断开连接（CLOSE_WAIT）或者处于联机等待状态的（TIME_WAIT）等

`netstat -n` 显示所有已建立的有效连接。

组合用法:

`netstat -anp` 查看这个系统目前网络状况。

查看端口22的使用情况

```
[root@linux121 hadoop-2.7.2]# netstat -anp | grep 22
```

```
[root@linux121 test]# netstat -anp | grep 22
tcp      0      0 0.0.0.0:22          0.0.0.0:*
              ::::*
LISTEN      6801/sshd
tcp6     0      0 ::::22          ::::*
LISTEN      6801/sshd
unix  3      [ ]      STREAM   CONNECTED    38122  6800/python2
unix  3      [ ]      STREAM   CONNECTED    22419  3139/systemd-udevd
unix  3      [ ]      STREAM   CONNECTED    22420  1/systemd           /run/systemd/journal
/stdout
unix  3      [ ]      STREAM   CONNECTED    22171  3126/lvmetad
unix  3      [ ]      DGRAM        22451  3139/systemd-udevd
unix  3      [ ]      DGRAM        22452  3139/systemd-udevd
unix  3      [ ]      STREAM   CONNECTED    22172  1/systemd           /run/systemd/journal
/stdout
unix  2      [ ]      DGRAM        22433  3139/systemd-udevd
[root@linux121 test]#
```

第十二章 软件安装

1) rpm 软件包管理器

1) 目标

- 通过 `rpm命令` 实现对软件的安装、查询、卸载
- RPM 是Red-Hat Package Manager (RPM软件包管理器) 的缩写
- 虽然打上了 red-hat 的标记,但是理念开放,很多发行版都采用,已经成为行业标准

2) 路径

- 第一步: rpm包 的查询命令
- 第二步: rpm包 的卸载
- 第三步: rpm包 的安装

3) 实现

第一步: rpm包 的查询命令

选项	英文	含义
<code>-q</code>	query	查询
<code>-a</code>	all	所有
<code>-i</code>	info	信息
<code>-l</code>	list	显示所有相关文件
<code>-f</code>	file	文件, 显示文件对应 <code>rpm</code> 包

- 查询已安装的rpm列表

```
rpm -qa | grep XXX  
rpm -qa | less
```

- 查询软件包信息

```
rpm -qi 软件全包名
```

- 查看一个rpm包中的文件安装到那里去了？

```
rpm -ql 软件全包名
```

- 查看指定文件归属于那个软件包

```
rpm -qf 文件的全路径
```

第二步: rpm包的卸载

命令	英文	含义
rpm -e 软件包名称	erase 清除	卸载rpm软件包
rpm -e --nodeps 软件包名称	Don't check dependencies	卸载前跳过依赖检查

第三步: rpm包的安装

命令	含义
rpm -ivh rpm包的全路径	安装 rpm 包

参数	英文	含义
-i	install	安装
-v	verbose	打印提示信息
-h	hase	显示安装进度

2.4 小结

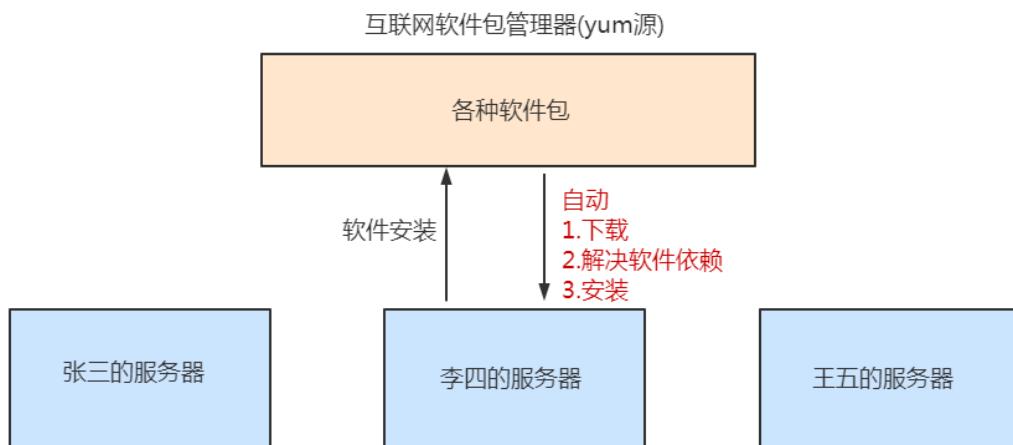
```
# 1 查询  
rpm -qa | grep rpm包  
  
# 2 卸载  
rpm -e rpm全包名  
rpm -e --nodeps rpm全包名  
  
# 3 安装  
rpm -ivh rpm包的全路径
```

2 yum

1) yum介绍

- Yum (全称为 Yellow dog Updater, Modified) 本质上也是一个软件包管理器。
- 特点: 基于 RPM 包管理, 能够从指定的服务器 自动下载、自动安装、自动处理依赖性关系

2) yum原理



3) 常用命令

注意: 必须联网

命令	含义
<code>yum list grep 需要的软件名</code>	yum list显示所有已经安装和可以安装的程序包
<code>yum -y install 需要的软件包</code>	下载安装
<code>yum -y remove 需要卸载的软件包</code>	卸载
<code>yum repolist</code>	列出设定yum源信息
<code>yum clean all</code>	清除yum缓存信息

3 安装httpd软件

- 安装httpd:

```
yum -y install httpd
```

- 启动 httpd 服务

```
service httpd start
```

- 测试

```
http://192.168.80.100:80
```

- 问题: 发现无法访问?

- 原因: 因为 linux 的防火墙 禁止他人 访问自己的80端口
- 解决: 通知 防火墙放行

关闭防火墙

```
systemctl stop firewalld    查看防火墙状态
```

4 安装JDK

系统默认安装OpenJDK，首先需要删除OpenJDK

- 1、查看以前是不是安装了openjdk

如果不是root用户需要切换到root用户 (su - root)

命令 : `rpm -qa | grep java`

显示如下：（有则卸载，没有就不用），注意版本可能会有些不一样，以实际操作的为准。

```
tzdata-java-2013g-1.el6.noarch  
java-1.7.0-openjdk-1.7.0.45-2.4.3.3.el6.x86_64  
java-1.6.0-openjdk-1.6.0.0-1.66.1.13.0.el6.x86_64
```

2、卸载openjdk：

（其中参数“tzdata-java-2013g-1.el6.noarch”为上面查看中显示的结果，粘进来就行，如果你显示的不一样，请复制你查询到的结果）

```
rpm -e --nodeps java-1.6.0-openjdk-1.6.0.0-1.66.1.13.0.el6.i686 rpm -e --nodeps java-1.7.0-openjdk-1.7.0.45-  
2.4.3.3.el6.i686 rpm -e --nodeps tzdata-java-2013g-1.el6.noarch
```

-e	erase package(uninstall)	移除包(卸载)
--nodeps	do not verify package dependencies	不要验证包依赖关系

3、安装jdk

- (1)、切换到root用户并进入usr目录： cd /usr
- (2)、在usr目录下创建java文件夹： mkdir java
- (3)、将jdk-7u71-linux-x64.tar.gz拷贝或上传到java目录下（也可以用工具）
- (4)、进入/usr/java文件夹下： cd /usr/java/
- (5)、修改权限，参数“jdk-7u71-linux-x64.tar.gz”为你自己上传的jdk安装文件

```
chmod 755 jdk-7u71-linux-x64.tar.gz
```

- (6)、解压：tar -zvxf jdk-7u71-linux-x64.tar.gz

- (7)、配置环境变量

```
vi /etc/profile
```

添加内容：

```
export 主要用来设置环境变量
```

```
export JAVA_HOME=/usr/java/jdk1.8.0_141/  
export  
CLASSPATH=.:${JAVA_HOME}/jre/lib/rt.jar:${JAVA_HOME}/lib/dt.jar:${JAVA_HOME}/lib/tools.jar  
export PATH=$PATH:${JAVA_HOME}/bin
```

- (8)、重新编译环境变量

```
source /etc/profile
```

- (9)、测试

```
java -version
```

5 安装tomcat

安装Tomcat

1、创建安装目录：mkdir /usr/tomcat

2、给apache-tomcat-7.0.57.tar.gz 文件权限

3、解压tomcat压缩文件：`tar -zxvf apache-tomcat-7.0.57.tar.gz -C /user/tomcat`

注意，这里也可以只打开tomcat所需端口：8080

4、启动tomcat：

进入tomcat的bin目录：`cd /usr/tomcat/apache-tomcat-7.0.57/bin/`

启动tomcat web服务器：`./startup.sh`

访问：`192.168.80.100:8080`

5、停止tomcat

：`./shutdown.sh`

6、查看tomcat日志信息：

```
tail -200f /usr/tomcat/apache-tomcat-7.0.57/logs/catalina.out
```

200表示最后显示行数

也可以用组合命令，启动并查看日志：

进入tomcat的bin目录

```
./startup.sh && tail -200f ../logs/catalina.out
```

6 安装mysql

- 下面看看如何在CentOS7系统上安装 MySQL5.6.44

1 查询系统自带的mysql

```
[root@hadoop01 yum.repos.d]# rpm -qa | grep mysql
mysql-libs-5.1.73-8.el6_8.x86_64
```

2 卸载系统自带的mysql

```
rpm -e --nodeps mysql-libs-5.1.73-8.el6_8.x86_64
```

3 下载安装官网yum源

查看yum源仓库
ll /etc/yum.repos.d/
下载yum源
wget -P /usr/software http://repo.mysql.com/mysql-community-release-el6-5.noarch.rpm

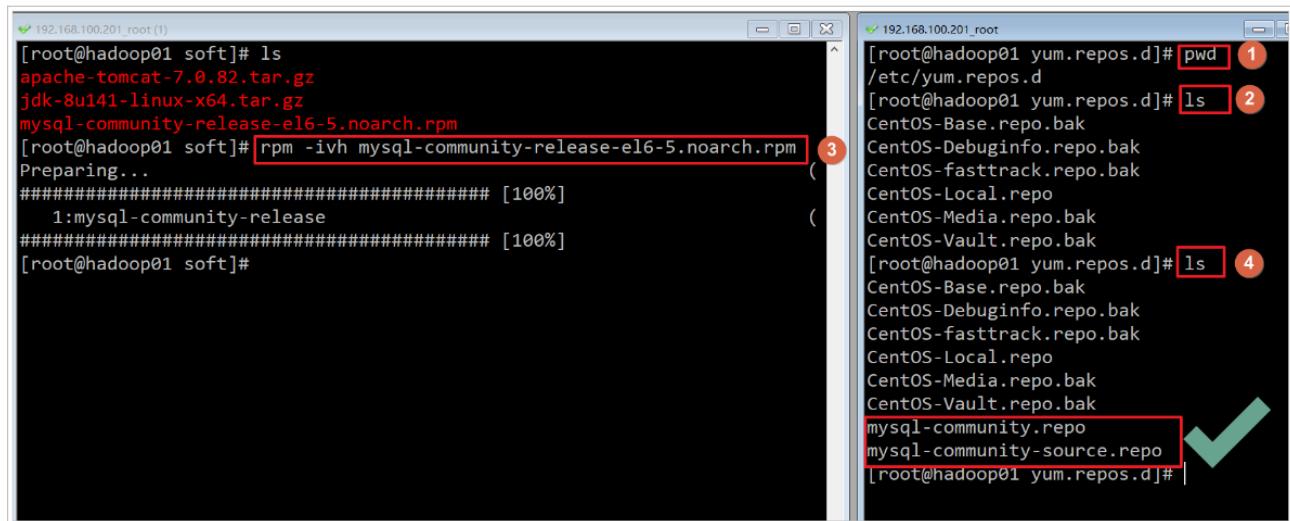
这里要是不能安装 需要yum install wget
在执行这个命令

- 注意: 需要联网

```
[root@hadoop01 soft]# wget -P /export/soft http://repo.mysql.com/mysql-community-release-el6-5.noarch.rpm  
--2019-05-21 07:57:27-- http://repo.mysql.com/mysql-community-release-el6-5.noarch.rpm  
正在解析主机 repo.mysql.com... 23.41.23.231  
正在连接 repo.mysql.com[23.41.23.231]:80... 已连接。  
已发出 HTTP 请求, 正在等待回应... 200 OK  
长度: 5824 (5.7K) [application/x-redhat-package-manager]  
正在保存至: "/export/soft/mysql-community-release-el6-5.noarch.rpm"  
  
100%[=====] 5,824      --.-K/s  
  
2019-05-21 07:57:28 (527 MB/s) - 已保存 "/export/soft/mysql-community-release-el6-5.noarch.rpm" [5824/5824])  
  
[root@hadoop01 soft]# pwd  
/export/soft  
[root@hadoop01 soft]# ll  
总用量 189972  
-rw-r--r-- 1 root root 8997403 1月 24 2018 apache-tomcat-7.0.82.tar.gz  
-rw-r--r-- 1 root root 185516505 10月 4 2018 jdk-8u141-linux-x64.tar.gz  
-rw-r--r-- 1 root root 5824 11月 12 2015 mysql-community-release-el6-5.noarch.rpm  
[root@hadoop01 soft]# |
```

4 安装下载好的rpm文件

```
cd /usr/software  
rpm -ivh mysql-community-release-el6-5.noarch.rpm
```



5 安装mysql 服务器

```
yum -y install mysql-community-server
```

- 注意:需要联网下载,估计需要等待十几分钟!
- 测试是否安装成功

```
[root@hadoop01 soft]# rpm -qa | grep mysql
mysql-community-client-5.6.44-2.el6.x86_64
mysql-community-release-el6-5.noarch
mysql-community-libs-5.6.44-2.el6.x86_64
mysql-community-server-5.6.44-2.el6.x86_64
mysql-community-common-5.6.44-2.el6.x86_64
```

6 启动服务

```
service mysqld start
```

如果出现:serivce: command not found
安装service
yum install initscripts

```
[root@hadoop01 soft]# service mysqld status
mysqld 已停                                         查看mysql状态
[root@hadoop01 soft]# service mysqld start
                                                 启动mysql
```

7 修改密码

```
192.168.100.201_root x 192.168.100.201_root (1)

PLEASE REMEMBER TO SET A PASSWORD FOR THE MySQL root USER !
To do so, start the server, then issue the following commands:

/usr/bin/mysqladmin -u root password 'new-password'
/usr/bin/mysqladmin -u root -h hadoop01 password 'new-password'
```

```
# 设置密码  
/usr/bin/mysqladmin -u root password '123'  
  
# 进入mysql  
mysql -uroot -p123
```

8 问题1: 解决中文乱码

由于MySQL编码原因会导致数据库出现乱码。

解决办法 :

修改MySQL数据库字符编码为UTF-8 , UTF-8包含全世界所有国家需要用到的字符,是国际编码。

具体操作 :

1 进入MySQL控制台

```
# 进入mysql  
mysql -uroot -p123  
  
# 查看编码集 发现不是utf8  
show variables like 'character_set_%';
```

```
mysql> show variables like 'character_set_%';  
+-----+-----+  
| Variable_name      | Value |  
+-----+-----+  
| character_set_client | utf8  |  
| character_set_connection | utf8  |  
| character_set_database | latin1 |  
| character_set_filesystem | binary |  
| character_set_results | utf8  |  
| character_set_server | latin1 |  
| character_set_system | utf8  |  
| character_sets_dir   | /usr/share/mysql/charsets/ |  
+-----+-----+  
8 rows in set (0.00 sec)
```

2 修改mysql配置文件

```
# 清空 mysql 配置文件内容  
[root@Hadoop-NN-01 ~]# >/etc/my.cnf  
  
# 修改mysql 软件的编码集  
[root@Hadoop-NN-01 ~]# vi /etc/my.cnf
```

修改内容如下:

```
[client]  
default-character-set=utf8  
  
[mysql]  
default-character-set=utf8  
  
[mysqld]  
character-set-server=utf8
```

3 重启MySQL服务

```
[root@Hadoop-NN-01 ~]# service mysqld restart
```

```
#查看MySQL字符集  
show variables like 'character_set_%';
```

```
mysql> show variables like 'character_set_%';  
+-----+-----+  
| Variable_name      | Value |  
+-----+-----+  
| character_set_client | utf8 |  
| character_set_connection | utf8 |  
| character_set_database | utf8 |  
| character_set_filesystem | binary |  
| character_set_results | utf8 |  
| character_set_server | utf8 |  
| character_set_system | utf8 |  
| character_sets_dir | /usr/local/mysql/share/charsets/ |  
+-----+-----+  
8 rows in set (0.00 sec)
```

MySQL数据库字符集编码修改完成！

9 问题2: 默认情况下 mysql 服务端不允许客户端远程访问

问题: 使用客户端远程连接mysql报错?



- 原因: 因为 用户 没有 远程访问的权限
- 解决: 授权

```
# 给root授权:既可以本地访问, 也可以远程访问
grant all privileges on *.* to 'root'@'%' identified by '123' with grant option;

# 刷新权限(可选)
flush privileges;
```

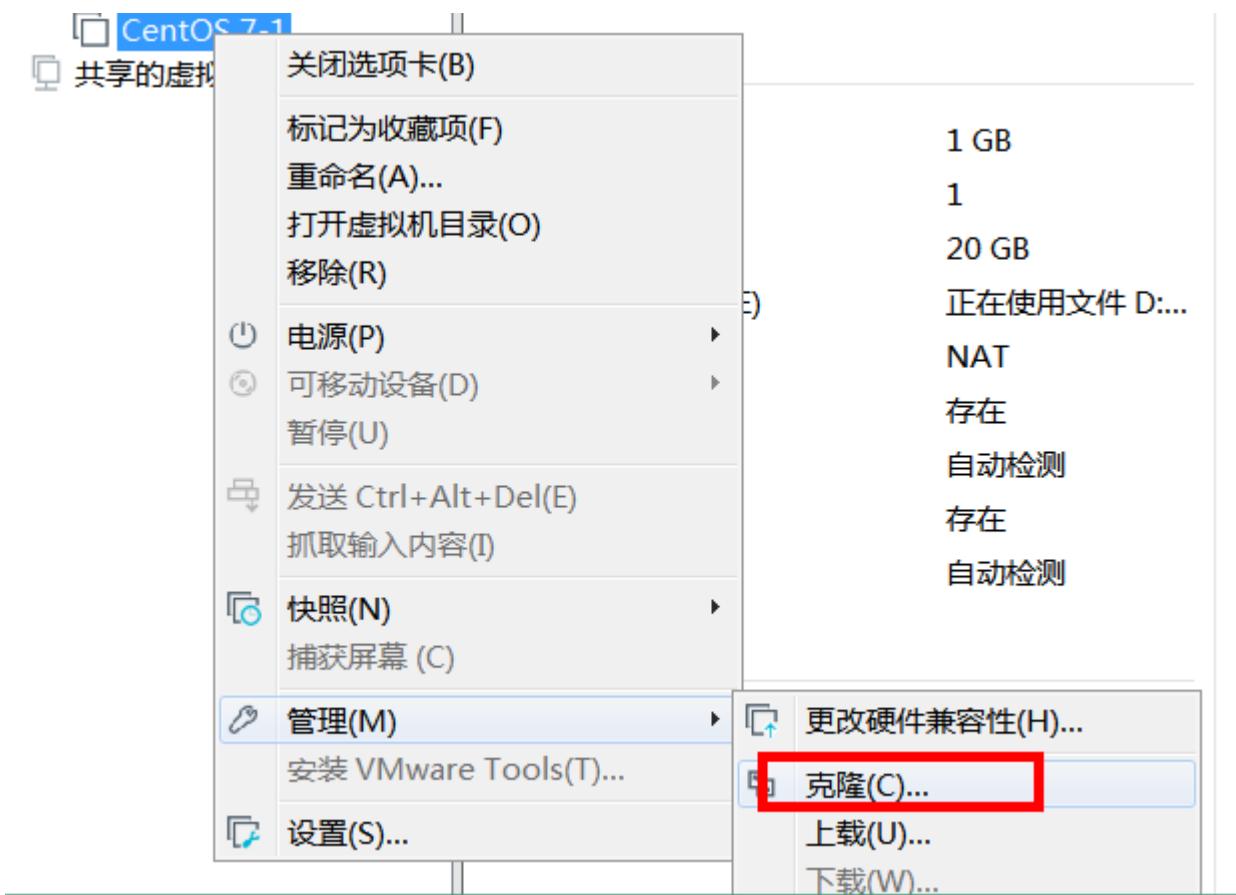
第十三章 大数据linux操作

1.克隆2台机器

1.克隆操作

关闭当前centos7-1

右键点击克隆



克隆虚拟机向导

X

克隆类型

您希望如何克隆此虚拟机？

克隆方法

创建链接克隆(L)

链接克隆是对原始虚拟机的引用，所需的存储磁盘空间较少。但是，必须能够访问原始虚拟机才能运行。

创建完整克隆(F)

完整克隆是原始虚拟机当前状态的完整副本。此副本虚拟机完全独立，但需要较多的存储磁盘空间。

< 上一步(B)

下一步(N) >

取消

克隆虚拟机向导

X

新虚拟机名称

您希望该虚拟机使用什么名称?

虚拟机名称(V)

CentOS 7-2

位置(L)

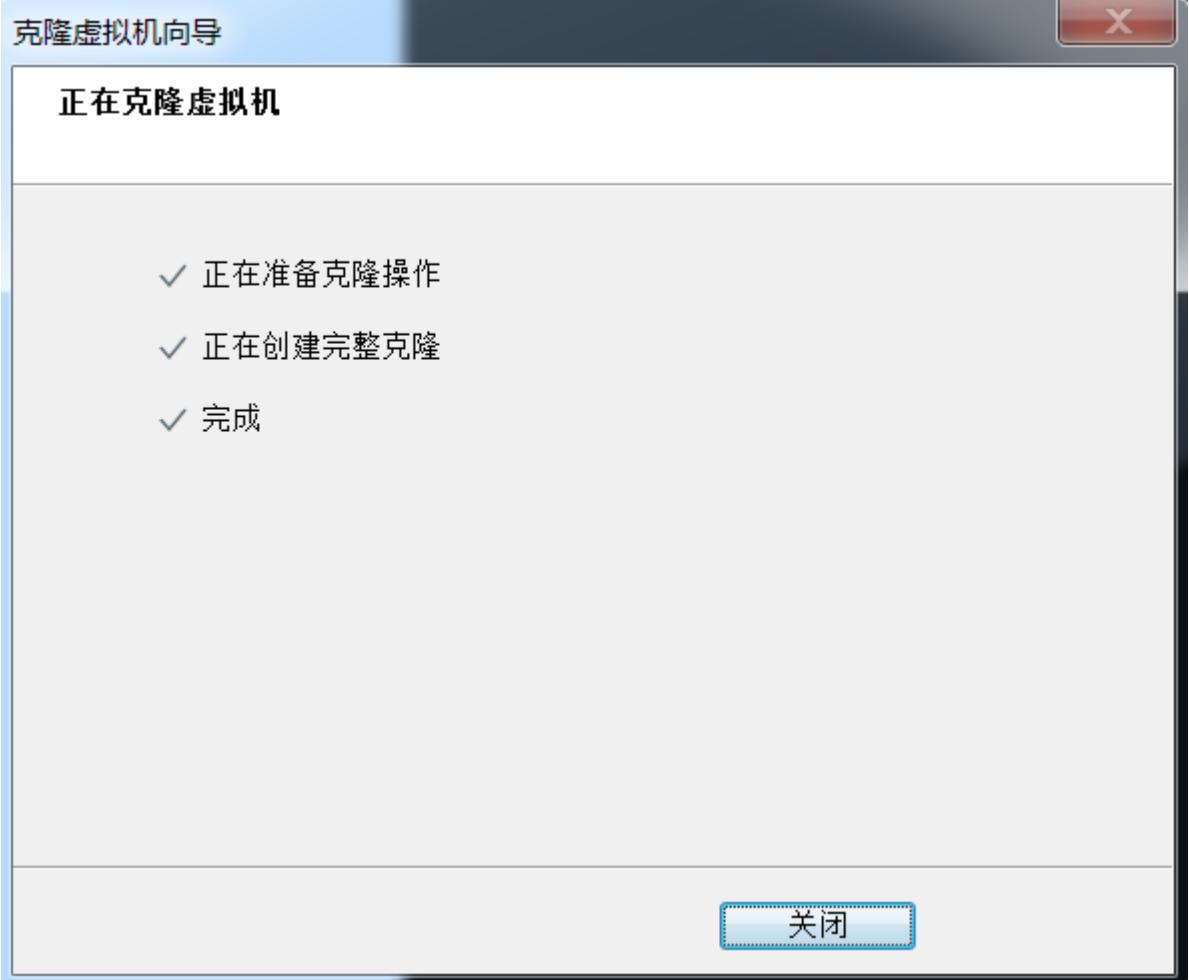
E:\developer\centos7-2

浏览(R)...

< 上一步(B)

完成

取消



2.配置克隆后机器

- 网络配置文件ifcfg-ens33的ip地址

```
vi /etc/sysconfig/network-scripts/ifcfg-ens33
```

```
TYPE=Ethernet
PROXY_METHOD=none
BROWSER_ONLY=no
BOOTPROTO=static
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=a9bb28aa-c5d5-4a88-a3d2-b3f4434b8e62
DEVICE=ens33
ONBOOT=yes
IPADDR=192.168.80.102
NETMASK=255.255.255.0
GATEWAY=192.168.80.2
DNS1=8.8.8
~
```

重启网卡服务器

```
sudo systemctl restart network
```

检查ip地址是否变更

```
ip addr
```

```
"/etc/sysconfig/network-scripts/ifcfg-ens33" 19L, 360C written
[root@localhost ~]# systemctl network restart
Unknown operation 'network'.
[root@localhost ~]# systemctl restart network
[root@localhost ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN qlen 1
    link/loopback 00:00:00:00:00 brd 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_fast state UP qlen 1000
    link/ether 00:0c:29:74:86:e0 brd ff:ff:ff:ff:ff:ff
        inet 192.168.80.102/24 brd 192.168.80.255 scope global ens33
            valid_lft forever preferred_lft forever
        inet6 fe80::8def:aa11:5c16:3f1c%64 scope link
            valid_lft forever preferred_lft forever
        inet6 fe80::7d54:9b08:69fc:d04d%64 scope link tentative dadfailed
            valid_lft forever preferred_lft forever
[root@localhost ~]# _
```

```
[root@localhost ~]# ping www.baidu.com
PING www.wshifen.com (103.235.46.39) 56(84) bytes of data.
64 bytes from 103.235.46.39 (103.235.46.39): icmp_seq=1 ttl=128 time=393 ms
64 bytes from 103.235.46.39 (103.235.46.39): icmp_seq=2 ttl=128 time=365 ms
64 bytes from 103.235.46.39 (103.235.46.39): icmp_seq=3 ttl=128 time=438 ms
64 bytes from 103.235.46.39 (103.235.46.39): icmp_seq=4 ttl=128 time=418 ms
64 bytes from 103.235.46.39 (103.235.46.39): icmp_seq=5 ttl=128 time=384 ms
^C
--- www.wshifen.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4004ms
rtt min/avg/max/mdev = 365.827/400.089/438.186/25.632 ms
[root@localhost ~]# _
```

- hostname 主机名，不要跟源克隆机一致

查看主机名命令

```
hostnamectl
```

更改主机名

```
hostnamectl set-hostname xxx
```

重启机器 **reboot**

2.关闭三台虚拟机防火墙

1.查看防火墙状态

```
systemctl status firewalld
```

2.设置防火墙停用状态

```
systemctl stop firewalld
```

3.设置防火墙功能失效,开机自动关闭

永久关闭

```
systemctl disable firewalld
```

```
[root@linux121 ~]# systemctl status firewalld    查看防火墙状态
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; disabled; vendor preset: enabled)
   Active: inactive (dead)    关闭的状态
     Docs: man:firewalld(1)
[root@linux121 ~]# systemctl stop firewalld    停用防火墙
[root@linux121 ~]# systemctl disable firewalld 使防火墙功能失效, 开机自动关闭
[root@linux121 ~]# _
```

3.三态机器关闭selinux

安全增强型 Linux (Security-Enhanced Linux) 简称 SELinux , 它是一个 Linux 内核模块 , 也是 Linux 的一个安全子系统。

关闭安全策略否则SELinux服务开启后导致SSH连接异常。

修改文件

```
vi /etc/selinux/config
```

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
#       disabled - No SELinux policy is loaded.
#SELINUX=enforcing          注释掉enforcing
#SELINUX=disabled           添加为disabled
# SELINUXTYPE= can take one of three values:
#       targeted - Targeted processes are protected,
#       minimum - Modification of targeted policy. Only selected processes are protected.
#       mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

4.三台机器免密登录

1.ssh协议[安全外壳协议]

SSH 为 **Secure Shell** 的缩写，由 IETF 的网络小组（ Network Working Group ）所制定；SSH 为建立在应用层基础上的安全协议。SSH 是较可靠，专为 **远程登录** 会话和其他网络服务提供安全性的协议。利用 SSH 协议可以有效防止远程管理过程中的信息泄露问题。

SSH 提供两种级别的安全验证：

第一种级别（基于口令的安全验证）

只要你知道自己帐号和口令，就可以登录到远程主机。所有传输的数据都会被加密，但是不能保证你正在连接的服务器就是你想连接的服务器。可能会有别的服务器在冒充真正的服务器，也就是受到“中间人”这种方式的攻击。

第二种级别（基于密匙的安全验证）

需要依靠密匙，也就是你必须为自己创建一对密匙，并把公用密匙放在需要访问的服务器上。如果你要连接到SSH服务器上，客户端软件就会向服务器发出请求，请求用你的密匙进行安全验证。

2.ssh基于密码远程登录

命令	含义
ssh ip地址	登录到远程指定的机器上（需要输入用户名和密码）

```
● 1 centos7-1 ✘ ● 2 centos7-2 ✘ ● 3 centos7-3 ✘ + 
[root@centos7-1 ~]# ssh 192.168.80.101
The authenticity of host '192.168.80.101 (192.168.80.101)' can't be established.
ECDSA key fingerprint is SHA256:VYy4rK1b7DAh5hFtWhGFEs2V2Rmm9AJl7oQV8AKavmg.
ECDSA key fingerprint is MD5:d3:f3:8c:13:4d:63:ed:10:c2:9d:65:31:fc:83:52:9b.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.80.101' (ECDSA) to the list of known hosts.
root@192.168.80.101's password:
Last login: Mon Aug 24 05:56:59 2020 from 192.168.80.1
[root@centos7-2 ~]#
```

可以通过主机名登录

需要配置主机名和ip地址的映射关系

分别在 centos7-1 centos7-2 centos7-3 中配置
vi /etc/hosts

```

1 centos7-1 ✘ 2 centos7-2 ✘ 3 centos7-3 ✘ +
127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4
::1 localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.80.100 centos7-1
192.168.80.101 centos7-2
192.168.80.102 centos7-3
~ ~

```

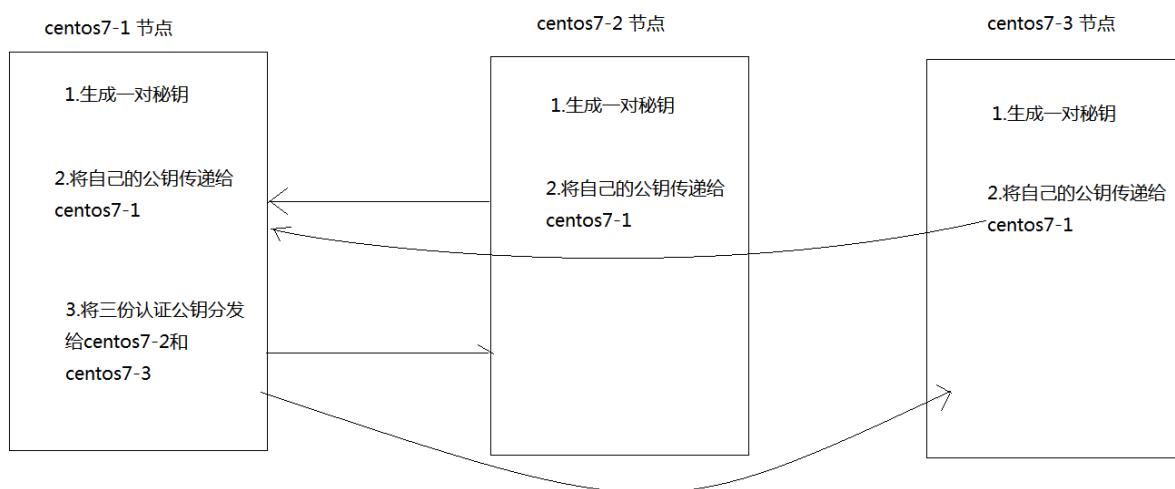
接下来就可以用主机名登录了

```

[root@centos7-2 ~]# ssh centos7-1
root@centos7-1's password:
Last login: Mon Aug 24 08:14:12 2020 from 192.168.80.101
[root@centos7-1 ~]#

```

3.ssh基于秘钥实现免密登录



前提：配置hostname与IP映射 第一步：在三台机器执行以下命令，生成公钥与私钥

第二步：将centos7-2和centos7-3的公钥拷贝到centos7-1

第三步：再将centos7-1的公钥分发给centos7-2和centos7-3

```

第一步: ssh-keygen -t rsa 在centos7-1和centos7-2和centos7-3上面都要执行，产生公钥和私钥
第二步：在centos7-1 ,centos7-2和centos7-3上执行:
          ssh-copy-id centos7-1 将公钥拷贝到centos7-1上面去
第三步：
          scp authorized_keys centos7-2:$PWD
          scp authorized_keys centos7-3:$PWD

```

使用ssh完成免密登录即可（ ssh centos7-2 ）

5.三台机器时钟同步

1.原理

内网的所有服务器都和时钟服务器进行同步时间

1534477026131

2 如何同步

1. 查看本机当前时间

```
date
```

2. 设置本机当前时间

```
date -s "2018-08-17 20:08:09"
```

3. 通过命名和时钟服务器同步时间:

网络计时协议 (NTP) : net time protocol

```
ntpdate us.pool.ntp.org
```

4. 编辑定时任务

```
crontab -e
```

1) 定时任务内容如下

```
*/1 * * * * /usr/sbin/ntpdate us.pool.ntp.org;
```

每隔1分钟执行指令一次

Crontab格式说明					
*	*	*	*	*	command
取值 范围 0-59	取值 范围 0-23	取值 范围 1-31	取值 范围 1-12	取值 范围 0-7	需要执行的命令
					星期几
					月份
					几号
					小时
					分钟

Crontab使用

格式说明

```
# Example of job definition:
# ----- minute (0 - 59)
# | ----- hour (0 - 23)
# | | ----- day of month (1 - 31)
# | | | ----- month (1 - 12) OR jan,feb,mar,apr ...
# | | | | ----- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# | | | |
# * * * * * user-name command to be executed
```

使用实例

实例1：每1分钟执行一次command

命令：`* * * * * command`

实例2：每小时的第3和第15分钟执行

命令：`3,15 * * * * command`

实例3：在上午8点到11点的第3和第15分钟执行

命令：`3,15 8-11 * * * command`

实例4：每隔两天的上午8点到11点的第3和第15分钟执行

命令：`3,15 8-11 */2 * * command`

实例5：每个星期一的上午8点到11点的第3和第15分钟执行

命令：`3,15 8-11 * * 1 command`

Crontab在线生成

<https://cron.qqe2.com/>

第十四章 Shell编程

1 简介

Linux中的shell脚本是一个特殊的应用程序，它介于操作系统和系统内核之间，充当一个命令解释器的角色。负责接收用户输入的操作指令并进行解释，将需要执行的操作传递给内核执行，并输出执行结果。同时它又是一种程序设计语言。作为命令语言，它交互式解释和执行用户输入的命令或者自动地解释和执行预先设定好的一连串的命令；作为程序设计语言，它定义了各种变量和参数，并提供了许多在高级语言中才具有的控制结构，包括循环和分支。Linux 的 Shell 解释器 种类众多，一个系统可以存在多个 shell，可以通过 cat /etc/shells 命令查看系统中安装的 shell 解释器。Bash 由于易用和免费，在日常工作中被广泛使用。同时，Bash 也是大多数Linux 系统默认的 Shell。

2 shell 解释器

java 需要 虚拟机解释器, 同理 shell脚本也需要 解析器

```
[root@node04 shells]# cat /etc/shells
/bin/sh
/bin/bash
/sbin/nologin
/bin/dash
/bin/tcsh
/bin/csh
```

3 快速入门

1 编写脚本

新建 /usr/shell/hello.sh 文件

```
#!/bin/bash
echo 'hello lagou'
```

!是一个约定的标记，它告诉系统这个脚本需要什么解释器来执行，即使用哪一种 Shell。

echo 命令用于向窗口输出文本。

2 执行shell脚本

执行方式1

```
[root@centos7-1 shell]# /bin/sh hello.sh  
hello lagou  
[root@centos7-1 shell]# /bin/bash hello.sh  
hello lagou
```

```
[root@centos7-1 shell]# vim hello.sh  
[root@centos7-1 shell]# /bin/sh hello.sh  
hello lagou  
[root@centos7-1 shell]# /bin/bash hello.sh  
hello lagou  
[root@centos7-1 shell]#
```

问题: bash 和 sh 是什么关系?

sh 是 bash 的 快捷方式

执行方式2

```
[root@node04 shells]# bash hello.sh  
hello world  
[root@node04 shells]# sh hello.sh  
hello world
```

1.查看环境变量path:

echo \$PATH

2.添加执行权限

chmod +x 文件名称

4. 变量

在shell脚本中, 定义变量时, 变量名不加美元符号 (\$) , 如 : `username="tom"`

注意: 变量名和等号之间不能有空格, 这可能和你熟悉的所有编程语言都不一样。同时, 变量名的命名须遵循如下规则:

- 命名只能使用英文字母，数字和下划线，首个字符不能以数字开头。
- 中间不能有空格，可以使用下划线（_）。
- 不能使用标点符号。
- 不能使用bash里的关键字（可用help命令查看保留关键字）。

有效的 Shell 变量名示例如下：

```
JACK  
LD_LIBRARY_PATH  
_demo  
var2
```

无效的变量命名：

```
?var=123  
user*name=runoob
```

使用语句给变量赋值

举例：

```
for file in `ls /etc`
```

或

```
for file in $(ls /etc)
```

以上语句是将/etc下目录的文件名循环出来

使用变量

使用一个定义过的变量，只要在变量名前面加美元符号即可，如：

```
your_name="tom"  
echo $your_name  
echo ${your_name}
```

变量名外面加花括号是可选的，添加花括号是帮助解释器识别变量边界

举例：

```
habby="i like ${course}script"  
habby="i like $coursescript"
```

只读变量

使用readonly命令可以将变量定义为只读变量，只读变量的值不能被改变

```
readonly 变量名
```

举例：

```
username=tom
readonly username
username=jack 报错 /bin/sh: NAME: This variable is read only.
```

删除变量

使用unset命令可以删除变量

```
unset variable_name
```

注意:unset命令不能删除只读变量

5.字符串

字符串是shell编程中最常用最有用的数据类型（除了数字和字符串，也没啥其它类型好用了），字符串可以用单引号，也可以用双引号，也可以不用引号。

单引号

```
skill='java'
str='I am goot at $skill'
echo $str
```

输出结果为:

```
I am goot at $skill
```

单引号字符串的限制：

- 单引号里的任何字符都会原样输出，单引号字符串中的变量是无效的；
- 单引号字符串中不能出现单独一个的单引号（对单引号使用转义符后也不行）

双引号

```
skill='java'
str="I am goot at $skill"
echo $str
```

输出结果为：

```
I am goot at java
```

双引号的优点：

- 双引号里可以有变量
- 双引号里可以出现转义字符

获取字符串长度

```
skill='java'

echo ${skill} # 输出结果: java

echo ${#skill} # 输出结果: 4
```

提取子字符串

一下实例从字符串第2个字符开始截取4个字符

```
str="i like hadoop"

echo ${str:2} #substring(2)

echo ${str:2:2} #substring(2,2)
```

查找子字符串

查找字符o在那个位置(最先出现的字符)

```
str="hadoop is so easy"
echo `expr index "$str" o`
```

找的时候是从1开始查找

6.运算符

Shell 和其他编程一样，支持包括：算术、关系、布尔、字符串等运算符。原生 bash 不支持简单的数学运算，但是可以通过其他命令来实现，例如expr。expr 是一款表达式计算工具，使用它能完成表达式的求值操作。例如，两个数相加

```
val=`expr 1 + 2`
echo $val
```

注意：

表达式和运算符之间要有空格，例如 `1+2` 是不对的，必须写成 `1 + 2`。

完整的表达式要被 ` 包含，注意不是单引号，在 Esc 键下边。

下表列出了常用的算术运算符，假定变量 `a` 为 `10`，变量 `b` 为 `20`：

1) 算数运算符

运算符	说明	举例
<code>+</code>	加法	<code>expr \$a + \$b</code> 结果为 30。
<code>-</code>	减法	<code>expr \$a - \$b</code> 结果为 -10。
<code>*</code>	乘法	<code>expr \$a * \$b</code> 结果为 200。
<code>/</code>	除法	<code>expr \$b / \$a</code> 结果为 2。
<code>%</code>	取余	<code>expr \$b % \$a</code> 结果为 0。
<code>=</code>	赋值	<code>a=\$b</code> 将把变量 b 的值赋给 a。
<code>==</code>	相等。用于比较两个数字，相同则返回 true。	<code>[\$a == \$b]</code> 返回 false。
<code>!=</code>	不相等。用于比较两个数字，不相同则返回 true。	<code>[\$a != \$b]</code> 返回 true。

注意：条件表达式要放在方括号之间，并且要有空格，例如：`[$a==$b]` 是错误的，

必须写成 `[$a == $b]`。

案例

```
#!/bin/bash
a=4
b=20

#加法运算
each expr $a + $b

#减法运算
echo expr $a - $b

#乘法运算，注意*号前面需要反斜杠
echo expr $a \* $b

#除法运算
echo $a / $b
此外，还可以通过(( ))、$(())、$[]进行算术运算。
((a++))
echo "a = $a"
c=$((a + b))
d=$((a + b))
```

```
echo "c = $c"
echo "d = $d"
```

2)关系运算符

关系运算符只支持数字，不支持字符串，除非字符串的值是数字。下表列出了常用的关系运算符，假定变量 a 为 10，变量 b 为 20：

运算符	说明	英文	举例
-eq	检测两个数是否相等，相等返回 true。	equal	[\$a -eq \$b] 返回 false。
-ne	检测两个数是否不相等，不相等返回 true。	not equal	[\$a -ne \$b] 返回 true。
-gt	检测左边的数是否大于右边的，如果是，则返回 true。	greater than	[\$a -gt \$b] 返回 false。
-lt	检测左边的数是否小于右边的，如果是，则返回 true。	less than	[\$a -lt \$b] 返回 true。
-ge	检测左边的数是否大于等于右边的，如果是，则返回 true。	Greater than or equal to	[\$a -ge \$b] 返回 false。
-le	检测左边的数是否小于等于右边的，如果是，则返回 true。	Less than or equal to	[\$a -le \$b] 返回 true。

7.流程控制

1)if语句

- if语句格式

```
if condition
then
    command1
    command2
    ...
    commandN
fi
```

- if.else语句

```
if condition
then
    command1
    command2
    ...
    commandN
else
    command
fi
```

- if..elif..else

```
if condition1
then
    command1
elif condition2
then
    command2
else
    commandN
fi
```

案例

```
[root@hadoop01 export]# cat if_test.sh
#!/bin/bash
a=20
b=10
# 需求1：判断 a 是否 100
if [ $a > 100 ]; then
echo "$a 大于 100"
fi

# 需求2：判断 a 是否等于 b
if [ $a -eq $b ]; then
echo "$a 等于 $b"
else
echo "$a 不等于 $b"
fi

# 需求3：判断 a 与 b 比较
if [ $a -lt $b ]; then
echo "$a 小于 $b"
elif [ $a -eq $b ]; then
echo "$a 等于 $b"
else
echo "$a 大于 $b"
fi

# 需求4：判断 (a + 10) 和 (b * b) 比较小
if test $[ a + 10 ] -gt $[ b * b ]; then
```

```
echo "(a+10) 大于 (b * b)"
else
echo "(a+10) 小于或等于 (b*b)"
fi
```

2)for 循环

格式

```
for variable in (list); do
command
command
...
done
```

练习

```
# 需求1：遍历 1~5
# 需求2：遍历 1~100
# 需求3：遍历 1~100之间的奇数
# 需求4：遍历 根目录 下的内容
```

```
[root@hadoop01 export]# cat for_test.sh
#!/bin/bash

# 需求1：遍历 1~5
for i in 1 2 3 4 5; do
echo $i;
done

# 需求2：遍历 1~100
for i in {1..100}; do
echo $i
done

# 需求3：遍历 1~100之间的奇数
for i in {1..100..2}; do
echo $i
done

# 需求4：遍历 根目录 下的内容
for f in `ls /`; do
echo $f
done
```

3) while 语句

while循环用于不断执行一系列命令，也用于从输入文件中读取数据；命令通常为测试条件。其格式为

```
while condition; do
    command
done
```

需求：计算 1~100 的和

```
#!/bin/bash

sum=0
i=1
while [ $i -le 100 ]; do
    sum=$((sum + i))
    i=$((i + 1))
done
echo $sum
```

运行脚本，输出：

```
5050
```

4) 无限循环

```
for ((;))
do
    command
done

while true
do
    command
done
```

8 case

Shell case语句为多选择语句。可以用case语句匹配一个值与一个模式，如果匹配成功，执行相匹配的命令。case语句格式如下：

```
case 值 in
    模式1)
        command1
        command2
        ...
        commandN
        ;;
    模式2 )
        command1
        command2
        ...
        commandN
        ;;
esac
```

case工作方式如上所示。取值后面必须为单词in，每一模式必须以右括号结束。取值可以为变量或常数。匹配发现取值符合某一模式后，其间所有命令开始执行直至;;。取值将检测匹配的每一个模式。一旦模式匹配，则执行完匹配模式相应命令后不再继续其他模式。如果无一匹配模式，使用星号*捕获该值，再执行后面的命令。下面的脚本提示输入1到4，与每一种模式进行匹配：

```
echo '输入 1 到 4 之间的数字：'
read aNum
case $aNum in
    1) echo '你选择了 1'
    ;;
    2) echo '你选择了 2'
    ;;
    3) echo '你选择了 3'
    ;;
    4) echo '你选择了 4'
    ;;
    *) echo '你没有输入 1 到 4 之间的数字'
    ;;
esac
```

输入不同的内容，会有不同的结果，例如：

```
输入 1 到 4 之间的数字：
你输入的数字为：
3
你选择了 3
```

9 跳出循环

在循环过程中，有时候需要在未达到循环结束条件时强制跳出循环，Shell使用两个命令来实现该功能：break和continue。

1) break命令

break命令允许跳出所有循环（终止执行后面的所有循环）。

需求：执行死循环 每隔1秒打印当前时间，执行10次停止

```
#!/bin/bash
# 需求：执行死循环 每隔1秒打印当前时间，执行10次停止
i=0;
while true; do
    sleep 1
    echo $i `date +"%Y-%m-%d %H:%M:%S"`
    i=$((i + 1))
    if [ $i -eq 10 ]; then
        break
    fi
done
```

2) continue

continue命令与break命令类似，只有一点差别，它不会跳出所有循环，仅仅跳出当前循环。

需求：打印 1~30，注意 跳过3的倍数

```
#!/bin/bash
# 需求：打印 1~30，注意 跳过3的倍数
for i in {1..30}; do
    if test $[ i % 3 ] -eq 0; then
        continue
    fi
    echo $i
done
```

10. 函数使用

1) 函数的快速入门

格式

```
[ function ] funname()
{
    action;
    [return int;]
}
```

- 可以带function fun() 定义，也可以直接fun() 定义,不带任何参数。
- 参数返回，可以显示加：return 返回，如果不加，将以最后一条命令运行结果，作为返回值。 return后跟数值n(0-255)

快速入门

```
#!/bin/bash
demoFun(){
    echo "这是我的第一个 shell 函数!"
}
echo "-----函数开始执行-----"
demoFun
echo "-----函数执行完毕-----"
```

2) 传递参数给函数

在Shell中，调用函数时可以向其传递参数。在函数体内部，通过 n 的形式来获取参数的值，例如， 1 表示第一个参数， $$2$ 表示第二个参数... 带参数的函数示例：

```
#!/bin/bash
funWithParam(){
    echo "第一个参数为 $1 !"
    echo "第二个参数为 $2 !"
    echo "第十个参数为 ${10} !"
    echo "第十个参数为 ${10} !"
    echo "第十一一个参数为 ${11} !"
    echo "参数总数有 $# 个!"
    echo "作为一个字符串输出所有参数 $* !"
}
funWithParam 1 2 3 4 5 6 7 8 9 34 73
```

输出结果：

```
第一个参数为 1 !
第二个参数为 2 !
第十个参数为 10 !
第十个参数为 34 !
第十一一个参数为 73 !
参数总数有 11 个!
作为一个字符串输出所有参数 1 2 3 4 5 6 7 8 9 34 73 !
```