



尚硅谷 Linux 课程

尚硅谷 Linux 课程(大数据、JavaEE, Python 通用版)

尚硅谷





第 1 章 LINUX 开山篇.....	1
1.1 本套 LINUX 课程的内容介绍.....	1
1.2 LINUX 的学习方向.....	1
1.2.1 Linux 运维工程师.....	1
1.2.2 Linux 嵌入式开发工程师.....	2
1.2.3 在 linux 下做各种程序开发.....	2
1.2.4 示意图.....	2
1.3 LINUX 的应用领域.....	2
1.3.1 个人桌面应用领域.....	2
1.3.2 服务器应用领域.....	3
1.3.3 嵌入式应用领域.....	3
1.4 学习 LINUX 的阶段（高手进阶过程）.....	3
1.5 LINUX 的学习方法和建议.....	4
第 2 章 基础篇 LINUX 入门.....	5
2.1 LINUX 介绍.....	5
2.2 UNIX 是怎么来的.....	6
2.3 LINUX 是怎么来的.....	7
2.4 LINUX 和 UNIX 关系一览图.....	8
2.5 LINUX 和 WINDOWS 比较.....	9
第 3 章 基础篇 VM 和 LINUX 系统(CENTOS)安装.....	11
3.1 安装 VM 和 CENTOS.....	11
3.2 VM 软件和 CENTOS 的安装软件.....	11
3.3 VM 安装的步骤.....	12
3.4 CENTOS 安装的步骤.....	12
3.5 CENTOS 的终端使用和联网.....	12
3.6 VMTOOLS 安装.....	13
3.6.1 介绍.....	13
3.7 VMTOOLS 的安装和使用.....	14
3.7.1 安装 vmtools 的步骤说明.....	14
3.7.2 使用 vmtools 来设置 windows 和 linux 的共享文件夹.....	14
3.7.3 安装 vmtools 的课堂练习.....	15
第 4 章 基础篇 LINUX 的目录结构.....	16
4.1 基本介绍.....	16
4.2 目录结构的具体介绍.....	16



4.3 LINUX 目录总结一下.....	18
第 5 章 实操篇 远程登录 LINUX 系统.....	20
5.1 为什么需要远程登录 LINUX.....	20
5.1.1 示意图.....	20
5.1.2 说明.....	20
5.2 远程登录 LINUX-XSHELL5.....	20
5.3 安装 XSHELL5 并使用.....	21
5.3.1 安装过程.....	21
5.3.2 XShell5 的关键配置.....	21
5.3.3 XShell5 远程登录到 Linux 后，就可以使用指令来操作 Linux 系统.....	21
5.4 远程上传下载文件 XFTP5.....	22
5.4.1 XFtp5 软件介绍.....	22
5.4.2 XFtp5 软件的安装.....	22
5.4.3 Xftp5 的配置和使用.....	22
5.4.4 如何解决 XFTP5 中文乱码的问题.....	24
5.5 XFTP5 和 XSHELL5 的使用.....	25
第 6 章 实操篇 VI 和 VIM 编辑器.....	26
6.1 vi 和 vim 的基本介绍.....	26
6.2 vi 和 vim 的三种常见模式.....	26
6.2.1 正常模式.....	26
6.2.2 插入模式/编辑模式.....	26
6.2.3 命令行模式.....	26
6.3 快速入门案例.....	26
6.4 vi 和 vim 三种模式的相互转化图.....	27
6.5 快捷键的使用案例.....	28
6.6 vim 和 vi 的快捷键键盘一览图.....	29
6.7 vi 和 vim 课堂练习.....	29
第 7 章 实操篇 开机、重启和用户登录注销.....	30
7.1 关机&重启命令.....	30
7.1.1 基本介绍.....	30
7.1.2 注意细节.....	30
7.2 用户登录和注销.....	30
7.2.1 基本介绍.....	30
7.2.2 使用细节.....	31
第 8 章 实操篇 用户管理.....	32
8.1 基本介绍.....	32



8.2 添加用户.....	32
8.2.1 基本语法.....	32
8.2.2 实际案例.....	32
8.2.3 细节说明.....	33
8.3 给用户指定或者修改密码.....	33
8.4 删 除 用户.....	34
8.4.1 基本语法.....	34
8.4.2 应用案例.....	34
8.4.3 思考题.....	34
8.5 查询用户信息.....	34
8.5.1 基本语法.....	34
8.5.2 应用实例.....	35
8.5.3 细节说明.....	35
8.6 切换用户.....	35
8.6.1 介绍.....	35
8.6.2 基本语法.....	35
8.6.3 应用实例.....	35
8.6.4 细节说明.....	36
8.7 用户组.....	36
8.7.1 介绍.....	36
8.7.2 增加组.....	36
8.7.3 案例演示.....	36
8.7.4 删 除 组.....	36
8.7.5 案例演示.....	36
8.8 增加用户时直接加上组.....	37
8.8.1 指令(基本语法).....	37
8.8.2 案例演示.....	37
8.9 修改用户的组.....	37
8.9.1 指令(基本语法).....	37
8.9.2 案例演示.....	37
8.10 /ETC/PASSWD 文件.....	38
8.11 /ETC/SHADOW 文件.....	38
8.12 /ETC/GROUP 文件.....	38
第 9 章 实操篇 实用指令.....	39
9.1 指定运行级别.....	39
9.2 切换到指定运行级别的指令.....	40
9.2.1 基本语法.....	40
9.2.2 应用实例.....	40



9.2.3 面试题.....	40
9.2.4 课堂练习:.....	41
9.3 帮助指令.....	41
9.3.1 介绍.....	41
9.3.2 man 获得帮助信息.....	41
9.3.3 help 指令.....	41
9.3.4 当一个指令不熟悉如何学习的建议.....	42
9.4 文件目录类.....	43
9.4.1 pwd 指令.....	43
9.4.2 ls 指令.....	43
9.4.3 cd 指令.....	44
9.4.4 mkdir 指令.....	45
9.4.5 rmdir 指令.....	46
9.4.6 touch 指令.....	47
9.4.7 cp 指令[重要].....	47
9.4.8 rm 指令.....	49
9.4.9 mv 指令.....	50
9.4.10 cat 指令.....	50
9.4.11 more 指令.....	51
9.4.12 less 指令.....	52
9.4.13 > 指令 和 >> 指令.....	53
9.4.14 echo 指令.....	55
9.4.15 head 指令.....	55
9.4.16 tail 指令.....	56
9.4.17 ln 指令.....	57
9.4.18 history 指令.....	58
9.5 时间日期类.....	59
9.5.1 date 指令-显示当前日期.....	59
9.5.2 date 指令-设置日期.....	60
9.5.3 cal 指令.....	60
9.6 搜索查找类.....	61
9.6.1 find 指令.....	61
9.6.2 locate 指令.....	63
9.6.3 grep 指令和 管道符号 	64
9.7 压缩和解压类.....	64
9.7.1 gzip/gunzip 指令.....	64
9.7.2 zip/unzip 指令.....	65
9.7.3 tar 指令.....	66
第 10 章 实操篇 组管理和权限管理.....	69



10.1 LINUX 组基本介绍.....	69
10.2 文件/目录 所有者.....	69
10.2.1 查看文件的所有者.....	69
10.2.2 修改文件所有者.....	70
10.3 组的创建.....	71
10.3.1 基本指令.....	71
10.3.2 应用实例.....	71
10.4 文件/目录 所在组.....	71
10.4.1 查看文件/目录所在组.....	71
10.4.2 修改文件所在的组.....	72
10.5 其它组.....	72
10.6 改变用户所在组.....	72
10.6.1 改变用户所在组.....	73
10.6.2 应用实例.....	73
10.7 权限的基本介绍.....	73
10.8 RWX 权限详解.....	74
10.8.1 rwx 作用到文件.....	74
10.8.2 rwx 作用到目录.....	74
10.9 文件及目录权限实际案例.....	74
10.10 修改权限-CHMOD.....	75
10.10.1 基本说明:	75
10.10.2 第一种方式: + 、 - 、 = 变更权限.....	75
10.10.3 第二种方式: 通过数字变更权限.....	76
10.11 修改文件所有者-CHOWN.....	77
10.11.1 基本介绍.....	77
10.11.2 案例演示:	77
10.12 修改文件所在组-CHGRP.....	78
10.12.1 基本介绍.....	78
10.12.2 案例演示:	78
10.13 最佳实践-警察和土匪游戏.....	78
10.14 课后练习.....	80
10.15 课后练习题 2.....	80
第 11 章 实操篇 CROND 任务调度.....	82
11.1 原理示意图.....	82
11.2 概述.....	82
11.3 基本语法.....	83
11.3.1 常用选项.....	83
11.4 快速入门.....	83



11.4.1 任务的要求.....	83
11.4.2 步骤如下.....	83
11.4.3 参数细节说明.....	83
11.5 任务调度的几个应用实例.....	84
11.5.1 案例 1：每隔 1 分钟，就将当前的日期信息，追加到 /tmp/mydate 文件中.....	85
11.5.2 案例 2：每隔 1 分钟， 将当前日期和日历都追加到 /home/mycal 文件中.....	85
11.5.3 案例 3： 每天凌晨 2:00 将 mysql 数据库 testdb ， 备份到文件中 mydb.bak。	85
11.6 CROND 相关指令:.....	86
第 12 章 实操篇 LINUX 磁盘分区、挂载.....	87
12.1 分区基础知识.....	87
12.1.1 分区的方式:	87
12.1.2 windows 下的磁盘分区.....	87
12.2 LINUX 分区.....	88
12.2.1 原理介绍.....	88
12.2.2 硬盘说明.....	89
12.2.3 使用 lsblk 指令查看当前系统的分区情况.....	89
12.3 挂载的经典案例.....	90
12.3.1 如何增加一块硬盘.....	90
12.4 具体的操作步骤整理.....	91
12.4.1 虚拟机增加硬盘步骤 1.....	91
12.4.2 虚拟机增加硬盘步骤 2.....	91
12.4.3 虚拟机增加硬盘步骤 3.....	92
12.4.4 虚拟机增加硬盘步骤 4.....	92
12.4.5 虚拟机增加硬盘步骤 5.....	92
12.5 磁盘情况查询.....	93
12.5.1 查询系统整体磁盘使用情况.....	93
12.5.2 查询指定目录的磁盘占用情况.....	93
12.6 磁盘情况-工作实用指令.....	94
第 13 章 实操篇 网络配置.....	96
13.1 LINUX 网络配置原理图(含虚拟机).....	96
13.2 查看网络 IP 和网关.....	96
13.2.1 查看虚拟网络编辑器.....	96
13.2.2 修改 ip 地址(修改虚拟网络的 ip).....	96
13.2.3 查看网关.....	97
13.2.4 查看 windows 环境的中 VMnet8 网络配置 (ipconfig 指令).....	97
13.3 PING 测试主机之间网络连通.....	98
13.3.1 基本语法.....	98



13.3.2 应用实例.....	98
13.4 LINUX 网络环境配置.....	98
13.4.1 第一种方法(自动获取).....	98
13.4.2 第二种方法(指定固定的 ip).....	99
第 14 章 实操篇 进程管理.....	101
14.1 进程的基本介绍.....	101
14.2 显示系统执行的进程.....	101
14.2.1 说明:	101
14.2.2 ps 指令详解.....	102
14.2.3 应用实例.....	103
14.3 终止进程 KILL 和 KILLALL.....	104
14.3.1 介绍:.....	104
14.3.2 基本语法:	104
14.3.3 常用选项:	104
14.3.4 最佳实践:	104
14.4 查看进程树 PSTREE.....	106
14.4.1 基本语法:	106
14.4.2 常用选项:	106
14.4.3 应用实例:	106
14.5 服务(SERVICE)管理.....	106
14.5.1 介绍:.....	107
14.5.2 service 管理指令:	107
14.5.3 使用案例:	107
14.5.4 细节讨论:	108
14.5.5 查看服务名:.....	108
14.5.6 服务的运行级别(runlevel):.....	109
14.5.7 开机的流程说明.....	110
14.5.8 chkconfig 指令.....	110
介绍.....	110
通过 chkconfig 命令可以给每个服务的各个运行级别设置自启动/关闭.....	110
基本语法.....	110
1) 查看服务 chkconfig --list grep xxx.....	110
2) chkconfig 服务名 --list.....	110
3) chkconfig --level 5 服务名 on/off.....	110
14.5.9 应用实例:	111
14.5.10 • 使用细节.....	111
14.6 动态监控进程.....	111
14.6.1 介绍:	111



14.6.2 基本语法:	112
14.6.3 选项说明:	112
14.6.4 应用实例:	112
14.6.5 查看系统网络情况 netstat(重要).....	114
第 15 章 实操篇 RPM 和 YUM.....	115
15.1 RPM 包的管理.....	115
15.1.1 介绍:	115
15.1.2 rpm 包的简单查询指令:	115
15.1.3 rpm 包名基本格式:	115
15.1.4 rpm 包的其它查询指令:	116
15.1.5 卸载 rpm 包:	118
15.1.6 安装 rpm 包:	118
15.2 YUM.....	119
15.2.1 介绍:	119
15.2.2 yum 的基本指令.....	120
15.2.3 yum 应用实例:	120
第 16 章 JAVAEE 定制篇 搭建 JAVAEE 环境.....	122
16.1 概述.....	122
16.1.1 示意图:	122
16.2 安装 JDK.....	122
16.2.1 看老师演示，一会整理笔记:	122
16.2.2 安装步骤.....	122
16.2.3 测试是否安装成功.....	123
16.3 安装 TOMCAT.....	124
16.3.1 步骤	124
16.3.2 测试是否安装成功:	125
16.4 ECLIPSE 的安装.....	126
16.4.1 步骤	126
16.5 MYSQL 的安装和配置.....	127
16.5.1 安装的步骤和文档.....	127
16.5.2 说明.....	127
第 17 章 大数据定制篇 SHELL 编程.....	128
17.1 为什么要学习 SHELL 编程.....	128
17.2 SHELL 是什么.....	128
17.3 SHELL 编程快速入门-SHELL 脚本的执行方式.....	128
17.3.1 脚本格式要求.....	128
17.3.2 编写第一个 Shell 脚本.....	129



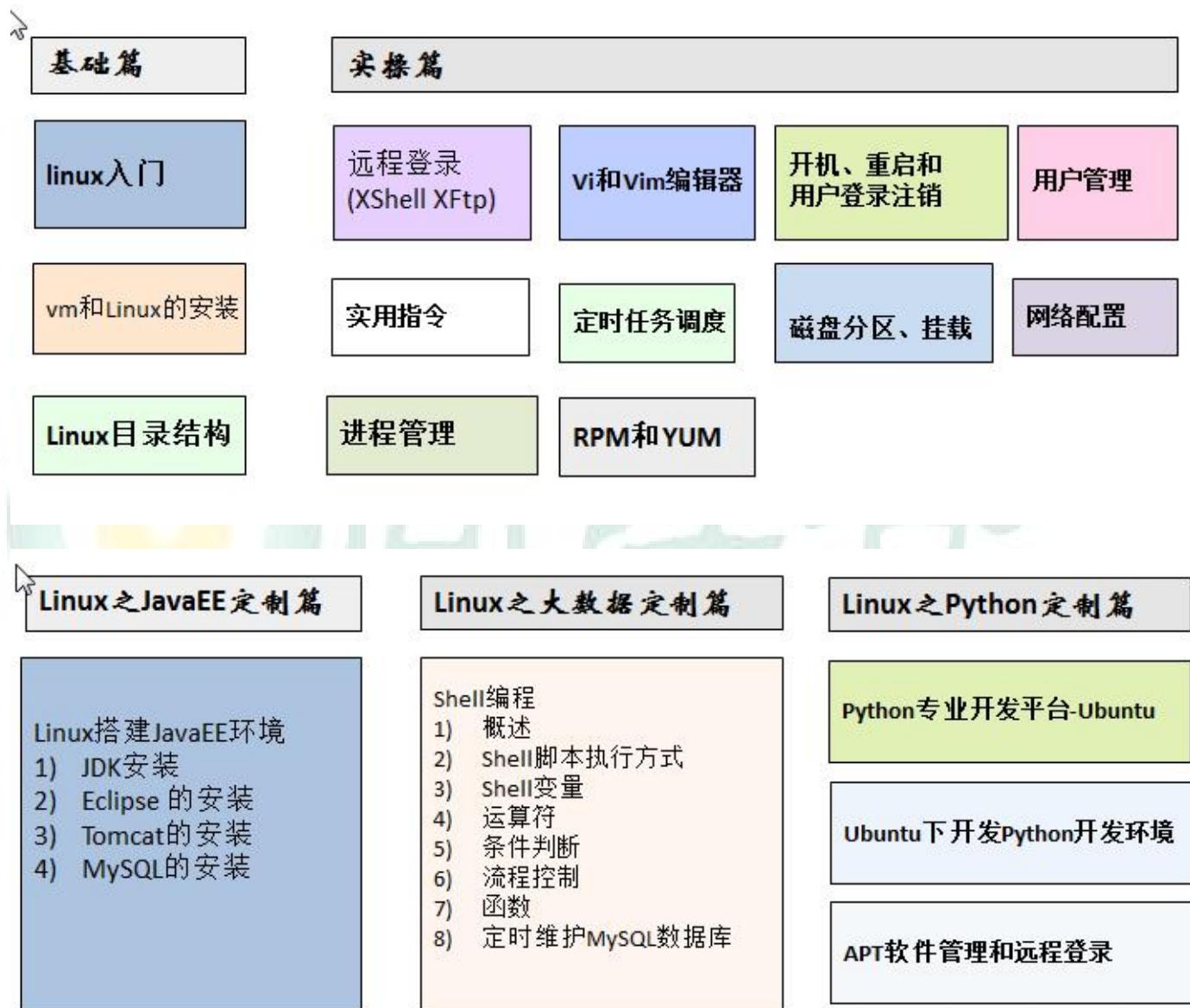
17.3.3 脚本的常用执行方式.....	129
17.4 SHELL 的变量.....	130
17.4.1 Shell 的变量的介绍.....	130
17.4.2 shell 变量的定义.....	130
17.4.3 •定义变量的规则.....	131
17.4.4 •将命令的返回值赋给变量（重点）.....	131
17.5 设置环境变量.....	132
17.5.1 基本语法.....	132
17.5.2 快速入门.....	132
17.6 位置参数变量.....	134
17.6.1 介绍.....	134
17.6.2 基本语法.....	134
17.6.3 位置参数变量应用实例.....	134
17.7 预定义变量.....	135
17.7.1 基本介绍.....	135
17.7.2 基本语法.....	135
17.7.3 应用实例.....	135
17.8 运算符.....	135
17.8.1 基本介绍.....	135
17.8.2 基本语法.....	135
17.9 条件判断.....	137
17.9.1 •基本语法.....	137
17.9.2 •应用实例.....	137
17.9.3 •常用判断条件.....	137
17.9.4 应用实例.....	138
17.10 流程控制.....	139
17.10.1 if 判断.....	139
17.10.2 case 语句.....	140
17.10.3 for 循环.....	141
17.10.4 while 循环.....	143
17.11 READ 读取控制台输入.....	143
17.11.1 基本语法.....	143
17.11.2 应用实例.....	144
17.12 函数.....	144
17.12.1 函数介绍.....	144
17.12.2 系统函数.....	144
17.12.3 •应用实例.....	145
17.12.4 自定义函数.....	145
17.13 SHELL 编程综合案例.....	146



第 18 章 PYTHON 定制篇 开发平台 UBUNTU.....	149
18.1 UBUNTU 的介绍.....	149
18.2 UBUNTU 的安装.....	149
18.2.1 安装的步骤.....	149
18.2.2 设置 Ubuntu 支持中文.....	149
18.3 UBUNTU 的 ROO 用户.....	150
18.3.1 介绍.....	151
18.3.2 给 root 用户设置密码并使用.....	151
18.4 UBUNTU 下开发 PYTHON.....	151
18.4.1 说明.....	151
18.4.2 在 Ubuntu 下开发一个 Python 程序.....	152
第 19 章 PYTHON 定制篇 APT 软件管理和远程登录.....	153
19.1 APT 介绍.....	153
19.2 UBUNTU 软件操作的相关命令.....	153
19.3 更新 UBUNTU 软件下载地址.....	154
19.3.1 原理示意图.....	154
19.3.2 寻找国内镜像源.....	154
19.3.3 备份 Ubuntu 默认的源地址.....	156
19.3.4 更新源服务器列表.....	157
19.4 UBUNTU 软件安装, 卸载的最佳实践.....	157
19.4.1 案例说明: 使用 apt 完成安装和卸载 vim 软件, 并查询 vim 软件的信息:	157
19.5 使用 SSH 远程登录 UBUNTU.....	159
19.5.1 ssh 介绍.....	159
19.5.2 原理示意图:	159
19.6 使用 SSH 远程登录 UBUNTU.....	160
19.6.1 安装 SSH 和启用.....	160
19.6.2 在 Windows 使用 XShell5/XFTP5 登录 Ubuntu.....	160
19.6.3 从 linux 系统客户机远程登陆 linux 系统服务机.....	161

第 1 章 Linux 开山篇

1.1 本套 Linux 课程的内容介绍



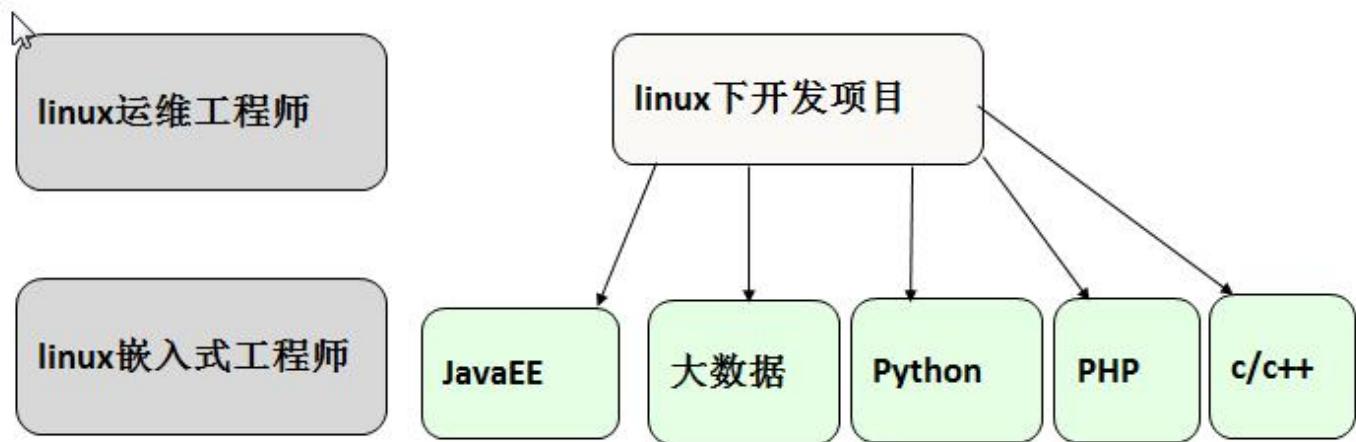
1.2 Linux 的学习方向

1.2.1 Linux 运维工程师

1.2.2 Linux 嵌入式开发工程师

1.2.3 在 linux 下做各种程序开发

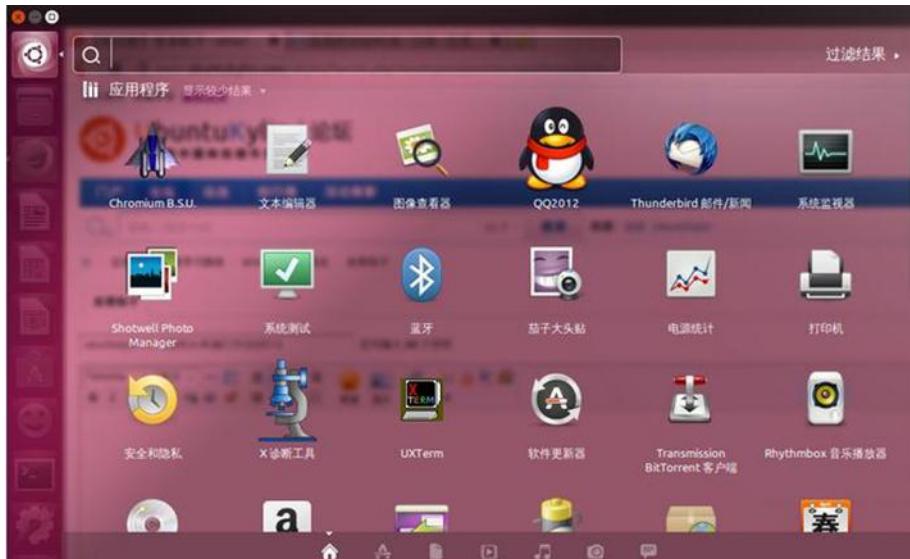
1.2.4 示意图



1.3 Linux 的应用领域

1.3.1 个人桌面应用领域

此领域是传统 linux 应用最薄弱的环节，传统 linux 由于界面简单、操作复杂、应用软件少的缺点，一直被 windows 所压制，但近些年来随着 ubuntu、fedora [fi'dɔ:rə] 等优秀桌面环境的兴起，同时各大硬件厂商对其支持的加大，linux 在个人桌面领域的占有率在逐渐的提高。



1.3.2 服务器应用领域

linux 在服务器领域的应用是最强的。

linux 免费、稳定、高效等特点在这里得到了很好的体现，近些年来 linux 服务器市场得到了飞速的提升，尤其在一些高端领域尤为广泛。

1.3.3 嵌入式应用领域

近些年来 linux 在嵌入式领域的应用得到了飞速的提高

linux 运行稳定、对网络的良好支持性、低成本，且可以根据需要进行软件裁剪，内核最小可以达到几百 KB 等特点，使其近些年来在嵌入式领域的应用得到非常大的提高

主要应用：机顶盒、数字电视、网络电话、程控交换机、手机、PDA、智能家居、智能硬件等都是其应用领域。以后再物联网中应用会更加广泛

1.4 学习 Linux 的阶段（高手进阶过程）

linux 是一个开源、免费的操作系统，其稳定性、安全性、处理多并发已经得到



业界的认可，目前很多中型，大型甚至是集群项目都在使用 linux,很多软件公司考虑到开发成本都首选 linux,在中国软件公司得到广泛的使用。

我个人认为学习 linux 流程为：

第 1 阶段：linux 环境下的基本操作命令，包括 文件操作命令(rm mkdir chmod, chown) 编辑工具使用 (vi vim) linux 用户管理(useradd userdel usermod)等

第 2 阶段：linux 的各种配置（环境变量配置，网络配置，服务配置）

第 3 阶段：linux 下如何搭建对应语言的开发环境（大数据，JavaEE, Python 等）

第 4 阶段：能编写 shell 脚本，对 Linux 服务器进行维护。

第 5 阶段：能进行安全设置，防止攻击，保障服务器正常运行，能对系统调优。

第 6 阶段：深入理解 Linux 系统（对内核有研究），熟练掌握大型网站应用架构组成、并熟悉各个环节的部署和维护方法。

1.5 Linux 的学习方法和建议

- 1) 高效而愉快的学习
- 2) 先建立一个整体框架，然后细节
- 3) 不需要掌握所有的 Linux 指令，要学会查询手册和百度
- 4) 先 know how ,再 know why
- 5) 计算机是一门”做中学”的学科 ,不是会了再做，而是做了才会.
- 6) 适当的囫囵吞枣
- 7) Linux 不是编程，重点是实际操作，各种常用指令要玩的溜

第 2 章 基础篇 Linux 入门

2.1 Linux 介绍

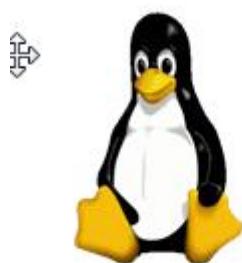
- 1) Linux 怎么读 【里纽克斯，利尼克斯，里纳克斯】
- 2) Linux 是一款操作系统，免费，开源，安全，高效，稳定，处理高并发非常强悍，现在很多企业级的项目都部署到 Linux/unix 服务器运行。
- 3) Linux 创始人-linus 林纳斯



Linus Torvalds

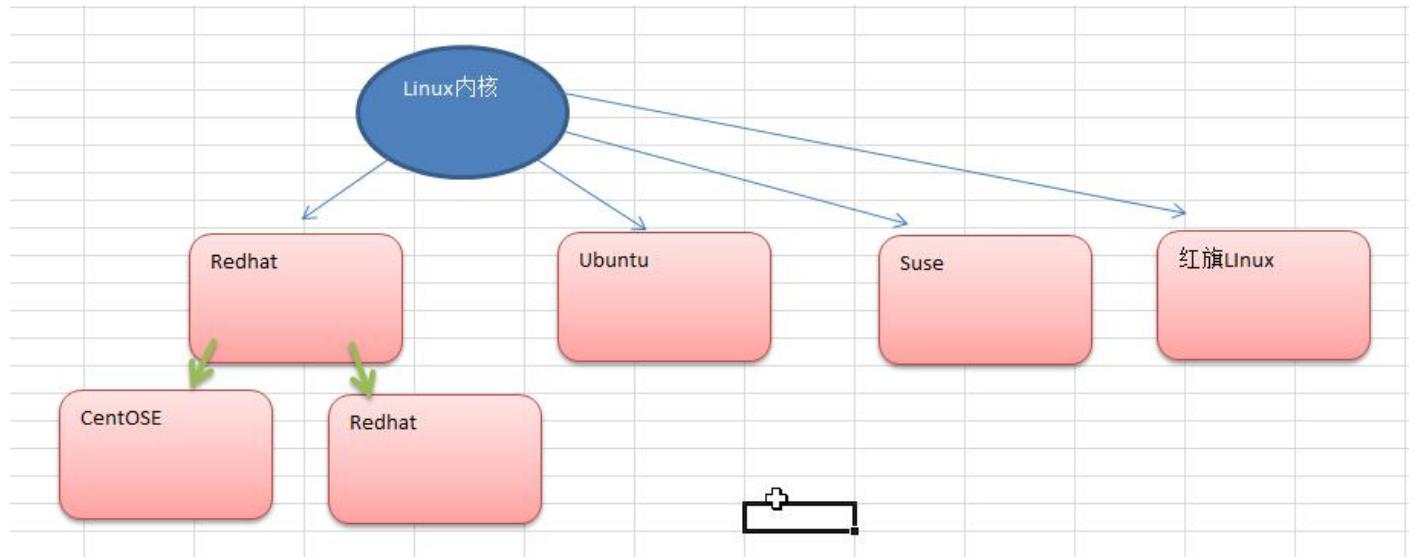
- 4) Linux 的吉祥物

企鹅 tux



Tux

- 5) Linux 的主要的发行版



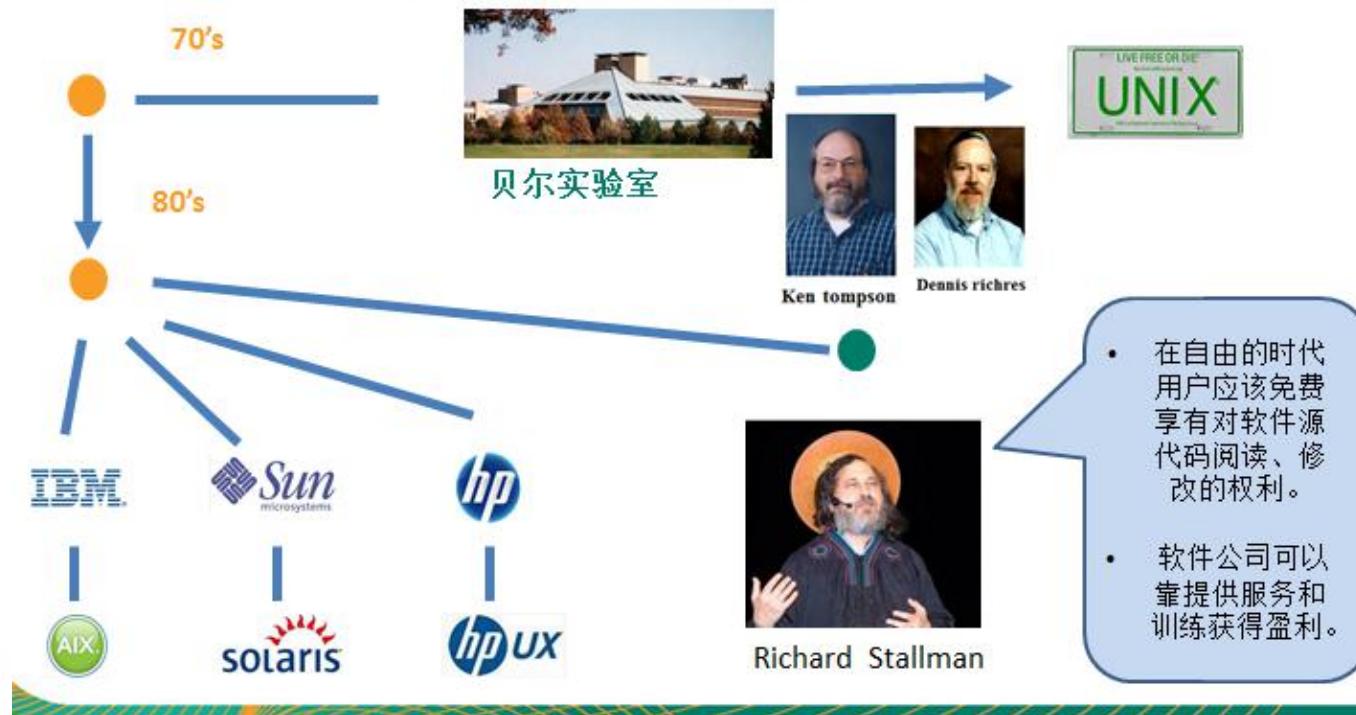
6) 目前主要的操作系统有

windows, android , 车载系统,linux 等。

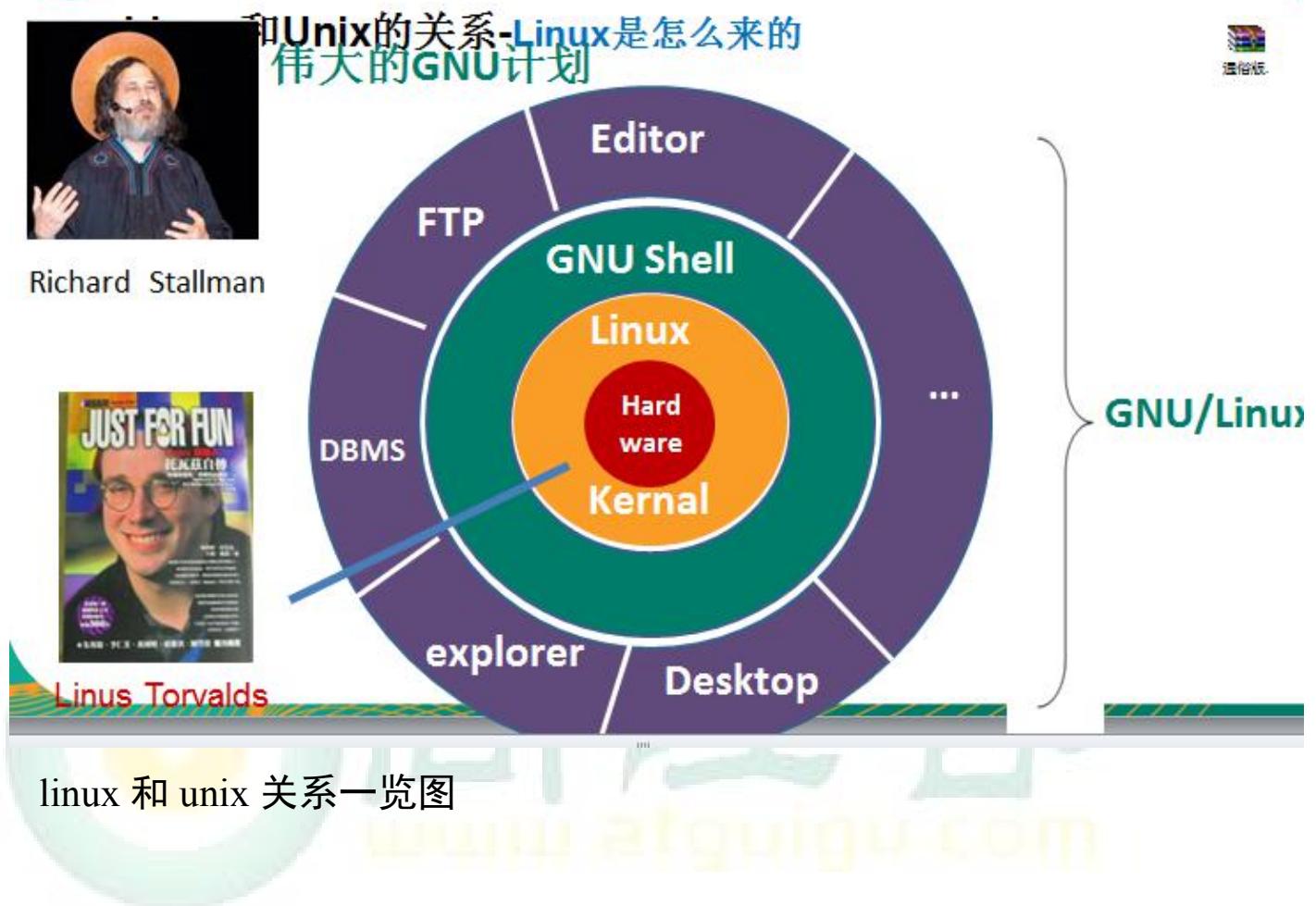


2.2 unix 是怎么来的

● Linux和Unix的关系- unix是怎么来的

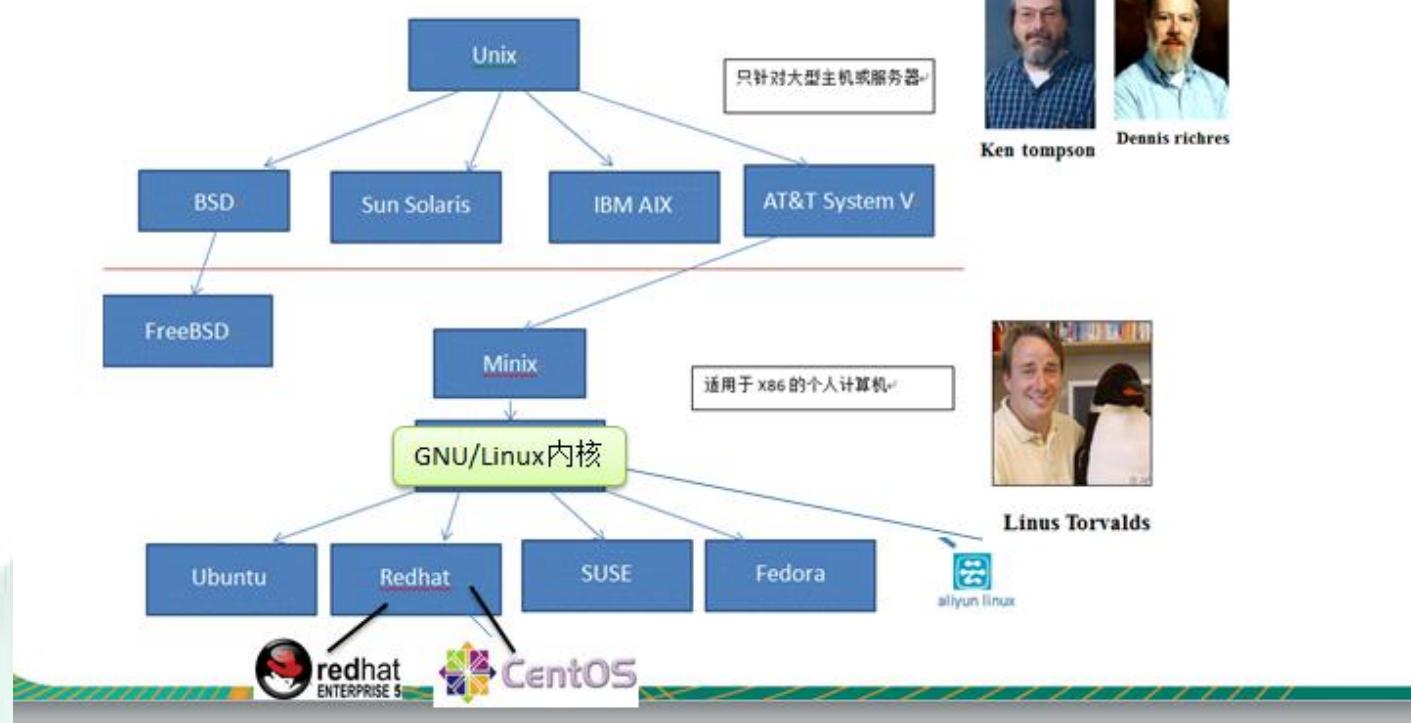


2.3 Linux 是怎么来的



2.4 linux 和 unix 关系一览图

● Linux和Unix的关系



2.5 linux 和 Windows 比较



比较

免费与收费

Window

软件与支持

收费且很贵

安全性

数量和质量的优势，不过大部分为收费软件；由微软官方提供支持和服务；

使用习惯

普通用户基本都是纯图形界面下操作使用，依靠鼠标和键盘完成一切操作，用户上手容易入门简单；

可定制性

封闭的，系统可定制性很差；

应用场景

桌面操作系统主要使用的是window。

Linux

Linux 免费或少许费用。

开源自由软件，用户可以修改定制和再发布，由于基本免费没有资金支持，部分软件质量和体验欠缺；有全球所有的Linux开发者和自由软件社区提供支持。

要说 Linux 没有安全问题，那当然是不可能的，这一点仁者见仁智者见智，相对来说肯定比 Windows 平台要更加安全

兼具图形界面操作和完全的命令行操作，可以只用键盘完成一切操作，新手入门较困难，需要一些学习和指导，一旦熟练之后效率极高。

开源，可定制化非常强。

支撑百度，谷歌，淘宝等应用软件和服务的，是后台成千上万的Linux服务器主机。世界上大部分软件和服务都是运行在Linux之上的。

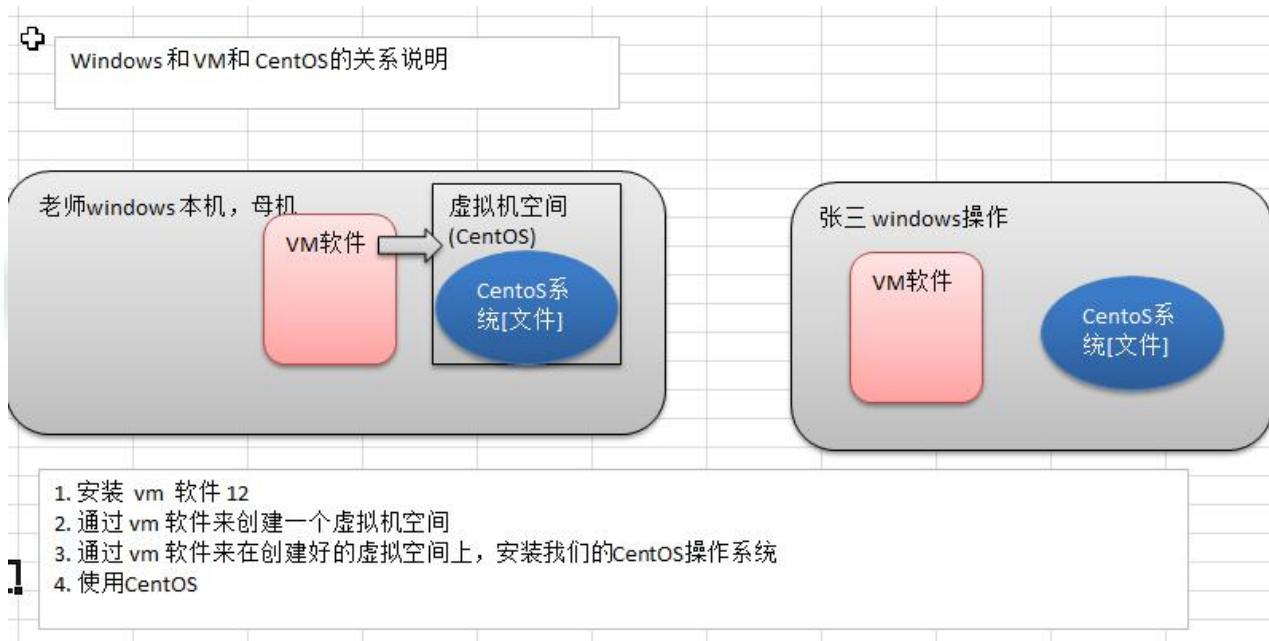


第 3 章 基础篇 VM 和 Linux 系统(CentOS)安装

3.1 安装 vm 和 CentOS

学习 Linux 需要一个环境，我们需要创建一个虚拟机，然后在虚拟机上安装一个 CentOS 系统来学习。

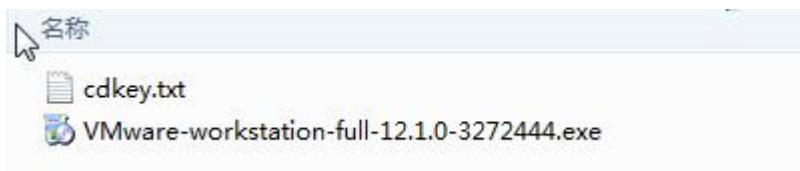
- 1)先安装 virtual machine ,vm12
- 2)再安装 Linux (CentOS 6.8)
- 3)原理示意图，这里我们画图说明一下 VM 和 CentOS 的关系。



3.2 vm 软件和 CentOS 的安装软件

CentOS 下载地址：

- 网易镜像：<http://mirrors.163.com/centos/6/isos/>
- 搜狐镜像：<http://mirrors.sohu.com/centos/6/isos>



3.3 VM 安装的步骤

- 1) 去 BIOS 里修改设置开启虚拟化设备支持 (f2, f10)
- 2) 安装虚拟机软件 (vm12)

看老师的具体演示：因为老师的电脑上已经安装了 VM12，因此我先卸载然后再安装。

3.4 CentOS 安装的步骤

- 1) 创建虚拟机(空间)

这里在配置网络连接时，有三种形式，需要大家伙注意：



- 2) 开始安装系统(CentOS6.8)

- 3) 看老师安装的演示过程。

3.5 CentOS 的终端使用和联网

- 1) 终端的使用，点击鼠标右键，即可选择打开终端



```
root@hadoop1:/  
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)  
[root@hadoop1 桌面]# cd /  
[root@hadoop1 /]# ll  
总用量 98  
dr-xr-xr-x. 2 root root 4096 3月 17 21:22 bin  
dr-xr-xr-x. 5 root root 1024 3月 17 21:27 boot  
drwxr-xr-x. 19 root root 3760 3月 17 21:32 dev  
drwxr-xr-x. 103 root root 12288 3月 17 22:13 etc  
drwxr-xr-x. 2 root root 4096 9月 23 2011 home  
dr-xr-xr-x. 13 root root 4096 3月 17 21:22 lib  
dr-xr-xr-x. 9 root root 12288 3月 17 21:22 lib64  
drwx----- 2 root root 16384 3月 17 21:02 lost+found  
drwxr-xr-x. 2 root root 4096 9月 23 2011 media  
drwxr-xr-x. 2 root root 4096 9月 23 2011 mnt
```

2) 配置网络，可以上网。

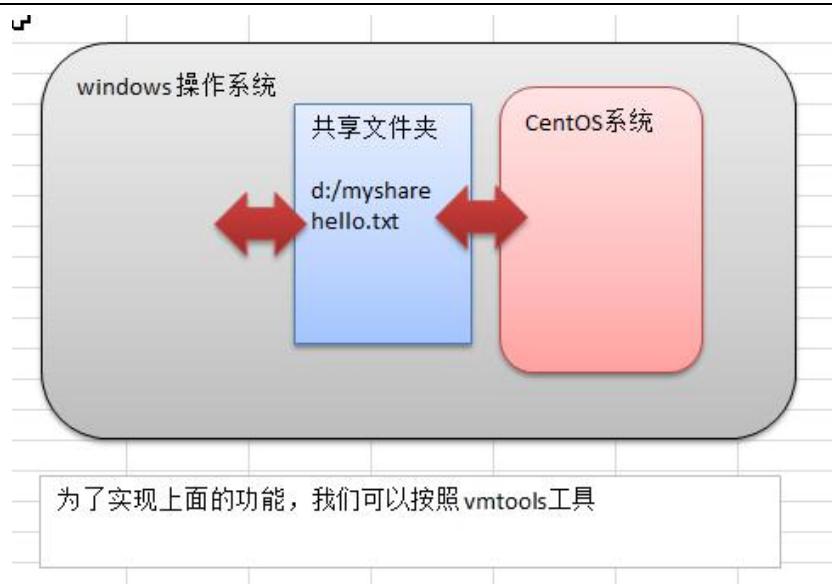
点击上面右侧的；两个计算机图片，选择启用 eth0,即可成功连接到网络，可以上网。



3.6 vmtools 安装

3.6.1 介绍

- 1) 可以直接粘贴命令在 windows 和 centos 系统之间
- 2) 可以设置 windows 和 centos 的共享文件夹
- 3) 示意图



3.7 vmtools 的安装和使用

3.7.1 安装 vmtools 的步骤说明

1. 进入 centos
2. 点击 vm 菜单的->install vmware tools
3. centos 会出现一个 vm 的安装包
4. 点击右键解压，得到一个安装文件
5. 进入该 vm 解压的目录，该文件在 /root/桌面/vmware-tools-distrib/下
6. 安装 ./vmware-install.pl
7. 全部使用默认设置即可
8. 需要 reboot 重新启动即可生效

3.7.2 使用 vmtools 来设置 windows 和 linux 的共享文件夹

- 1) 菜单->vm->setting, 如图设置即可

注意：设置选项为 always enable，
这样可以读写了
- 2) windows 和 centos 可共享 d:/share 目录
可以读写文件了



3)在 centos 的 /mnt/hgfs/ 下

3.7.3 安装 vmtools 的课堂练习

请安装vmtools:

- 1) 并且设置一个共享文件夹c:/myDir，该文件编写一个hello.txt 文件
- 2) 在linux 这边打开，可以看到hello.txt文件

5min



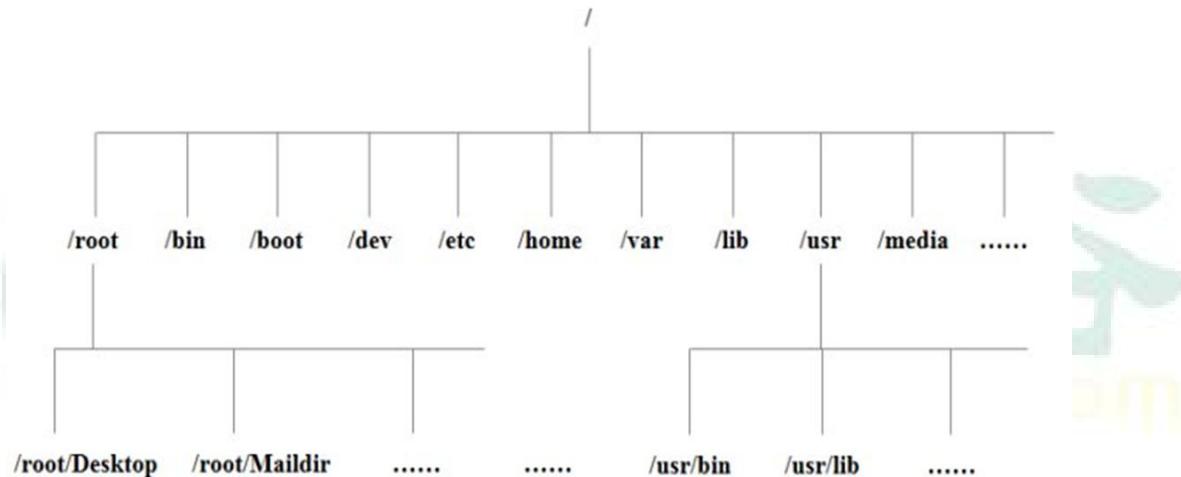
第 4 章 基础篇 Linux 的目录结构

4.1 基本介绍

linux 的文件系统是采用层级式的树状目录结构，在此结构中的最上层是根目录“/”，然后在此目录下再创建其他的目录。

深刻理解 linux 树状文件目录是非常重要的，这里我给大家说明一下。

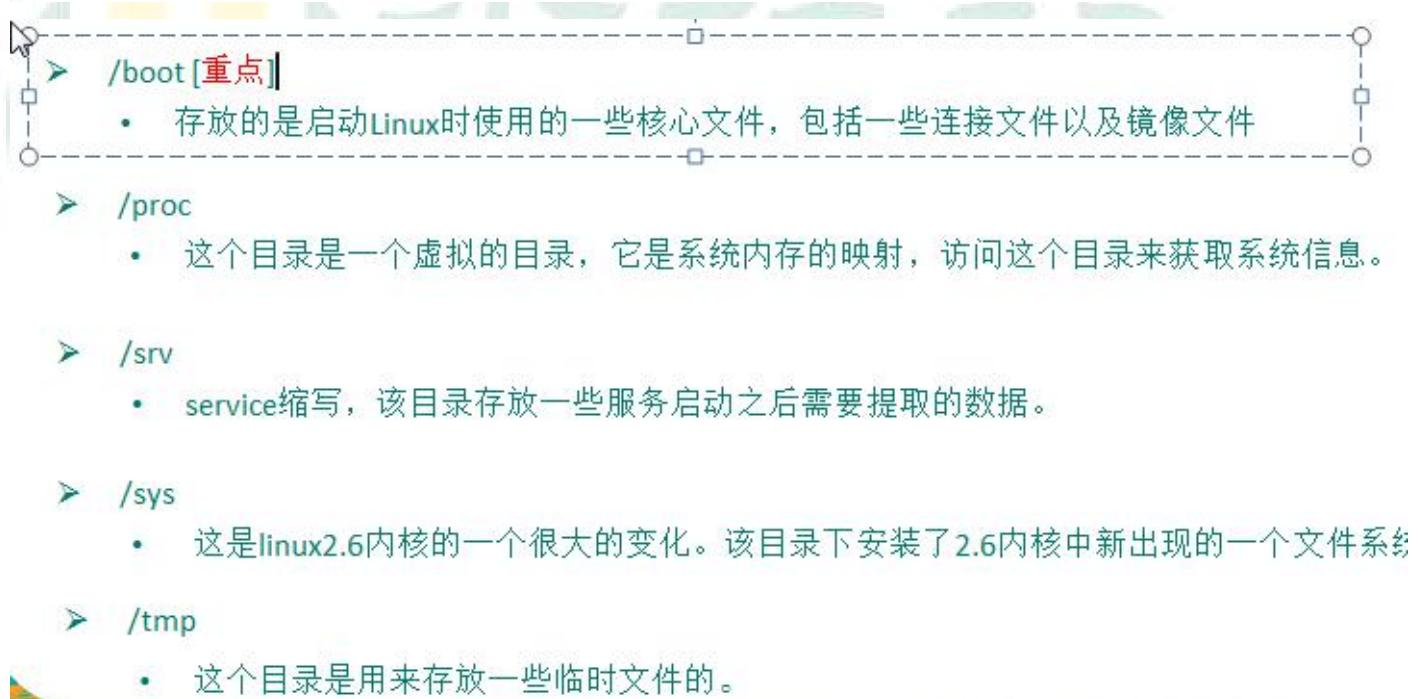
记住一句经典的话：**在 Linux 世界里，一切皆文件。**

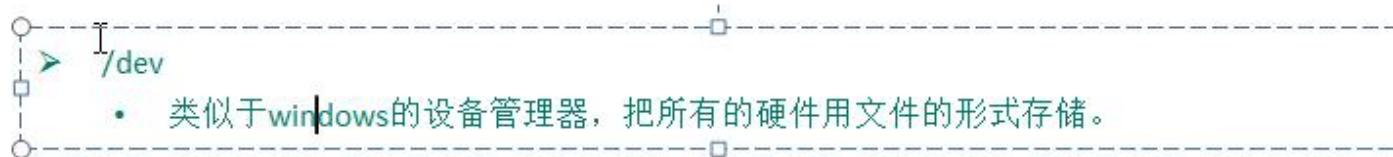


4.2 目录结构的具体介绍

具体的目录结构：

- /bin [重点] (/usr/bin、/usr/local/bin)
 - 是Binary的缩写, 这个目录存放着最经常使用的命令
- /sbin (/usr/sbin、/usr/local/sbin)
 - s就是Super User的意思, 这里存放的是系统管理员使用的系统管理程序。
- /home [重点]
 - 存放普通用户的主目录, 在Linux中每个用户都有一个自己的目录, 一般该目录名是以用户的账号命名的。
- /root [重点]
 - 该目录为系统管理员, 也称作超级权限者的用户主目录。





➤ /media [重点]

- linux 系统会自动识别一些设备，例如 U 盘、光驱等等，当识别后，linux 会把识别的设备挂载到这个目录下。

➤ /mnt [重点]

- 系统提供该目录是为了让用户临时挂载别的文件系统的，我们可以将外部的存储挂载在 /mnt 上，然后进入该目录就可以查看里的内容了。 d:/myshare

➤ /opt

- 这是给主机额外安装软件所摆放的目录。如安装 ORACLE 数据库就可放到该目录下。默认为空。

➤ /usr/local [重点]

- 这是另一个给主机额外安装软件所安装的目录。一般是通过编译源码方式安装的程序。

➤ /var [重点]

- 这个目录中存放着在不断扩充着的东西，习惯将经常被修改的目录放在这个目录下。包括各种日志文件。

➤ /selinux [security-enhanced linux] 360

- SELinux 是一种安全子系统，它能控制程序只能访问特定文件。

4.3 Linux 目录总结一下

- 1) linux 的目录中有且只要一个根目录 /
- 2) linux 的各个目录存放的内容是规划好，不用乱放文件。
- 3) linux 是以文件的形式管理我们的设备，因此 linux 系统，一切皆为文件。



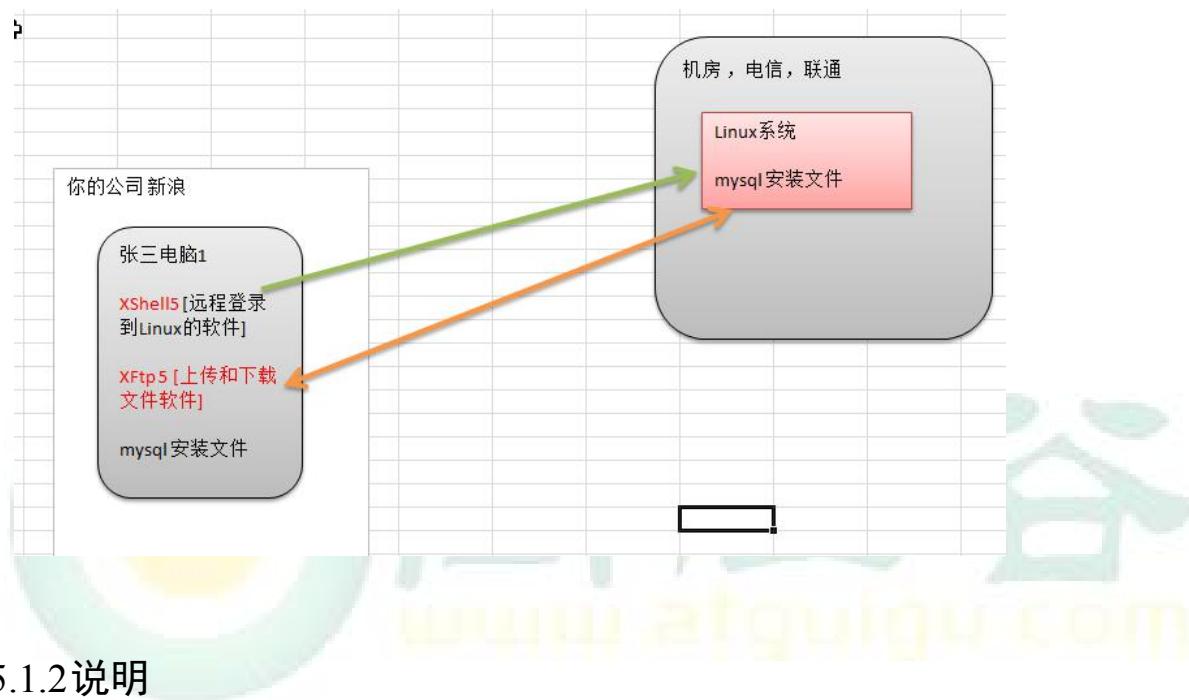
-
- 4) linux 的各个文件目录下存放什么内容，大家必须有一个认识。
 - 5) 学习后，你脑海中应该有一颗 linux 目录树



第 5 章 实操篇 远程登录 Linux 系统

5.1 为什么需要远程登录 Linux

5.1.1 示意图



5.1.2 说明

说明：公司开发时候，具体的情况是这样的

- 1) linux 服务器是开发小组共享的.
- 2) 正式上线的项目是运行在公网的.
- 3) 因此程序员需要远程登录到 centos 进行项目管理或者开发.
- 4) 画出简单的网络拓扑示意图(帮助理解)
- 5) 远程登录客户端有 Xshell5, Xftp5 , 我们学习使用 Xshell5 和 Xftp , 其它的远程工具大同小异.

5.2 远程登录 Linux-Xshell5

说明: Xshell 是目前最好的远程登录到 Linux 操作的软件，流畅的速度并且完美解决了中文乱码

的问题，是目前程序员首选的软件。

Xshell [1] 是一个强大的安全终端模拟软件，它支持 SSH1, SSH2, 以及 Microsoft Windows 平台的 TELNET 协议。

Xshell 可以在 Windows 界面下用来访问远端不同系统下的服务器，从而比较好的达到远程控制终端的目的。

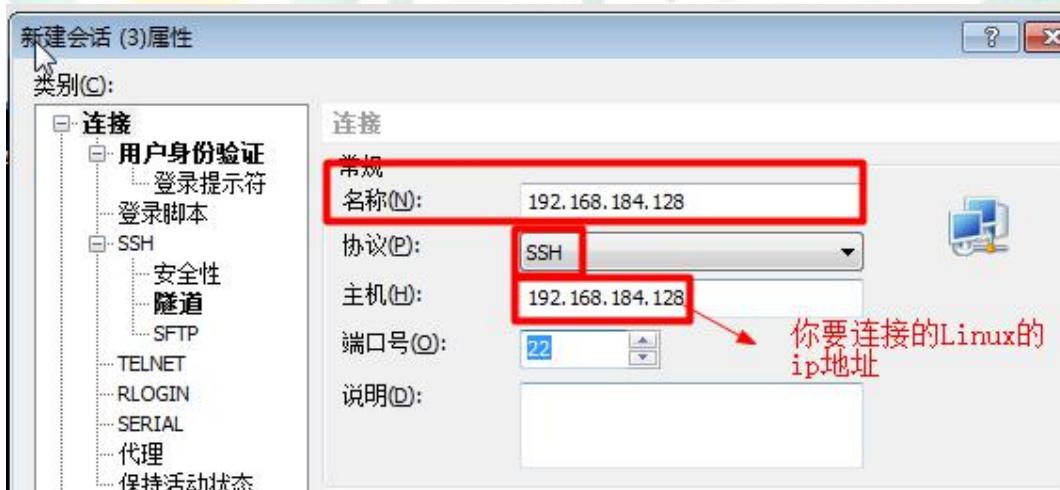
特别说明：如果希望安装好 XShell 5 就可以远程访问 Linux 系统的话，需要有一个前提，就是 Linux 启用了 SSHD 服务，该服务会监听 22 号端口。

5.3 安装 XShell5 并使用

5.3.1 安装过程

看老师的视频演示即可。基本是下一步即可

5.3.2 XShell5 的关键配置



5.3.3 XShell5 远程登录到 Linux 后，就可以使用指令来操作 Linux 系统

```
To escape to local shell, press 'Ctrl+Alt+]'.  
[root@hadoop1 ~]# ll  
总用量 96  
-rw----- 1 root root 1247 3月 17 21:25 anaconda-ks.cfg  
-rw-r--r-- 1 root root 41954 3月 17 21:25 install.log  
-rw-r--r-- 1 root root 9154 3月 17 21:20 install.log.syslog  
drwxr-xr-x. 2 root root 4096 3月 17 21:32 公共的  
drwxr-xr-x. 2 root root 4096 3月 17 21:32 模板  
drwxr-xr-x. 2 root root 4096 3月 17 21:32 视频  
drwxr-xr-x. 2 root root 4096 3月 17 21:32 图片  
drwxr-xr-x. 2 root root 4096 3月 17 21:32 文档  
drwxr-xr-x. 2 root root 4096 3月 17 21:32 下载  
drwxr-xr-x. 2 root root 4096 3月 17 21:32 音乐  
drwxr-xr-x. 2 root root 4096 3月 17 22:50 桌面  
[root@hadoop1 ~]#
```

5.4 远程上传下载文件 Xftp5

5.4.1 Xftp5 软件介绍

是一个基于 windows 平台的功能强大的 SFTP、FTP 文件传输软件。使用了 Xftp 以后，windows 用户能安全地在 UNIX/Linux 和 Windows PC 之间传输文件。(示意图)。



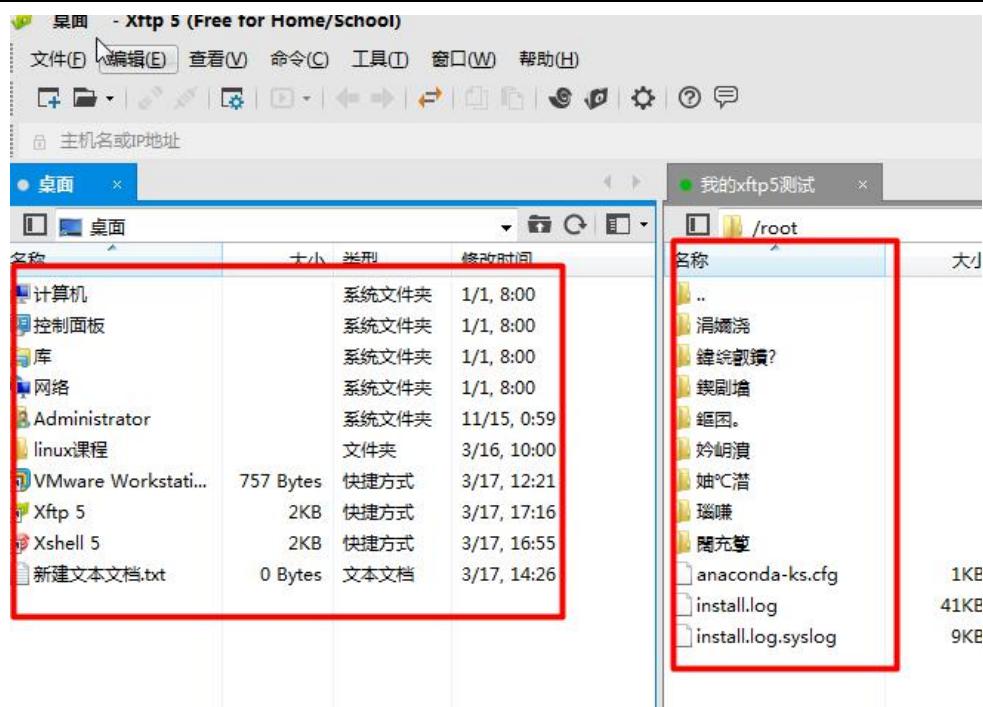
5.4.2 Xftp5 软件的安装

这个看老师的演示即可。

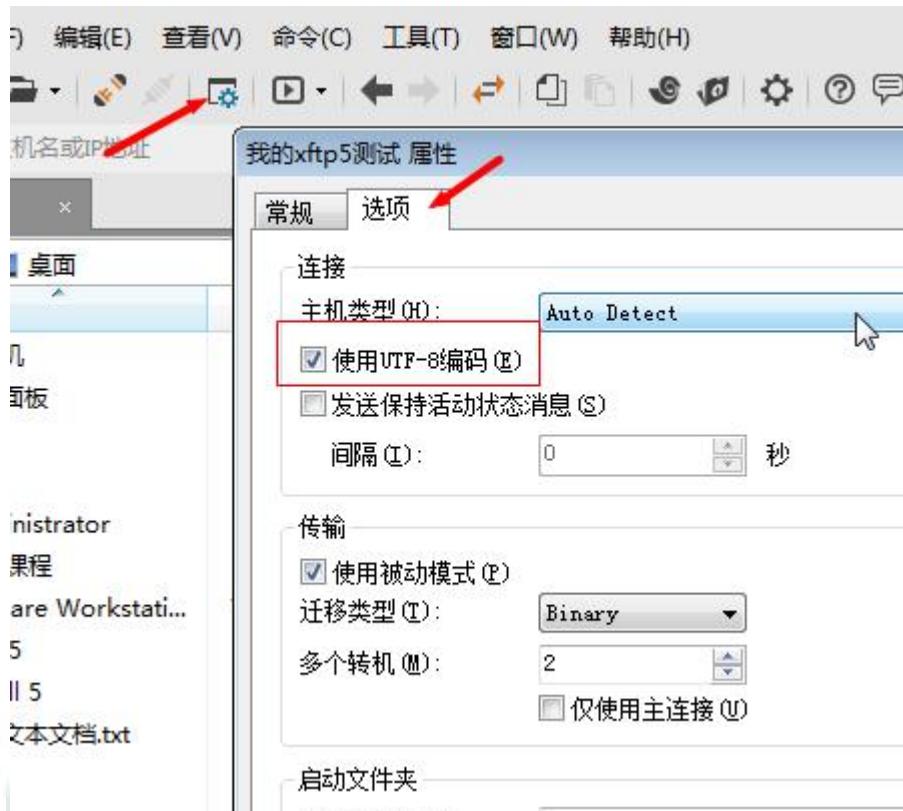
5.4.3 Xftp5 的配置和使用



连接到 Linux 的界面如下,就说明已经成功的远程连接到 Linux



5.4.4 如何解决 XFTP5 中文乱码的问题



说明：如上图配置后，还需要刷新一下，就可以解决中文乱码

5.5 Xftp5 和 XShell5 的使用

练习要求：

- 1. 通过Xftp5软件可以连接到linux，并传输一个文件到 /root目录
- 2. 使用xshell可以连接到linux，并且可以重新启动一把 linux系统 【reboot】

10min

第 6 章 实操篇 vi 和 vim 编辑器

6.1 vi 和 vim 的基本介绍

所有的 Linux 系统都会内建 **vi 文本编辑器**。

Vim 具有程序编辑的能力，可以看做是 **Vi 的增强版本**，可以主动的以字体颜色辨别语法的正确性，方便程序设计。代码补完、编译及错误跳转等方便编程的功能特别丰富，在程序员中被广泛使用。

6.2 vi 和 vim 的三种常见模式

6.2.1 正常模式

在正常模式下，我们可以使用快捷键。

以 vim 打开一个档案就直接进入一般模式了(这是默认的模式)。在这个模式中，你可以使用『上下左右』按键来移动光标，你可以使用『删除字符』或『删除整行』来处理档案内容，也可以使用『复制、贴上』来处理你的文件数据。

6.2.2 插入模式/编辑模式

在模式下，程序员可以输入内容。

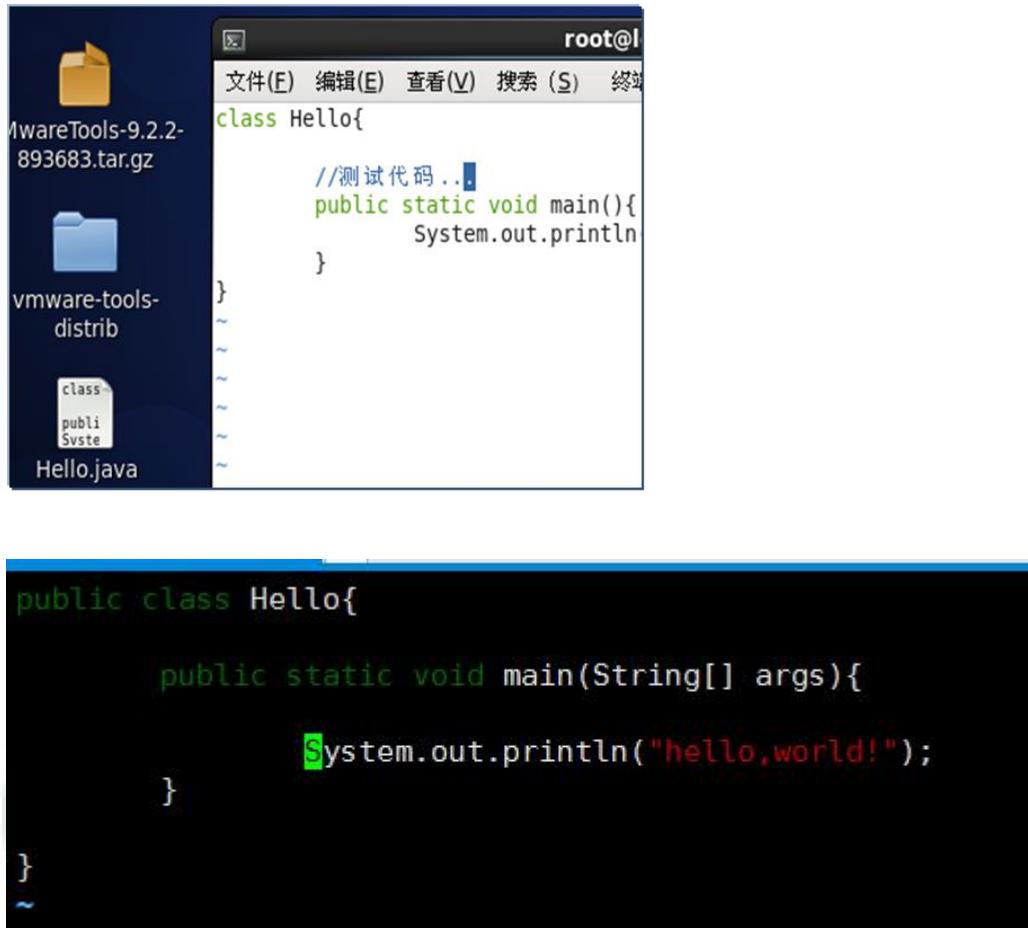
按下 i, I, o, O, a, A, r, R 等任何一个字母之后才会进入编辑模式，一般来说按 i 即可

6.2.3 命令行模式

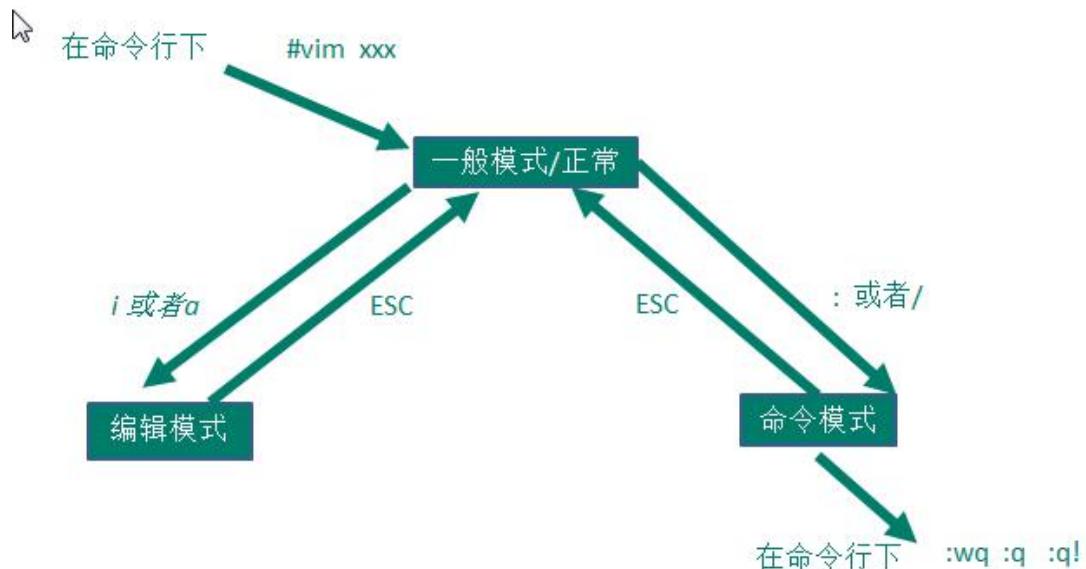
在这个模式当中，可以提供你相关指令，完成读取、存盘、替换、离开 vim 、显示行号等的动作则是在此模式中达成的！

6.3 快速入门案例

使用 vim 开发一个 Hello.java 程序，保存.步骤说明和演示



6.4 vi 和 vim 三种模式的相互转化图

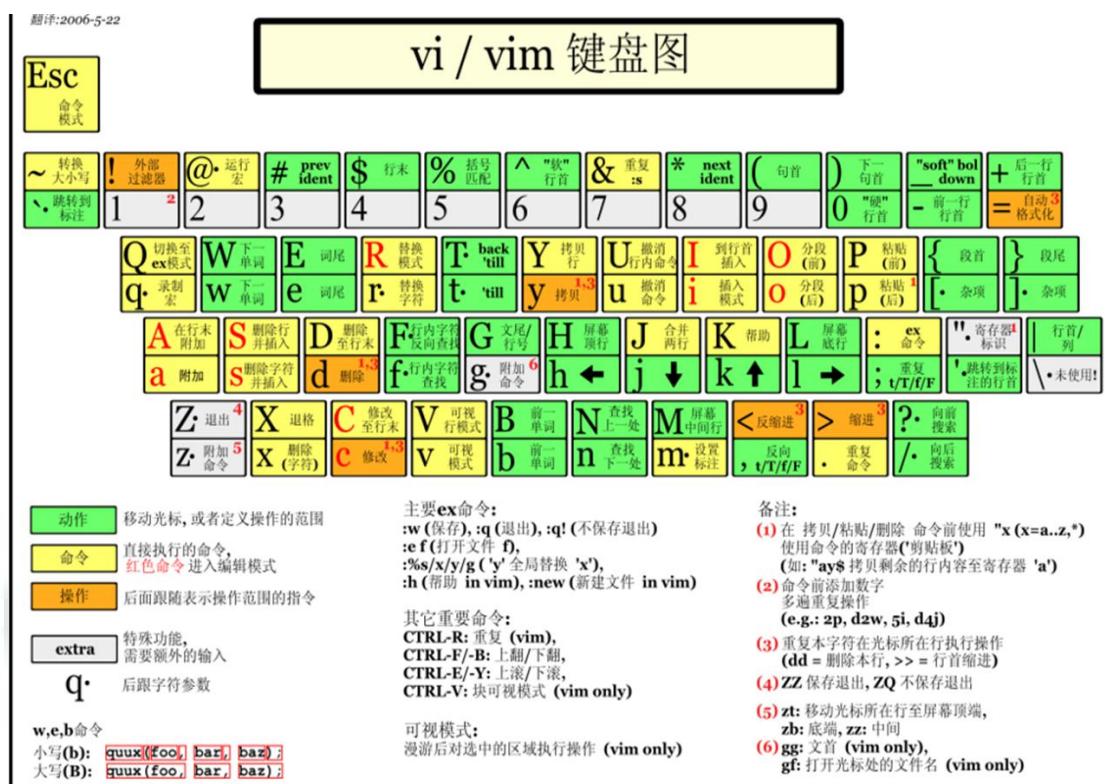


6.5 快捷键的使用案例

- 1) 拷贝当前行 yy , 拷贝当前行向下的 5 行 5yy , 并粘贴 (p) 。
- 2) 删除当前行 dd , 删除当前行向下的 5 行 5dd
- 3) 在文件中查找某个单词 [命令行下 /关键字 , 回车 查找 , 输入 n 就是查找下一个], 查询 hello.
- 4) 设置文件的行号, 取消文件的行号.[命令行下 :set nu 和 :set nonu]
- 5) 编辑 /etc/profile 文件, 使用快捷键到底文档的最末行[G]和最首行[gg], 注意这些都是在正常模式下执行的。
- 6) 在一个文件中输入 "hello" ,然后又撤销这个动作, 再正常模式下输入 u

- 7) 编辑 /etc/profile 文件, 并将光标移动到 第 20 行 shift+g
第一步: 显示行号 :set nu
第二步: 输入 20 这个数
第三步: 输入 shift+g

6.6 vim 和 vi 的快捷键键盘一览图



6.7 vi 和 vim 课堂练习



- 1) 看使用 vim 编写一个 Hello.java
- 2) 可以通过 javac 来编译, 通过 java 来运行 【暂不编译, 创建好即可】
- 3) 把老师课堂讲的各种快捷键自己练习一把即可

第 7 章 实操篇 开机、重启和用户登录注销

7.1 关机&重启命令

7.1.1 基本介绍

shutdown

shutdown -h now : 表示立即关机

shutdown -h 1 : 表示 1 分钟后关机

shutdown -r now: 立即重启

halt

就是直接使用，效果等价于关机

reboot

就是重启系统。

sync : 把内存的数据同步到磁盘

7.1.2 注意细节

当我们关机或者重启时，都应该先执行以下 sync 指令，把内存的数据写入磁盘，防止数据丢失。

7.2 用户登录和注销

7.2.1 基本介绍

- 1) 登录时尽量少用 root 帐号登录，因为它是系统管理员，最大的权限，避免操作失误。可以利用普通用户登录，登录后再用” su - 用户名’ 命令来切换成系统管理员身份.
- 2) 在提示符下输入 logout 即可注销用户

```
[root@hadoop1 ~]#  
[root@hadoop1 ~]#  
[root@hadoop1 ~]# logout  
Connection closing...socket close.  
  
Connection closed by foreign host.  
  
Disconnected from remote host(192.168.184.128) at 19:45:17.  
  
Type `help' to learn how to use Xshell prompt.  
[root@hadoop1 ~]
```

7.2.2 使用细节

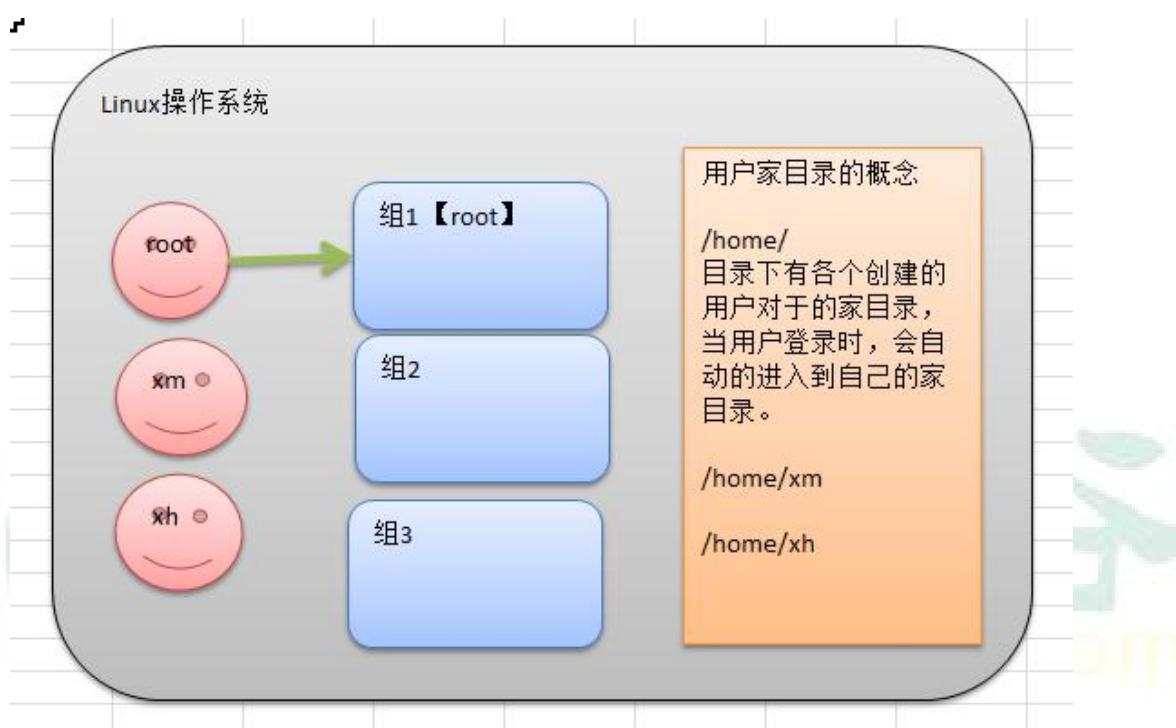
- 1) logout 注销指令在图形运行级别无效，在 运行级别 3 下有效.
- 2) 运行级别这个概念，后面给大家介绍



第 8 章 实操篇 用户管理

8.1 基本介绍

给大家画一个示意图，帮助大家理解用户管理的规则。



说明

- 1) Linux 系统是一个多用户多任务的操作系统，任何一个要使用系统资源的用户，都必须首先向系统管理员申请一个账号，然后以这个账号的身份进入系统。
- 2) Linux 的用户需要至少要属于一个组。

8.2 添加用户

8.2.1 基本语法

useradd [选项] 用户名

8.2.2 实际案例

添加一个用户 xm.

```
[root@hadoop1 ~]#  
[root@hadoop1 ~]# useradd xm  
[root@hadoop1 ~]#  
[root@hadoop1 ~]#
```

特别说明：

cd 表示 change directory, 切换目录.

8.2.3 细节说明

- 1) 当创建用户成功后，会自动的创建和用户名同名的家目录
- 2) 也可以通过 useradd -d 指定目录 新的用户名，给新创建的用户指定家目录

```
[root@hadoop1 ~]#  
[root@hadoop1 ~]#  
[root@hadoop1 ~]# useradd -d /home/dog xq
```

8.3 给用户指定或者修改密码

基本语法

passwd 用户名

应用案例

- 1) 给 xiaoming 指定密码

```
[root@hadoop1 ~]# passwd xm
更改用户 xm 的密码。
新的 密码: 1
无效的密码: WAY 过短
无效的密码: 过于简单
重新输入新的 密码:
passwd: 所有的身份验证令牌已经成功更新。
[root@hadoop1 ~]#
```

1. 在工作中，密码必须复杂点
2. 使用passwd 指定密码时，界面是看不到了

8.4 删除用户

8.4.1 基本语法

userdel 用户名

8.4.2 应用案例

- 1) 删除用户 xm，但是要保留家目录

```
[root@hadoop1 ~]#
[root@hadoop1 ~]# userdel xm
[root@hadoop1 ~]#
```

- 2) 删除用户 xh 以及用户主目录

```
[root@hadoop1 home]#
[root@hadoop1 home]# userdel -r xq
[root@hadoop1 home]#
```

8.4.3 思考题

在删除用户时，我们一般不会将家目录删除。

8.5 查询用户信息

8.5.1 基本语法

id 用户名

8.5.2 应用实例

案例 1：请查询 root 信息

```
[root@hadoop1 home]# id root          → 组名  
uid=0(root) gid=0(root) 组=0(root)  
[root@hadoop1 home]# id xq            → 所在组的ID号  
id: xq: 无此用户                      → 用户ID号  
[root@hadoop1 home]# id xh            → 用户ID号  
uid=501(xh) gid=501(xh) 组=501(xh)  
[root@hadoop1 home]#
```

8.5.3 细节说明

- 1) 当用户不存在时，返回“无此用户”

8.6 切换用户

8.6.1 介绍

在操作 Linux 中，如果当前用户的权限不够，可以通过 `su -` 指令，切换到高权限用户，比如 root

8.6.2 基本语法

`su -` 切换用户名

8.6.3 应用实例

- 1) 创建一个用户 zf，指定密码，然后切换到 zf.

```
[root@hadoop1 home]#  
[root@hadoop1 home]#  
[root@hadoop1 home]# su - xf           → 切换到zf  
[xf@hadoop1 ~]$  
[xf@hadoop1 ~]$  
[xf@hadoop1 ~]$  
[xf@hadoop1 ~]$ cd /root/             → 当用户的权限不足时，会给提示  
-bash: cd: /root/: 权限不够  
[xf@hadoop1 ~]$ exit                  → 返回到原来的用户  
logout  
[root@hadoop1 home]# cd /root/  
[root@hadoop1 ~]#
```

8.6.4 细节说明

- 1) 从权限高的用户切换到权限低的用户，不需要输入密码，反之需要。
- 2) 当需要返回到原来用户时，使用 exit 指令

8.7 用户组

8.7.1 介绍

类似于角色，系统可以对有共性的多个用户进行统一的管理。

8.7.2 增加组

groupadd 组名

8.7.3 案例演示

```
root@hadoop1 ~]#  
root@hadoop1 ~]#  
root@hadoop1 ~]# groupadd wudang  
root@hadoop1 ~]#  
root@hadoop1 ~]#
```

8.7.4 删除组

指令(基本语法)

groupdel 组名

8.7.5 案例演示

```
root@hadoop1 ~]#  
root@hadoop1 ~]# groupdel wudang  
root@hadoop1 ~]#  
root@hadoop1 ~]#
```

8.8 增加用户时直接加上组

8.8.1 指令(基本语法)

```
useradd -g 用户组 用户名
```

8.8.2 案例演示

增加一个用户 zwj, 直接将他指定到 wudang

步骤看演示:

```
[root@hadoop1 ~]# groupadd wudang
[root@hadoop1 ~]# useradd -g wudang zwj
[root@hadoop1 ~]# id zwj
uid=503(zwj) gid=503(wudang) 组=503(wudang)
[root@hadoop1 ~]# cd /home/
```



8.9 修改用户的组

8.9.1 指令(基本语法)

```
usermod -g 用户组 用户名
```

8.9.2 案例演示

创建一个 shaolin 组，让将 zwj 用户修改到 shaolin

```
[root@hadoop1 home]# groupadd shaolin
[root@hadoop1 home]#
[root@hadoop1 home]# usermod -g shaolin zwj
[root@hadoop1 home]# id zwj
uid=503(zwj) gid=504(shaolin) 组=504(shaolin)
[root@hadoop1 home]#
```



8.10 /etc/passwd 文件

用户 (user) 的配置文件，记录用户的各种信息

每行的含义：用户名:口令:用户标识号:组标识号:注释性描述:主目录:登录 Shell

```
pulse:x:497:496:PulseAudio System Daemon:/var/run/pulse
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/s
tcpdump:x:72:72::/:/sbin/nologin
xh:x:501:501::/home/tiger:/bin/bash
xf:x:502:502::/home/xf:/bin/bash
zwj:x:503:504::/home/zwj:/bin/bash
"/etc/passwd" 33L, 1491C
```

用户 id
组 id
家目录
shell

8.11 /etc/shadow 文件

口令的配置文件

每行的含义：登录名:加密口令:最后一次修改时间:最短时间间隔:最长时间间隔:警告时间:不活动时间:失效时间:标志

8.12 /etc/group 文件

组(group)的配置文件，记录 Linux 包含的组的信息

每行含义：组名:口令:组标识号:组内用户列表

```
tcpdump:x:72:
xh:x:501:          组名
xf:x:502:          组的id
wudang:x:503:
shaolin:x:504:
```

第 9 章 实操篇 实用指令

9.1 指定运行级别

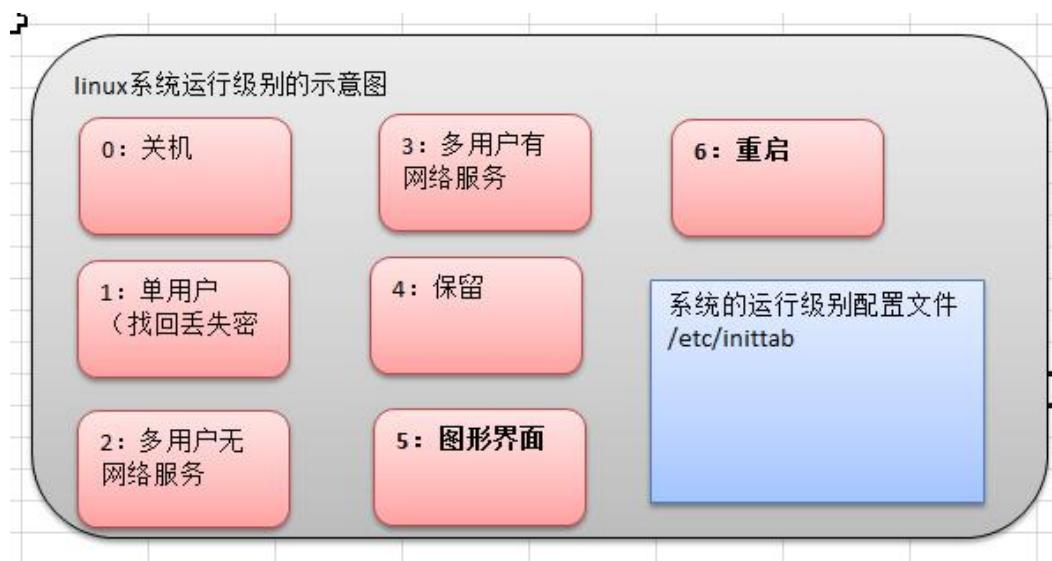
运行级别说明：

- 0 : 关机
- 1 : 单用户【找回丢失密码】
- 2: 多用户状态没有网络服务
- 3: 多用户状态有网络服务
- 4: 系统未使用保留给用户
- 5: 图形界面
- 6: 系统重启

常用运行级别是 3 和 5，要修改默认的运行级别可改文件

/etc/inittab 的 id:5:initdefault:这一行中的数字

运行级别的示意图：



9.2 切换到指定运行级别的指令

9.2.1 基本语法

init [012356]

```
# For information on how to write upstart event handlers, or
# upstart works, see init(5), init(8), and initctl(8).
#
# Default runlevel. The runlevels used are:
#   0 - halt (Do NOT set initdefault to this)
#   1 - Single user mode
#   2 - Multiuser, without NFS (The same as 3, if you do not
#        have NFS)
#   3 - Full multiuser mode
#   4 - unused
#   5 - X11
#   6 - reboot (Do NOT set initdefault to this)
#
#id: 5: initdefault: - /etc/inittab
```

9.2.2 应用实例

案例 1：通过 init 来切换不同的运行级别，比如从 5->3，然后关机

```
init 3
init 5
init 0
```

9.2.3 面试题

如何找回 root 密码，如果我们不小心，忘记 root 密码，怎么找回。

思路：进入到单用户模式，然后修改 root 密码。因为进入单用户模式，root 不需要密码就可以登录。

演示一把（注意观察）：



总结

开机->在引导时输入 回车键-> 看到一个界面输入 e -> 看到一个新的界面，选中第二行（编辑内核）在输入 e-> 在这行最后输入 1 ,再输入 回车键->再次输入 b ,这时就会进入到单用户模式。这时，我们就进入到单用户模式，使用 passwd 指令来修改 root 密码。

9.2.4课堂练习：

- 1) 假设我们的 root 密码忘记了，请问如何找回密码
- 2) 请设置我们的 运行级别，linux 运行后，直接进入到 命令行界面，即进入到 3 运行级别
 vim /etc/inittab
 将 id:5:initdefault:这一行中的数字, 5 这个数字改成对应的运行级别即可。

9.3 帮助指令

9.3.1介绍

当我们对某个指令不熟悉时，我们可以使用 Linux 提供的帮助指令来了解这个指令的使用方法。

9.3.2man 获得帮助信息

- 基本语法
 man [命令或配置文件] (功能描述：获得帮助信息)
- 应用实例

案例：查看 ls 命令的帮助信息

```
drwxr-xr-x. 2 root root 4096 3月
drwxr-xr-x. 2 root root 4096 3月
[root@hadoop1 ~]# man ls
```

9.3.3help 指令

- 基本语法

help 命令（功能描述：获得 shell 内置命令的帮助信息）

- 应用实例

案例：查看 cd 命令的帮助信息

```
[root@hadoop1 ~]# help cd
cd: cd [-L|-P] [dir]
      Change the shell working directory.

      Change the current directory to DIR.  The def
HOME shell variable.
```

9.3.4 当一个指令不熟悉如何学习的建议

百度帮助更直接

虽然上面两个都可以来获取指令帮助，但是需要英语功底，如果英语不太好的，我还是推荐大家直接百度靠谱。 ifconfig



9.4 文件目录类

9.4.1 pwd 指令

- 基本语法

pwd (功能描述：显示当前工作目录的绝对路径)

- 应用实例

案例：显示当前工作目录的绝对路径

```
[root@hadoop1 ~]#  
[root@hadoop1 ~]# pwd  
/root  
[root@hadoop1 ~]#
```

9.4.2 ls 指令

- 基本语法

ls [选项] [目录或是文件]

- 常用选项

-a : 显示当前目录所有的文件和目录，包括隐藏的。

-l : 以列表的方式显示信息

- 应用实例

案例：查看当前目录的所有内容信息

```
/root  
[root@hadoop1 ~]# ls  
anaconda-ks.cfg  Hello.java  install.log.syslog  Ok.java  模板  图片  
Desert.jpg      install.log  Koala.jpg          公共的  视频  文档  
[root@hadoop1 ~]# ls -l  
总用量 1696  
-rw----- 1 root root  1247 3月 17 21:25 anaconda-ks.cfg  
-rw-r--r-- 1 root root 845941 3月 18 01:36 Desert.jpg  
-rw-r--r-- 1 root root   108 3月 18 02:43 Hello.java  
-rw-r--r-- 1 root root  41954 3月 17 21:25 install.log  
-rw-r--r-- 1 root root  9154 3月 17 21:20 install.log.syslog
```

```
drwxr-xr-x. 2 root root 4096 3月 17 22:50 桌面
[root@hadoop1 ~]# ls -al
总用量 1816
dr-xr-x---. 27 root root 4096 3月 18 19:00 .
dr-xr-xr-x. 23 root root 4096 3月 18 18:59 ..
drwxr-xr-x. 2 root root 4096 3月 17 21:33 .abrt
-rw-----. 1 root root 1247 3月 17 21:25 anacond
-rw-----. 1 root root 1005 3月 18 18:55 .bash_h
-rw-r--r--. 1 root root 18 5月 20 2009 .bash_l
-rw-r--r--. 1 root root 176 5月 20 2009 .bash_p
```

9.4.3 cd 指令

- 基本语法

cd [参数] (功能描述: 切换到指定目录)

- 常用参数

绝对路径和相对路径

如何理解绝对路径和相对路径:



cd ~ 或者 cd : 回到自己的家目录

cd .. 回到当前目录的上一级目录

- 应用实例

案例 1: 使用绝对路径切换到 root 目录

```
cd /root
```

案例 2：使用相对路径到/root 目录

这里我们需要知道该用户目录在哪个目录下，才能写出这个指令，假设在/usr/lib

```
cd ../../root
```

案例 3：表示回到当前目录的上一级目录

```
cd ..
```

案例 4：回到家目录

```
cd
```

```
cd ~
```

9.4.4 mkdir 指令

mkdir 指令用于创建目录(make directory)

- 基本语法

```
mkdir [选项] 要创建的目录
```

- 常用选项

- p : 创建多级目录

- 应用实例

案例 1: 创建一个目录 /home/dog

```
[root@hadoop1 ~]# mkdir /home/dog
```

表示在/home目录下，创建 dog目录

```
[root@hadoop1 ~]# cd /home/
[root@hadoop1 home]# ls
dog hello tiger xf xm zwj
[root@hadoop1 home]#
```

案例 2: 创建多级目录 /home/animal/tiger

```
[root@hadoop1 home]# mkdir /home/animal/tiger  
mkdir: 无法创建目录"/home/animal/tiger": 没有那个文件或目录  
[root@hadoop1 home]# mkdir -p /home/animal/tiger  
[root@hadoop1 home]#  
[root@hadoop1 home]# ls  
animal dog hello tiger xf xm zwj           ↗ 如果希望一次创建多级目录  
[root@hadoop1 home]# cd animal/  
[root@hadoop1 animal]# pwd
```

9.4.5 rmdir 指令

- 介绍

rmdir 指令删除空目录

- 基本语法

`rmdir [选项] 要删除的空目录`

- 应用实例

案例 1: 删除一个目录 /home/dog

```
[root@hadoop1 animal]# rmdir /home/dog  
[root@hadoop1 animal]#  
[root@hadoop1 animal]# cd /home/  
[root@hadoop1 home]# ls  
animal hello tiger xf xm zwj
```

- 使用细节

rmdir 删除的是空目录，如果目录下有内容时无法删除的。

提示：如果需要删除非空目录，需要使用 `rm -rf` 要删除的目录

```
[root@hadoop1 dog]# pwd  
/home/dog  
[root@hadoop1 dog]# rmdir /home/dog          ↗ rmdir 不能删除非空的目录  
rmdir: 删除 "/home/dog" 失败: 目录非空  
[root@hadoop1 dog]#  
[root@hadoop1 dog]# rm -rf /home/dog          ↗ 如果希望删除非空的目录，可  
[root@hadoop1 dog]#  
[root@hadoop1 dog]#
```

9.4.6 touch 指令

touch 指令创建空文件

- 基本语法

touch 文件名称

- 应用实例

案例 1：创建一个空文件 hello.txt

```
[root@hadoop1 home]#  
[root@hadoop1 home]# touch hello.txt  
[root@hadoop1 home]# ls  
animal  hello  hello.txt  test  tiger  xf  xm  
[root@hadoop1 home]# ls -l  
drwxr-xr-x. 4 zhang  4096 3月 18 20:43 .  
[root@hadoop1 home]# touch ok1.txt ok2.txt  
[root@hadoop1 home]# ls  
animal  hello  hello.txt  ok1.txt  ok2.txt  test  tiger  xf  xm  
[root@hadoop1 home]#
```

9.4.7 cp 指令[重要]

cp 指令拷贝文件到指定目录

- 基本语法

cp [选项] source dest

- 常用选项

-r : 递归复制整个文件夹

- 应用实例

案例 1：将 /home/aaa.txt 拷贝到 /home/bbb 目录下[拷贝单个文件]

```
[root@hadoop1 home]# touch aaa.txt  
[root@hadoop1 home]# ls  
aaa.txt animal hello hello.txt ok1.txt ok2.txt test tiger xf  
[root@hadoop1 home]# mkdir bbb  
[root@hadoop1 home]# ls  
aaa.txt animal bbb hello hello.txt ok1.txt ok2.txt test tiger  
[root@hadoop1 home]# cp aaa.txt bbb/  
[root@hadoop1 home]# cd bbb/  
[root@hadoop1 bbb]# ls  
aaa.txt  
[root@hadoop1 bbb]#
```

表示将当前目录的aaa.txt文件拷贝到当前目录的bbb这个目录下

案例 2: 递归复制整个文件夹, 举例

将/home/test 整个目录拷贝到 /home/zwj 目录

```
[root@hadoop1 home]# cp -r test/ zwj/  
[root@hadoop1 home]# cd zwj/  
[root@hadoop1 zwj]# ls  
test  
[root@hadoop1 zwj]# cd test/  
[root@hadoop1 test]# ls  
a.txt b.txt d.txt  
[root@hadoop1 test]# pwd  
/home/zwj/test  
[root@hadoop1 test]#
```

注意: 一定要注意你当前的目录位置, 然后要准确的定位 源目录和目标目录

• 使用细节

强制覆盖不提示的方法: \cp

```
[root@hadoop1 home]#  
[root@hadoop1 home]# cp -r test/ zwj/  
cp: 是否覆盖"zwj/test/a.txt"? y  
cp: 是否覆盖"zwj/test/b.txt"? y  
cp: 是否覆盖"zwj/test/a.txt"? y  
[root@hadoop1 home]# \cp -r test/ zwj/  
[root@hadoop1 home]#  
[root@hadoop1 home]#
```

这个指令, 当发现目标目录下有相同文件, 会提示你是否覆盖

这个指令会强制覆盖原来的文件, 不会提示

技术小技巧:

可以通过 上下箭头的键, 调出原来使用过的指令。

9.4.8 rm 指令

rm 指令移除【删除】文件或目录

- 基本语法

```
rm [选项] 要删除的文件或目录
```

- 常用选项

-r : 递归删除整个文件夹

-f : 强制删除不提示

- 应用实例

案例 1: 将 /home/aaa.txt 删除

```
aaa.txt animal bbb hello hello.txt ok1.txt  
[root@hadoop1 home]# rm aaa.txt  
rm: 是否删除普通空文件 "aaa.txt"? y  
[root@hadoop1 home]# ls  
animal bbb hello hello.txt ok1.txt ok2.txt  
[root@hadoop1 home]#
```

案例 2: 递归删除整个文件夹 /home/bbb

```
animal bbb hello hello.txt ok1.txt ok2.txt  
[root@hadoop1 home]# rm -rf bbb/  
[root@hadoop1 home]# ls  
animal hello hello.txt ok1.txt ok2.txt  
[root@hadoop1 home]#
```

- 使用细节

强制删除不提示的方法: 带上 -f 参数即可

```
animal hello hello.txt ok1.txt ok2.txt test  
[root@hadoop1 home]# rm -f ok1.txt  
[root@hadoop1 home]# rm ok2.txt  
rm: 是否删除普通空文件 "ok2.txt"? y  
[root@hadoop1 home]#
```

9.4.9 mv 指令

mv 移动文件与目录或重命名

- 基本语法

```
mv oldNameFile newNameFile      (功能描述: 重命名)
```

```
mv /temp/movefile /targetFolder (功能描述: 移动文件)
```

- 应用实例

案例 1: 将 /home/aaa.txt 文件 重新命名为 pig.txt

```
[root@hadoop1 home]# ls  
aaa.txt animal hello hello.txt test tiger xf xm  
[root@hadoop1 home]# mv aaa.txt pig.txt      将aaa.txt改名  
[root@hadoop1 home]# ls  
animal hello hello.txt pig.txt test tiger xf xm  
[root@hadoop1 home]#
```

案例 2: 将 /home/pig.txt 文件 移动到 /root 目录下

```
animal hello hello.txt pid.txt test tiger xt xm zwj  
[root@hadoop1 home]# mv pig.txt /root/      将当前目录的pig.txt移动  
[root@hadoop1 home]# cd /root/  
[root@hadoop1 ~]# ls  
anaconda-ks.cfg Hello.java install.log.syslog Ok.java 公共的 视频  
Desert.jpg install.log Koala.jpg          pig.txt 模板 图片  
[root@hadoop1 ~]#
```

9.4.10 cat 指令

cat 查看文件内容，是以只读的方式打开。

- 基本语法

```
cat [选项] 要查看的文件
```

- 常用选项

-n : 显示行号

- 应用实例

案例 1: /etc/profile 文件内容，并显示行号

```
77 unset i  
78 unset -f pathmunge  
t@hadoop1 ~]# cat -n /etc/profile | more
```

以 cat 指令打开文件，并分页显示

- 使用细节

cat 只能浏览文件，而不能修改文件，为了浏览方便，一般会带上 管道命令 | more

cat 文件名 | more [分页浏览]

9.4.11 more 指令

more 指令是一个基于 VI 编辑器的文本过滤器，[它以全屏幕的方式按页显示文本文件的内容](#)。more 指令中内置了若干快捷键，详见操作说明

- 基本语法

more 要查看的文件

- 操作说明

- 应用实例

案例：采用 more 查看文件

/etc/profile

```
isr/bin/id ]; then  
op1 ~]#  
op1 ~]# more /etc/profile
```

快捷键一栏

操作	功能说明
空白键 (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
:f	输出文件名和当前行的行号

9.4.12 less 指令

less 指令用来**分屏查看文件内容**，它的功能与 more 指令类似，但是比 more 指令更加强大，支持各种显示终端。less 指令在显示文件内容时，并不是一次将整个文件加载之后才显示，而是根据显示需要加载内容，对于显示大型文件具有较高的效率。

- 基本语法

 less 要查看的文件

- 操作说明

- 应用实例

 案例：采用 less 查看一个大文件文件 /opt/金庸-射雕英雄传 txt 精校版.txt

```
~]# 
~]# cd /opt/
opt]# ls
ls-10.0.5-3228253.tar.gz  vmware-tools-distrib  金
opt]# less 金庸-射雕英雄传txt精校版.txt
opt]# 
```

快捷键

操作	功能说明
→ 空白键	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；
?字符串	向上搜寻「字符串」的功能；n：向上查找；N：向下查找；
q	离开 less 这个程序；

9.4.13 > 指令 和 >> 指令

介绍

> 指令 和 >> 指令

> 输出重定向：会将原来的文件的内容覆盖

>> 追加： 不会覆盖原来文件的内容，而是追加到文件的尾部。

• 基本语法

1) ls -l >文件 （功能描述：列表的内容写入文件 a.txt 中（覆盖写））

```
[root@hadoop1 ~]# ls -l > a.txt
[root@hadoop1 ~]# more a.txt
总用量 28
drwxr-xr-x. 3 root root 4096 3月 18 20:09 animal
-rw-r--r--. 1 root root 0 3月 18 23:14 a.txt
-rw-r--r--. 1 root root 0 3月 18 01:06 hello
-rw-r--r--. 1 root root 0 3月 18 20:45 hello.txt
-rw-r--r--. 1 root root 215 3月 18 23:07 mydate.txt
drwxr-xr-x. 2 root root 4096 3月 18 20:39 test
drwxr-xr-x. 2 root root 4096 3月 18 04:12 tiger
drwx----- 4 xf xf 4096 3月 18 04:45 xf
drwx----- 4 500 500 4096 3月 18 04:08 xm
drwx----- 5 zwj wudang 4096 3月 18 20:53 zwj
```

说明：ls -l > a.txt，将 ls -l 的显示的内容覆盖写入到 a.txt 文件，如果该文件不存在，就创建该文件。

2) ls -al >>文件 (功能描述: 列表的内容追加到文件 aa.txt 的末尾)

```
[root@hadoop1 home]# ls -l >> b.txt  
[root@hadoop1 home]# more b.txt  
将ls -l显示的内容追加到后面文件中。  
总用量 32  
drwxr-xr-x. 3 root root 4096 3月 18 20:09 animal  
-rw-r--r--. 1 root root 534 3月 18 23:14 a.txt  
-rw-r--r--. 1 root root 0 3月 18 23:16 b.txt  
-rw-r--r--. 1 root root 0 3月 18 01:06 hello  
-rw-r--r--. 1 root root 0 3月 18 20:45 hello.txt  
-rw-r--r--. 1 root root 215 3月 18 23:07 mydate.txt
```

3) cat 文件 1 > 文件 2 (功能描述: 将文件 1 的内容覆盖到文件 2)

```
i  
-f pathmunge  
[root@hadoop1 home]# cat /etc/profile > c.txt  
[root@hadoop1 home]#
```

4) echo "内容" >> 文件

• 应用实例

案例 1: 将 /home 目录下的文件列表 写入到 /home/info.txt 中

```
[root@hadoop1 home]# ls -l /home/ > /home/info.txt  
[root@hadoop1 home]# more info.txt  
[root@hadoop1 home]#  
[root@hadoop1 home]#  
[root@hadoop1 home]# 总用量 44  
drwxr-xr-x. 3 root root 4096 3月 18 20:09 animal  
-rw-r--r--. 1 root root 534 3月 18 23:14 a.txt  
-rw-r--r--. 1 root root 1172 3月 18 23:17 b.txt  
-rw-r--r--. 1 root root 18 3月 18 23:21 c.txt  
-rw-r--r--. 1 root root 18 3月 18 23:21 c.xt  
-rw-r--r--. 1 root root 0 3月 18 01:06 hello  
-rw-r--r--. 1 root root 0 3月 18 20:45 hello.txt
```

案例 2: 将当前日历信息 追加到 /home/mycal 文件中 [提示 cal]

```
root@hadoop1 home]#  
root@hadoop1 home]# cat >> /home/mycat
```

9.4.14 echo 指令

echo 输出内容到控制台。

- 基本语法

```
echo [选项] [输出内容]
```

- 应用实例

案例：使用 echo 指令输出环境变量，输出当前的环境路径。

```
root@hadoop1 home]#  
root@hadoop1 home]# echo $PATH  
/usr/lib64/qt-3.3/bin:/usr/local/sbin:/usr/local/bi  
root@hadoop1 home]#
```

案例：使用 echo 指令输出 hello,world!

9.4.15 head 指令

head 用于显示文件的开头部分内容，默认情况下 head 指令显示文件的前 10 行内容

- 基本语法

```
head 文件 (功能描述：查看文件头 10 行内容)
```

```
head -n 5 文件 (功能描述：查看文件头 5 行内容，5 可以是任意行数)
```

- 应用实例

案例：查看/etc/profile 的前面 5 行代码

```
[root@hadoop1 ~]# head -n 5 /etc/profile  
# /etc/profile  
  
# System wide environment and startup programs, for logi  
# Functions and aliases go in /etc/bashrc  
  
[root@hadoop1 ~]# head /etc/profile
```

9.4.16 tail 指令

tail 用于输出文件中尾部的内容， 默认情况下 tail 指令显示文件的后 10 行内容。

- 基本语法

- 1) tail 文件 （功能描述：查看文件后 10 行内容）
- 2) tail -n 5 文件 （功能描述：查看文件后 5 行内容， 5 可以是任意行数）
- 3) tail -f 文件 （功能描述：实时追踪该文档的所有更新， 工作经常使用）

- 应用实例

案例 1：查看/etc/profile 最后 5 行的代码

```
[root@hadoop1 ~]# tail -n 5 /etc/profile  
# /etc/profile  
  
# System wide environment and startup programs, for logi  
# Functions and aliases go in /etc/bashrc  
  
[root@hadoop1 ~]# head /etc/profile
```

案例 2：实时监控 mydate.txt，看看到文件有变化时，是否看到， 实时的追加日期

```
[root@hadoop1 ~]# ls
animal  a.txt  b.txt  c.txt  c.txt  hello  hello.txt  info.txt  myc
[root@hadoop1 ~]# tail -f mydate.txt
2018年 03月 18日 星期日 23:07:27 CST
2018年 03月 18日 星期日 23:07:37 CST
2018年 03月 18日 星期日 23:07:38 CST
2018年 03月 18日 星期日 23:07:38 CST
2018年 03月 18日 星期日 23:07:40 CST
总用量 52
drwxr-xr-x. 3 root root 4096 3月 18 20:09 animal
-rw-r--r--. 1 root root 524 2月 18 22:14 a.txt
```

实时的监控 mydate.txt
有没有变化，如果有变化，就会看到。

9.4.17 ln 指令

软链接也叫符号链接，类似于 windows 里的快捷方式，主要存放了链接其他文件的路径

- 基本语法

ln -s [原文件或目录] [软链接名] (功能描述：给原文件创建一个软链接)

- 应用实例

案例 1：在/home 目录下创建一个软连接 linkToRoot，连接到 /root 目录

```
[root@hadoop1 ~]#
[root@hadoop1 ~]# ls
animal  a.txt  b.txt  c.txt  c.txt  hello  hello.txt  info.txt  myc
[root@hadoop1 ~]# ln -s /root  linkToRoot
[root@hadoop1 ~]# ls
animal  a.txt  b.txt  c.txt  c.txt  hello  hello.txt  info.txt  linkToRoot
[root@hadoop1 ~]# ls -l
total 52
drwxr-xr-x. 5 zwj  wudang 4096 3月 18 20:53 zwj
-rw-r--r--. 1 root root 524 2月 18 22:14 a.txt
```

原目录或文件 软链接文件

案例 2：删除软连接 linkToRoot

```
drwxr-xr-x. 5 zwj  wudang 4096 3月 18 20:53 zwj
[root@hadoop1 ~]# rm -rf linkToRoot/
rm: 无法删除 "linkToRoot/.gvfs"：设备或资源忙
[root@hadoop1 ~]# rm -rf linkToRoot
[root@hadoop1 ~]# ls
```

在删除软链接文件时，不要带/，否则提示，资源忙。

- 细节说明

当我们使用 pwd 指令查看目录时，仍然看到的是软链接所在目录。

9.4.18 history 指令

查看已经执行过历史命令,也可以执行历史指令

- 基本语法

history (功能描述：查看已经执行过历史命令)

- 应用实例

案例 1：显示所有的历史命令

```
179 ls -l  
180 history  
root@hadoop1 home]# history
```

仅将文本发送到当前选项卡

案例 2：显示最近使用过的 10 个指令。

```
180 history  
[root@hadoop1 home]# history 10  
172 ls  
173 cd ..  
174 ls  
175 ls -l  
176 rm -rf linkToRoot/  
177 rm -rf linkToRoot  
178 ls  
179 ls -l  
180 history
```

案例 3：执行历史编号为 5 的指令

```
181 history 10
[root@hadoop1 home]# !178
ls
animal a.txt b.txt c.txt c.ct hello hell
[root@hadoop1 home]#
```

执行编号为178的指令

9.5 时间日期类

9.5.1 date 指令-显示当前日期

- 基本语法

- 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 %H:%M:%S" (功能描述: 显示年月日时分秒)

- 应用实例

案例 1: 显示当前时间信息

```
[root@hadoop1 home]# date
2018年 11月 11日 星期日 11:14:12 CST
[root@hadoop1 home]#
```

案例 2: 显示当前时间年月日

```
2018-11-11 11:14:12 CST
[root@hadoop1 home]# date "+%Y-%m-%d"
2018-11-11
[root@hadoop1 home]# date "+%Y %m %d"
2018 11 11
[root@hadoop1 home]#
```

案例 3: 显示当前时间年月日时分秒

```
[root@hadoop1 ~]# date "+%Y-%m-%d %H:%M:%S"
2018-11-11 11:22:05
[root@hadoop1 ~]#
```

9.5.2 date 指令-设置日期

- 基本语法

```
date -s 字符串时间
```

- 应用实例

案例 1：设置系统当前时间，比如设置成 2018-10-10 11:22:22

```
[root@hadoop1 ~]# date
2018-11-11 11:22:05
[root@hadoop1 ~]# date -s "2018-10-10 11:22:22"
2018年 10月 10日 星期三 11:22:22 CST
[root@hadoop1 ~]# date
2018年 10月 10日 星期三 11:22:28 CST
[root@hadoop1 ~]#
```

9.5.3 cal 指令

查看日历指令

- 基本语法

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

- 应用实例

案例 1：显示当前日历

```
2018年 10月 10日 星期三 11:22:28 CST
[ root@hadoop1 ~]# cal
    十月 2018
日 一 二 三 四 五 六
 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 31
```

案例 2: 显示 2020 年日历

```
[ root@hadoop1 ~]# cal 2020
2020

    一月          二月          三月
日 一 二 三 四 五 六 日 一 二 三 四 五 六 日 一 二 三 四 五 六
 1  2  3  4      1  2  3  4  5  6  7  1  2  3  4  5  6  7
 5  6  7  8  9 10 11  2  3  4  5  6  7  8  8  9 10 11 12 13 14
12 13 14 15 16 17 18  9 10 11 12 13 14 15 15 16 17 18 19 20 21
19 20 21 22 23 24 25 16 17 18 19 20 21 22 22 23 24 25 26 27 28
26 27 28 29 30 31 23 24 25 26 27 28 29 29 30 31
```

9.6 搜索查找类

9.6.1 find 指令

find 指令将从指定目录向下递归地遍历其各个子目录，将满足条件的文件或者目录显示在终端。

- 基本语法

```
find [搜索范围] [选项]
```

- 选项说明

选项	功能
-name<查询方式>	按照指定的文件名查找模式查找文件
-user<用户名>	查找属于指定用户名所有文件
-size<文件大小>	按照指定的文件大小查找文件。

- 应用实例

案例 1: 按文件名: 根据名称查找/home 目录下的 hello.txt 文件

```
[root@hadoop1 ~]# find /home -name hello.txt  
/home/hello.txt  
[root@hadoop1 ~]#
```

↑ 搜索的范围 按照名字 要查找的文件

案例 2：按拥有者：查找/opt 目录下，用户名为 nobody 的文件

```
[root@hadoop1 ~]# find /opt -user nobody  
[root@hadoop1 ~]#  
[root@hadoop1 ~]#
```

案例 3：查找整个 linux 系统下大于 20m 的文件 (+n 大于 -n 小于 n 等于)

```
[root@hadoop1 ~]#  
[root@hadoop1 ~]#  
[root@hadoop1 ~]# find / -size +20M  
/boot/initramfs-2.6.32-042.el6.x86_64.img  
/var/cache/yum/x86_64/6/updates/5b9fee46107f23caad93a81  
  
-rw-r--r--. 1 root root 20M 11月 11 2010 /usr/  
[root@hadoop1 ~]# find / -size -20M  
  
仅将文本发送到当前选项卡
```

```
-rw-r--r--. 1 root root 20M 11月 11 2010 /usr/share  
[root@hadoop1 ~]# find / -size 20M  
find: “/proc/3340/task/3340/fd/5”: 没有那个文件或目录  
find: “/proc/3340/task/3340/fdinfo/5”: 没有那个文件
```

1m = 1024k

```
root@hadoop1 ~]# find / -size +20480K  
find: 无效的 -size 类型“K”  
root@hadoop1 ~]# find / -size +20480k  
/boot/initramfs-2.6.32-042.el6.x86_64.img  
/var/cache/yum/x86_64/6/updates/5b9fee46107f23caad93  
/var/cache/yum/x86_64/6/updates/f7462271d9ac780ce9d8  
/var/cache/yum/x86_64/6/base/57de5f5b01f94e309bc9d2d
```

查询 / 目录下，所有 .txt 的文件

```
/usr/lib/python2.6/site-packages/mako-0.3.4-py2.6.e  
/usr/lib/python2.6/site-packages/Mako-0.3.4-py2.6.e  
/usr/lib/python2.6/site-packages/argparse-1.2.1-py2  
/usr/lib/python2.6/site-packages/argparse-1.2.1-py2  
/usr/lib/python2.6/site-packages/argparse-1.2.1-py2  
root@hadoop1 ~]# find / -name *.txt
```

仅将文本发送到当前选项卡

://192.168.184.129:22

9.6.2 locate 指令

locate 指令可以快速定位文件路径。locate 指令利用事先建立的系统中所有文件名称及路径的 locate 数据库实现快速定位给定的文件。Locate 指令无需遍历整个文件系统，查询速度较快。为了保证查询结果的准确度，管理员必须定期更新 locate 时刻。

- 基本语法

```
locate 搜索文件
```

- 特别说明

由于 locate 指令基于数据库进行查询，所以第一次运行前，必须使用 updatedb 指令创建 locate 数据库。

- 应用实例

案例 1：请使用 locate 指令快速定位 hello.txt 文件所在目录

```
/usr/lib/python2.6/site-packages/argparse-1.2.1-py2.6.e  
[root@hadoop1 ~]# updatedb → 创建locate数据库  
[root@hadoop1 ~]# locate hello.txt  
/home/hello.txt  
[root@hadoop1 ~]#
```

9.6.3 grep 指令和 管道符号 |

grep 过滤查找，管道符，“|”，表示将前一个命令的处理结果输出传递给后面的命令处理。

- 基本语法

```
grep [选项] 查找内容 源文件
```

- 常用选项

选项	功能
-n	显示匹配行及行号。
-i	忽略字母大小写

- 应用实例

案例 1：请在 hello.txt 文件中，查找 "yes" 所在行，并且显示行号

```
[root@hadoop1 ~]# cat hello.txt | grep -n yes  
4:yes  
7:yes  
[root@hadoop1 ~]# cat hello.txt | grep -ni yes  
4:yes  
7:yes  
9:Yes
```

区分大小写
不区分大小写

9.7 压缩和解压类

9.7.1 gzip/gunzip 指令

gzip 用于压缩文件，gunzip 用于解压的

- 基本语法

gzip 文件 (功能描述: 压缩文件, 只能将文件压缩为*.gz 文件)

gunzip 文件.gz (功能描述: 解压缩文件命令)

•应用实例

案例 1: gzip 压缩, 将 /home 下的 hello.txt 文件进行压缩

```
[root@hadoop1 home]# ls  
a1.txt a2.txt animal hello hello.txt info.txt  
[root@hadoop1 home]# gzip hello.txt  
[root@hadoop1 home]# ls  
a1.txt a2.txt animal hello hello.txt.gz info.txt  
[root@hadoop1 home]#
```

案例 2: gunzip 压缩, 将 /home 下的 hello.txt.gz 文件进行解压缩

```
[root@hadoop1 home]#  
[root@hadoop1 home]# gunzip hello.txt.gz  
[root@hadoop1 home]# ls  
.txt a2.txt animal hello hello.txt info.txt mycal mycat  
[root@hadoop1 home]#
```

细节说明

当我们使用 gzip 对文件进行压缩后, 不会保留原来的文件。

9.7.2 zip/unzip 指令

zip 用于压缩文件, unzip 用于解压的, 这个在项目打包发布中很有用的

•基本语法

zip [选项] XXX.zip 将要压缩的内容 (功能描述: 压缩文件和目录的命令)

unzip [选项] XXX.zip (功能描述: 解压缩文件)

•zip 常用选项

-r: 递归压缩, 即压缩目录

- unzip 的常用选项

- d<目录> : 指定解压后文件的存放目录

- 应用实例

- 案例 1: 将 /home 下的所有文件进行压缩成 mypackage.zip

```
[root@hadoop1 home]# zip -r mypackage.zip /home/
adding: home/ (stored 0%)
adding: home/xf/ (stored 0%)
adding: home/xf/.gnome2/ (stored 0%)
adding: home/xf/.mozilla/ (stored 0%)
.
.
.
.txt mypackage.zip t
```

- 案例 2: 将 mypackge.zip 解压到 /opt/tmp 目录下

```
[root@hadoop1 home]# 
[root@hadoop1 home]# unzip -d /opt/tmp/ mypackage.zip
Archive: mypackage.zip
creating: /opt/tmp/home/
creating: /opt/tmp/home/xf/
```

9.7.3 tar 指令

tar 指令 是打包指令，最后打包后的文件是 .tar.gz 的文件。

- 基本语法

- tar [选项] XXX.tar.gz 打包的内容 (功能描述: 打包目录, 压缩后的文件格式.tar.gz)

- 选项说明

选项	功能
-c	产生.tar打包文件
-v	显示详细信息
-f	指定压缩后的文件名
-z	打包同时压缩
-x	解包.tar文件

- 应用实例

案例 1：压缩多个文件，将 /home/a1.txt 和 /home/a2.txt 压缩成 a.tar.gz

```
[root@hadoop1 home]# ls  
al.txt a2.txt animal hello hello.txt info.txt mycal mydate.txt mypackage.  
[root@hadoop1 home]# tar -zcvf a.tar.gz a1.txt a2.txt  
al.txt  
a2.txt  
[root@hadoop1 home]# ls  
al.txt a2.txt animal a.tar.gz hello hello.txt info.txt mycal mydate.txt  
[root@hadoop1 home]#
```

对哪些文件打包 → 指向命令行参数 a1.txt 和 a2.txt
打包后的文件名 → 指向命令行参数 a.tar.gz

案例 2：将/home 的文件夹 压缩成 myhome.tar.gz

```
home/xm/.bashrc  
root@hadoop1 home]# ls  
1.txt a2.txt animal a.tar.gz hello hello.txt info  
root@hadoop1 home]# tar -zcvf myhome.tar.gz /home/  
仅将文本发送到当前选项卡
```

案例 3：将 a.tar.gz 解压到当前目录

```
[root@hadoop1 home]# rm a2.txt  
[root@hadoop1 home]# ls  
animal a.tar.gz hello hello.txt info.txt mycal mydate.  
[root@hadoop1 home]# tar -zxvf a.tar.gz  
al.txt  
a2.txt  
[root@hadoop1 home]# ls  
al.txt a2.txt animal a.tar.gz hello hello.txt info.txt
```

案例 4：将 myhome.tar.gz 解压到 /opt/ 目录下

```
root@hadoop1:~# cd home  
root@hadoop1 home]# ls  
.txt a2.txt animal a.tar.gz hello hello.txt info.txt mycal  
root@hadoop1 home]# tar -zxvf myhome.tar.gz -C /opt/
```

仅将文本发送到当前选项卡

指定解压到的目录事先要

指定解压到的那个目录，事先要存在才能成功，否则会报错。

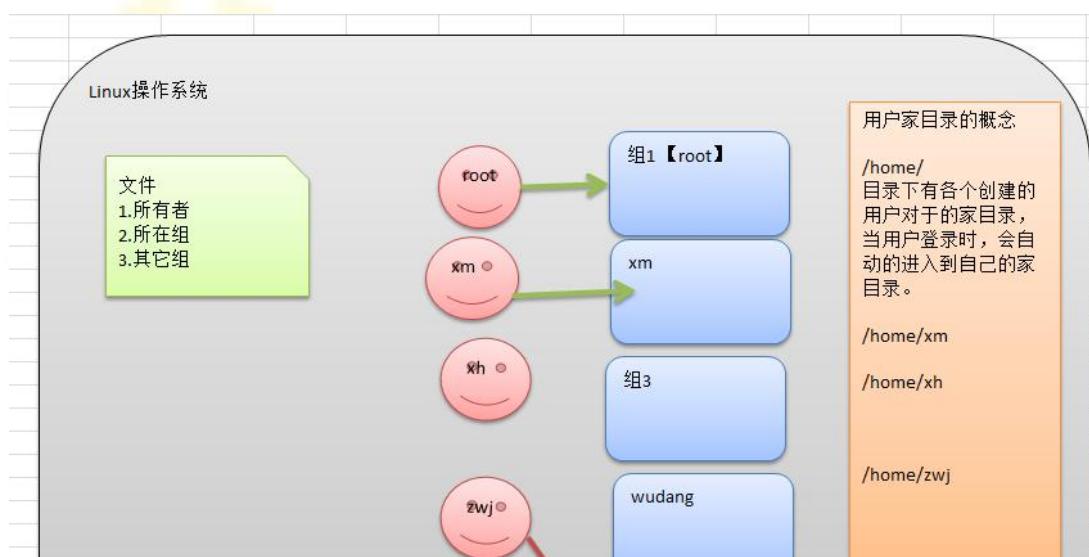


第 10 章 实操篇 组管理和权限管理

10.1 Linux 组基本介绍

在 linux 中的每个用户必须属于一个组，不能独立于组外。在 linux 中每个文件有所有者、所在组、其它组的概念。

- 1) 所有者
- 2) 所在组
- 3) 其它组
- 4) 改变用户所在的组



10.2 文件/目录 所有者

一般为文件的创建者,谁创建了该文件,就自然的成为该文件的所有者。

10.2.1 查看文件的所有者

- 1) 指令: ls -ahl
- 2) 应用实例: 创建一个组 police,再创建一个用户 tom,将 tom 放在 police 组 ,然后使用 tom 来创建一个文件 ok.txt,看看情况如何

```
[root@hadoop1 home]# groupadd police
[root@hadoop1 home]# useradd -g police tom
[root@hadoop1 home]# passwd tom
更改用户 tom 的密码。
新的 密码:
无效的密码: WAY 过短
无效的密码: 过于简单
重新输入新的 密码:
passwd: 所有的身份验证令牌已经成功更新。
[root@hadoop1 home]#
```

```
[tom@hadoop1 ~]$ pwd
/home/tom
[tom@hadoop1 ~]$ touch ok.txt
[tom@hadoop1 ~]$ ls -ahl
总用量 28K
drwx----- 4 tom  police 4.0K 3月 18 18:51 .
drwxr-xr-x. 9 root root 4.0K 3月 18 18:50 ..
-rw-r--r--. 1 tom  police 18 5月 11 2016 .bash_logout
-rw-r--r--. 1 tom  police 176 5月 11 2016 .bash_profile
-rw-r--r--. 1 tom  police 124 5月 11 2016 .bashrc
drwxr-xr-x. 2 tom  police 4.0K 11月 12 2010 .gnome2
drwxr-xr-x. 4 tom  police 4.0K 3月 17 21:07 .mozilla
-rw-r--r--. 1 tom  police 0 3月 18 18:51 ok.txt
```

这里就是显示ok.txt文件的所有者是tom

10.2.2 修改文件所有者

- 指令: chown 用户名 文件名

- 应用案例

要求: 使用 root 创建一个文件 apple.txt , 然后将其所有者修改成 tom

```
chown: 无效的用户: "apple.txt"
[root@hadoop1 home]# chown tom apple.txt
[root@hadoop1 home]# ls -l
总用量 56
-rw-r--r--. 1 root root      0 3月 18 17:45 a1.txt
-rw-r--r--. 1 root root      0 3月 18 17:45 a2.txt
drwxr-xr-x. 3 root root 4096 3月 18 2018 animal
-rw-r--r--. 1 tom  root      0 3月 18 18:57 apple.txt
-rw-r--r--. 1 root root    121 3月 18 18:10 a.tar.gz
-rw-r--r--. 1 root root      0 3月 18 01:06 hello
-rw-r--r--. 1 root root     54 3月 18 17:24 hello.txt
-rw-r--r--. 1 root root   744 3月 18 2018 info.txt
```

10.3组的创建

10.3.1 基本指令

groupadd 组名

10.3.2 应用实例:

创建一个组, ,monster

创建一个用户 fox , 并放入到 monster 组中

```
[root@hadoop1 home]#
[root@hadoop1 home]# groupadd monster
[root@hadoop1 home]# useradd -g monster fox
[root@hadoop1 home]# id fox
uid=505(fox) gid=506(monster) 组=506(monster)
[root@hadoop1 home]#
```

10.4文件/目录 所在组

当某个用户创建了一个文件后， 默认这个文件的所在组就是该用户所在的组。

10.4.1 查看文件/目录所在组

- 基本指令

```
ls -ahl
```

- 应用实例

10.4.2 修改文件所在的组

- 基本指令

```
chgrp 组名 文件名
```

- 应用实例

使用 root 用户创建文件 orange.txt ,看看当前这个文件属于哪个组, 然后将这个文件所在组, 修改到 police 组。

```
[root@hadoop1 ~]# chgrp police orange.txt
[root@hadoop1 ~]# ls -l
总用量 56
-rw-r--r--. 1 root root      0 三月 18 17:45 a1.txt
-rw-r--r--. 1 root root      0 三月 18 17:45 a2.txt
drwxr-xr-x. 3 root root 4096 三月 18 2018 animal
-rw-r--r--. 1 tom  root      0 三月 18 18:57 apple.txt
-rw-r--r--. 1 root  root    121 三月 18 18:10 a.tar.gz
-rw-r--r--. 1 root  root      0 三月 18 01:06 hello
-rw-r--r--. 1 root  root     54 三月 18 17:24 hello.txt
-rw-r--r--. 1 root  root    744 三月 18 2018 info.txt
-rw-r--r--. 1 root  root    785 三月 18 2018 mycal
-rw-r--r--. 1 root  root   1011 三月 18 2018 mydate.txt
-rw-r--r--. 1 root  root  8992 三月 18 18:04 mypackage.zip
-rw-r--r--. 1 root  police    0 三月 18 19:04 orange.txt
drwxr-xr-x. 2 root  root 4096 三月 18 2018 test
drwxr-xr-x. 2 root  root 4096 三月 18 04:12 tiger
drwx----- 4 xf   xf   4096 三月 18 04:45 xf
drwx----- 4 500  500 4096 三月 18 04:08 xm
drwx----- 5 zwj  wudang 4096 三月 18 16:39 zwj
[root@hadoop1 ~]#
```

10.5 其它组

除文件的所有者和所在组的用户外, 系统的其它用户都是文件的其它组.

10.6 改变用户所在组

在添加用户时，可以指定将该用户添加到哪个组中，同样的用 root 的管理权限可以改变某个用户所在的组。

10.6.1 改变用户所在组

- 1) usermod -g 组名 用户名
- 2) usermod -d 目录名 用户名 改变该用户登陆的初始目录。

10.6.2 应用实例

创建一个土匪组 (bandit) 将 tom 这个用户从原来所在的 police 组，修改到 bandit(土匪) 组

```
[root@hadoop1 home]#  
[root@hadoop1 home]# id tom  
uid=504(tom) gid=505(police) 组=505(police)  
[root@hadoop1 home]# groupadd bandit  
[root@hadoop1 home]# chmod -g bandit tom  
chmod: 无法访问"bandit": 没有那个文件或目录  
chmod: 无法访问"tom": 没有那个文件或目录  
[root@hadoop1 home]# usermod -g bandit tom  
[root@hadoop1 home]# id tom  
uid=504(tom) gid=507(bandit) 组=507(bandit)  
[root@hadoop1 home]#
```

表示将tom修改到bandit组

10.7 权限的基本介绍

ls -l 中显示的内容如下：

-rwxrw-r-- 1 root root 1213 Feb 2 09:39 abc

0-9 位说明

- 1) 第 0 位确定文件类型(d, -, l, c, b)
- 2) 第 1-3 位确定所有者（该文件的所有者）拥有该文件的权限。---User
- 3) 第 4-6 位确定所属组（同用户组的）拥有该文件的权限，---Group
- 4) 第 7-9 位确定其他用户拥有该文件的权限 ---Other

```

-rw-r--r--. 1 root root    744 3月 18 2018 info.txt 表示文件其它组的用户的权限r--
-rw-r--r--. 1 root root    785 3月 18 2018 mycal
-rw-r--r--. 1 root root 1011 3月 18 2018 mydate.txt
-rw-r--r--. 1 root root 4796 3月 18 18:13 myhome.tar.gz
-rw-r--r--. 1 root root 3992 3月 18 18:04 mypackage.zip
drwxr-xr-x. 2 root root 4096 3月 18 2018 test      如果是文件，表示文件大小
drwxr-xr-x. 2 root root 4096 3月 18 04:12 tiger    硬链接的数，如果是目录则表示该目录的子目录个数
drwx-----. 4 tom police 4096 3月 18 19:24 tom
drwx-----. 4 xf xf     4096 3月 18 04:45 xf
drwx-----. 4 500 500   4096 3月 18 04:08 xm
drwx-----. 5 zwj wudang 4096 3月 18 16:39 zwj
[tom@hadoop1 ~]$ cd tom/
[tom@hadoop2 ~]$ ls          文件的类型: - 普通文件 d:目录
                               1: 软链接 c: 字符设备【键盘，鼠标】
                               b:块文件，硬盘
ok.txt                         表示文件所有者权限rw
[tom@hadoop1 ~]$ ls -l
总用量 4
-rw-r--r--. 1 tom police 6 3月 18 19:24 ok.txt       文件名
[tom@hadoop1 ~]$                                         文件所在组的用户的权限r--,只有读权限

```

10.8 rwx 权限详解

10.8.1 rwx 作用到文件

- 1) [r] 代表可读(read): 可以读取, 查看
- 2) [w] 代表可写(write): 可以修改, 但是不代表可以删除该文件, 删除一个文件的前提条件是对该文件所在的目录有写权限, 才能删除该文件.
- 3) [x] 代表可执行(execute): 可以被执行

10.8.2 rwx 作用到目录

- 1) [r] 代表可读(read): 可以读取, ls 查看目录内容
- 2) [w] 代表可写(write): 可以修改, 目录内创建+删除+重命名目录
- 3) [x] 代表可执行(execute): 可以进入该目录

10.9 文件及目录权限实际案例

ls -l 中显示的内容如下: (记住)

-rwxrw-r-- 1 root root 1213 Feb 2 09:39 abc

10 个字符确定不同用户能对文件干什么

第一个字符代表文件类型： 文件 (-), 目录(d), 链接(l)

其余字符每 3 个一组(rwx) 读(r) 写(w) 执行(x)

第一组 rwx：文件拥有者的权限是读、写和执行

第二组 rw-：与文件拥有者同一组的用户的权限是读、写但不能执行

第三组 r--：不与文件拥有者同组的其他用户的权限是读不能写和执行

可用数字表示为: r=4,w=2,x=1 因此 rwx=4+2+1=7

1 文件：硬连接数或 目录：子目录数

root 用户

root 组

1213 文件大小(字节), 如果是文件夹, 显示 4096 字节

Feb 2 09:39 最后修改日期

abc 文件名

10.10 修改权限-chmod

10.10.1 基本说明：

通过 chmod 指令，可以修改文件或者目录的权限

10.10.2 第一种方式：+、-、= 变更权限

u:所有者 g:所有组 o:其他人 a:所有人(u、g、o 的总和)

1) chmod u=rwx,g=rx,o=x 文件目录名

2) chmod o+w 文件目录名

3) chmod a-x 文件目录名

• 案例演示

1) 给 abc 文件 的所有者读写执行的权限，给所在组读执行权限，给其它组读执行权限。

```
[tom@hadoop1 ~]$ chmod u=rwx,g=rx,o=rx abc  
[tom@hadoop1 ~]$ ls -l  
总用量 8  
-rwxr-xr-x. 1 tom police 0 3月 18 20:41 abc  
-rw-r--r--. 1 tom police 8 3月 18 20:32 ok2.txt  
-rw-r--r--. 1 tom police 9 3月 18 20:33 ok.txt  
[tom@hadoop1 ~]$
```

2) 给 abc 文件的所有者除去执行的权限，增加组写的权限

```
[tom@hadoop1 ~]$ chmod u-x,g+w abc  
[tom@hadoop1 ~]$ ls -l  
总用量 8  
-rw-rwrxr-x. 1 tom police 0 3月 18 20:41 abc  
-rw-r--r--. 1 tom police 8 3月 18 20:32 ok2.txt  
-rw-r--r--. 1 tom police 9 3月 18 20:33 ok.txt  
[tom@hadoop1 ~]$
```

3) 给 abc 文件的所有用户添加读的权限

```
[tom@hadoop1 ~]$ chmod a+r abc  
[tom@hadoop1 ~]$ ls -l  
总用量 8  
-rw-rwrxr-x. 1 tom police 0 3月 18 20:41 abc  
-rw-r--r--. 1 tom police 8 3月 18 20:32 ok2.txt  
-rw-r--r--. 1 tom police 9 3月 18 20:33 ok.txt  
[tom@hadoop1 ~]$
```

10.10.3 第二种方式：通过数字变更权限

规则： r=4 w=2 x=1 ,rwx=4+2+1=7

chmod u=rwx,g=rx,o=x 文件目录名

相当于 chmod 751 文件目录名

- 案例演示

要求：将 /home/abc.txt 文件的权限修改成 **rwxr-xr-x**，使用给数字的方式实现：

$rwx = 4+2+1 = 7$

$r-x = 4+1=5$

$r-x = 4+1 =5$

指令： chmod 755 /home/abc.txt

10.11 修改文件所有者-chown

10.11.1 基本介绍

chown newowner file 改变文件的所有者

chown newowner:newgroup file 改变用户的所有者和所有组

-R 如果是目录 则使其下所有子文件或目录递归生效

10.11.2 案例演示：

1) 请将 /home/abc.txt 文件的所有者修改成 tom

```
[tom@hadoop1 ~]$ chown tom abc.txt
[tom@hadoop1 ~]$ ls -l
总用量 8
-rw-rw-r-x. 1 tom police 0 3月 18 20:41 abc
-rw-r--r--. 1 tom police 0 3月 18 20:53 abc.txt
-rw-r--r--. 1 tom police 8 3月 18 20:32 ok2.txt
-rw-r--r--. 1 tom police 9 3月 18 20:33 ok.txt
[tom@hadoop1 ~]$
```

2) 请将 /home/kkk 目录下所有的文件和目录的所有者都修改成 tom

首选我们应该使用 root 操作。

```
[root@hadoop1 home]# chown -R tom kkk/
[root@hadoop1 home]# cd kkk/
[root@hadoop1 kkk]# ls -l
总用量 0
-rw-r--r--. 1 tom root 0 3月 18 20:55 a.txt
-rw-r--r--. 1 tom root 0 3月 18 20:55 b.txt
[root@hadoop1 kkk]#
```

将 kkk 目录下所有的文件，包括子目录和子目录下的所有文件，递归的将文件、目录的所有者改成 tom

10.12 修改文件所在组-chgrp

10.12.1 基本介绍

chgrp newgroup file 改变文件的所有组

10.12.2 案例演示：

- 1) 请将 /home/abc .txt 文件的所在组修改成 bandit (土匪)

```
chgrp bandit /home/abc.txt
```

- 2) 请将 /home/kkk 目录下所有的文件和目录的所在组都修改成 bandit(土匪)

```
chgrp -R bandit /home/kkk
```

```
[root@hadoop1 home]# chgrp -R bandit kkk/
[root@hadoop1 home]# cd kkk/
[root@hadoop1 kkk]# ls -l
总用量 0
-rw-r--r--. 1 tom bandit 0 3月 18 20:55 a.txt
-rw-r--r--. 1 tom bandit 0 3月 18 20:55 b.txt
[root@hadoop1 kkk]#
```

10.13 最佳实践-警察和土匪游戏

police , bandit

jack, jerry: 警察

xh, xq: 土匪

(1) 创建组

```
bash> groupadd police
```

```
bash> groupadd bandit
```

(2) 创建用户

```
[root@hadoop1 kkk]# useradd -g police jack
[root@hadoop1 kkk]# useradd -g police jerry
[root@hadoop1 kkk]# useradd -g bandit xh
[root@hadoop1 kkk]# useradd -g bandit xq
[root@hadoop1 kkk]#
[root@hadoop1 kkk]# passwd jack
更改用户 jack 的密码。
新的 密码：
无效的密码： WAY 过短
```

(3) jack 创建一个文件，自己可以读写，本组人可以读，其它组没人任何权限

```
[jack@hadoop1 ~]$ chmod 640 jack01.txt
[jack@hadoop1 ~]$ ls -l
总用量 4
-rw-r-----. 1 jack police 6 3月 18 21:20 jack01.txt
[jack@hadoop1 ~]$
```

(4) jack 修改该文件，让其它组人可以读，本组人可以读写

```
[jack@hadoop1 ~]$
[jack@hadoop1 ~]$ chmod o=r,g=rw jack01.txt
[jack@hadoop1 ~]$ ls -l
总用量 4
-rw-rw-r--. 1 jack police 6 3月 18 21:20 jack01.txt
[jack@hadoop1 ~]$
```

给其它组的用户读r
给自己所在组的用户rw, 读写权限

(5) xh 投靠 警察，看看是否可以读写。

先用 root 修改 xh 的组：

```
[root@hadoop1 kkk]#
[root@hadoop1 kkk]#
[root@hadoop1 kkk]# usermod -g police xh
[root@hadoop1 kkk]#
```

使用 jack 给他的家目录 /home/jack 的所在组一个 rx 的权限

```
rwx----- 5 zwj wudana 4096 3月 18 16:31  
jack@hadoop1 home]$ chmod g=rx jack/  
jack@hadoop1 home]$ ls -l  
总用量 92
```

xh 需要重新注销在到 jack 目录就可以操作 jack 的文件

```
[xh@hadoop1 jack]$ ls -l  
总用量 4  
-rw-rw-r-- 1 jack police 6 3月 18 21:20 jack01.txt  
[xh@hadoop1 jack]$ vim jack01.txt → xh可以对 jack01.txt 进行读写  
[xh@hadoop1 jack]$
```

10.14 课后练习

练习文件权限管理[课堂练习]

建立两个组 (神仙,妖怪)

建立四个用户(唐僧,悟空, 八戒, 沙僧)

设置密码

把悟空, 八戒放入妖怪 唐僧 沙僧 在神仙

用悟空建立一个文件 (monkey.java 该文件要输出 i am monkey)

给八戒一个可以 r w 的权限

八戒修改 monkey.java 加入一句话(i am pig)

唐僧 沙僧 对该文件没有权限

把 沙僧 放入妖怪组

让沙僧 修改 该文件 monkey, 加入一句话 ("我是沙僧, 我是妖怪!");

10.15 课后练习题 2

- 1 用 root 登录, 建立用户 mycentos,自己设定密码
- 2.用 mycentos 登录, 在主目录下建立目录 test/t11/t1
- 3.在 t1 中建立一个文本文件 aa,用 vi 编辑其内容为 ls - al

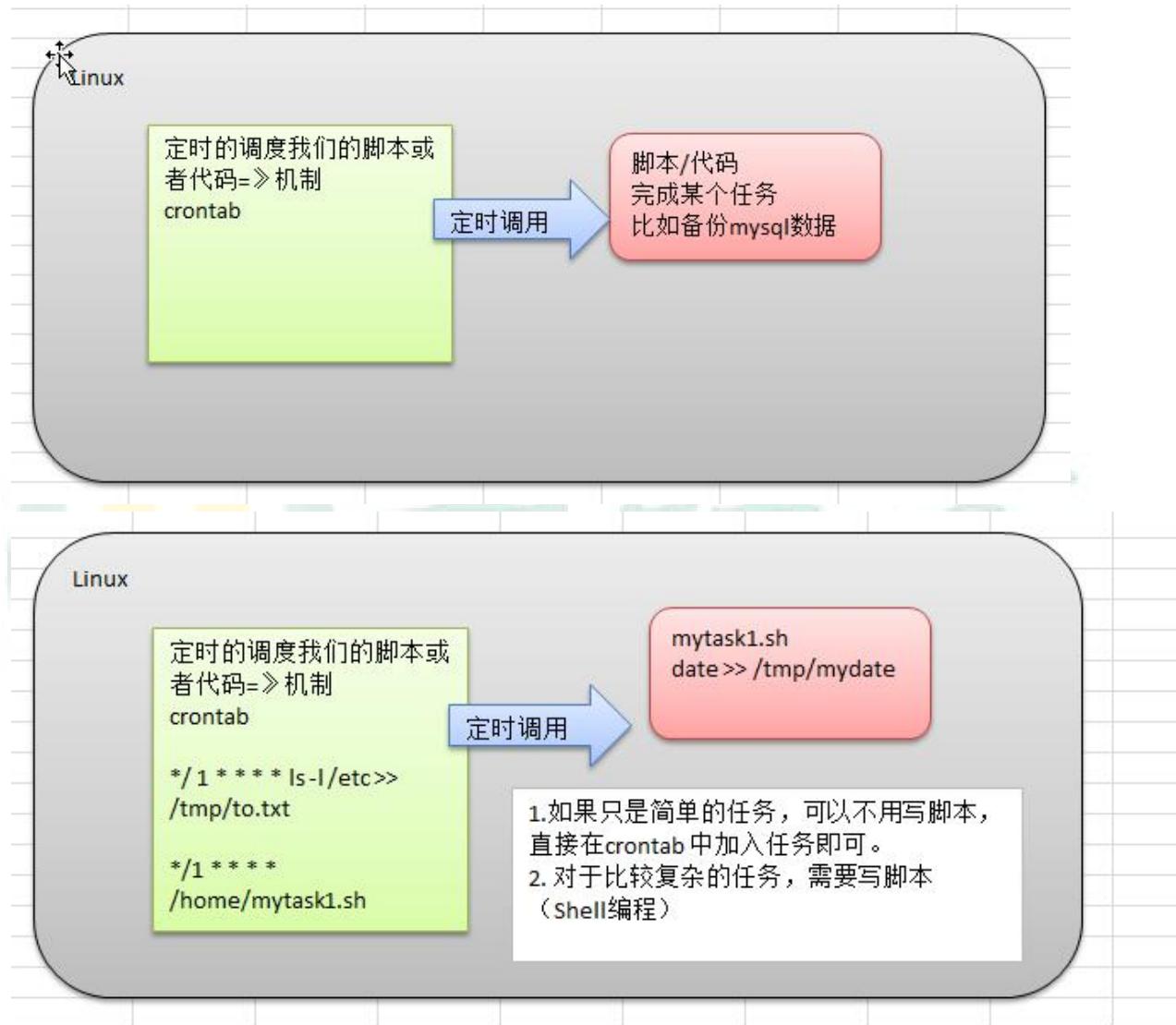


4. 改变 aa 的权限为可执行文件[可以将当前日期追加到一个文件],运行该文件./aa
5. 删除新建立的目录 test/t11/t1
6. 删除用户 mycentos 及其主目录中的内容
7. 将 linux 设置成进入到图形界面的
8. 重新启动 linux 或关机



第 11 章 实操篇 crond 任务调度

11.1 原理示意图



crontab 进行 定时任务的设置，。

11.2 概述

任务调度：是指系统在某个时间执行的特定的命令或程序。

任务调度分类：1. 系统工作：有些重要的工作必须周而复始地执行。如病毒扫描等

2.个别用户工作：个别用户可能希望执行某些程序，比如对 mysql 数据库的备份。

11.3 基本语法

crontab [选项]

11.3.1 常用选项

-e	编辑crontab定时任务
-l	查询crontab任务
-r	删除当前用户所有的crontab任务

11.4 快速入门

11.4.1 任务的要求

设置任务调度文件：/etc/crontab

设置个人任务调度。执行 crontab -e 命令。

接着输入任务到调度文件

如： */1 * * * * ls -l /etc > /tmp/to.txt

意思说每小时的每分钟执行 ls -l /etc > /tmp/to.txt 命令

11.4.2 步骤如下

- 1) cron -e
- 2) */1 * * * * ls -l /etc >> /tmp/to.txt
- 3) 当保存退出后就程序。
- 4) 在每一分钟都会自动的调用 ls -l /etc >> /tmp/to.txt

11.4.3 参数细节说明

• 5个占位符的说明

[

项目	含义	范围
第一个“*”	一小时当中的第几分钟	0-59
第二个“*”	一天当中的第几小时	0-23
第三个“*”	一个月当中的第几天	1-31
第四个“*”	一年当中的第几月	1-12
第五个“*”	一周当中的星期几	0-7 (0和7都代表星期日)

• 特殊符号的说明

特殊符号	含义
*	代表任何时间。比如第一个“*”就代表一小时中每分钟都执行一次的意思。
,	代表不连续的时间。比如“0 8,12,16 * * * 命令”，就代表在每天的8点0分，12点0分，16点0分都执行一次命令
-	代表连续的时间范围。比如“0 5 * * 1-6命令”，代表周一到周六的凌晨5点0分执行命令
/n	代表每隔多久执行一次。比如“/10 * * * * 命令”，代表每隔10分钟就执行一遍命令

• 特定时间执行任务案例

时间	含义
45 22 * * * 命令	在22点45分执行命令
0 17 * * 1 命令	每周1的17点0分执行命令
0 5 1,15 * * 命令	每月1号和15号的凌晨5点0分执行命令
0 40 * * 1-5 命令	每周一到周五的凌晨4点40分执行命令
0 */10 4 * * * 命令	每天的凌晨4点，每隔10分钟执行一次命令
0 0 1,15 * 1 命令	每月1号和15号，每周1的0点0分都会执行命令。注意：星期几和几号最好不要同时出现，因为他们定义的都是天。非常容易让管理员混乱。

11.5 任务调度的几个应用实例



11.5.1 案例 1：每隔 1 分钟，就将当前的日期信息，追加到 /tmp/mydate 文件中

1) 先编写一个文件 /home/mytask1.sh

```
date >> /tmp/mydate
```

2) 给 mytask1.sh 一个可以执行权限

```
chmod 744 /home/mytask1.sh
```

3) crontab -e

4) */1 * * * * /home/mytask1.sh

5) 成功

11.5.2 案例 2：每隔 1 分钟， 将当前日期和日历都追加到 /home/mycal 文件中

1) 先编写一个文件 /home/mytask2.sh

```
date >> /tmp/mycal
```

```
cal >> /tmp/mycal
```

2) 给 mytask1.sh 一个可以执行权限

```
chmod 744 /home/mytask2.sh
```

3) crontab -e

4) */1 * * * * /home/mytask2.sh

5) 成功

11.5.3 案例 3： 每天凌晨 2:00 将 mysql 数据库 testdb ， 备份到文件中 mydb.bak。

1) 先编写一个文件 /home/mytask3.sh

```
/usr/local/mysql/bin/mysqldump -u root -proot testdb > /tmp/mydb.bak
```

2) 给 mytask3.sh 一个可以执行权限

```
chmod 744 /home/mytask3.sh
```

- 3) crontab -e
- 4) 0 2 * * * /home/mytask3.sh
- 5) 成功

11.6crond 相关指令:

- 1) crontab -r: 终止任务调度。
- 2) crontab -l: 列出当前有那些任务调度
- 3) service crond restart [重启任务调度]



第 12 章 实操篇 Linux 磁盘分区、挂载

12.1 分区基础知识

12.1.1 分区的方式：

1) mbr 分区：

1. 最多支持四个主分区
2. 系统只能安装在主分区
3. 扩展分区要占一个主分区
4. MBR 最大只支持 2TB，但拥有最好的兼容性

2) gtp 分区：

1. 支持无限多个主分区（但操作系统可能限制，比如 windows 下最多 128 个分区）
2. 最大支持 18EB 的大容量（1EB=1024 PB, 1PB=1024 TB）
3. windows7 64 位以后支持 gtp

12.1.2 windows 下的磁盘分区



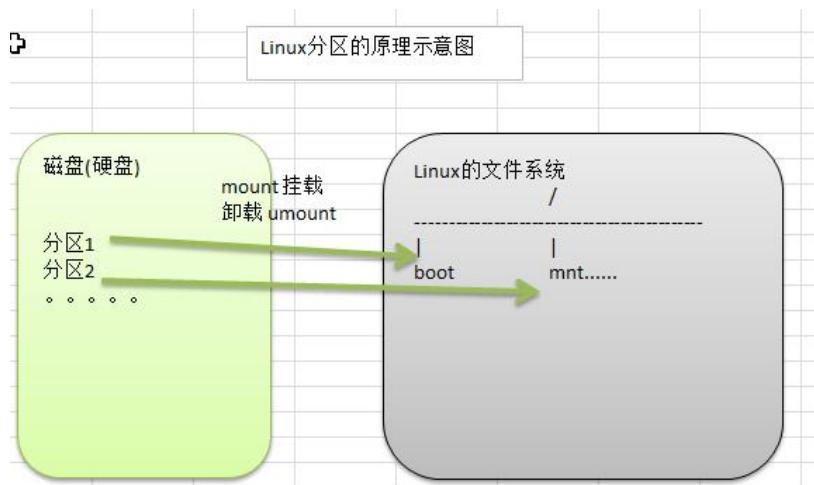
12.2 Linux 分区

12.2.1 原理介绍

1)Linux 来说无论有几个分区，分给哪一目录使用，它归根结底就只有一个根目录，一个独立且唯一的文件结构，Linux 中每个分区都是用来组成整个文件系统的一部分。

2)Linux 采用了一种叫“载入”的处理方法，它的整个文件系统中包含了一整套的文件和目录，且将一个分区和一个目录联系起来。这时要载入的一个分区将使它的存储空间在一个目录下获得。

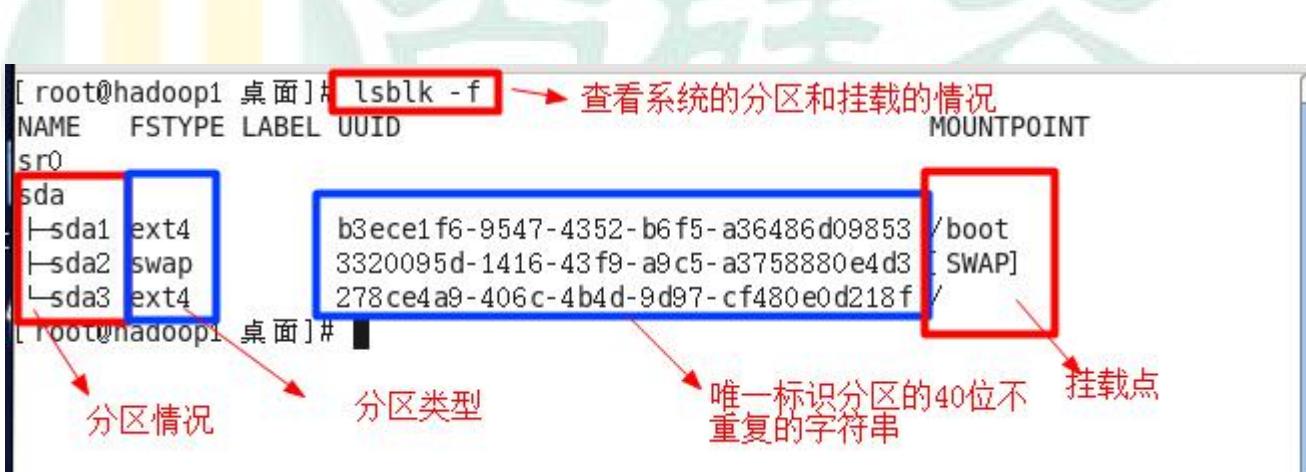
3)示意图



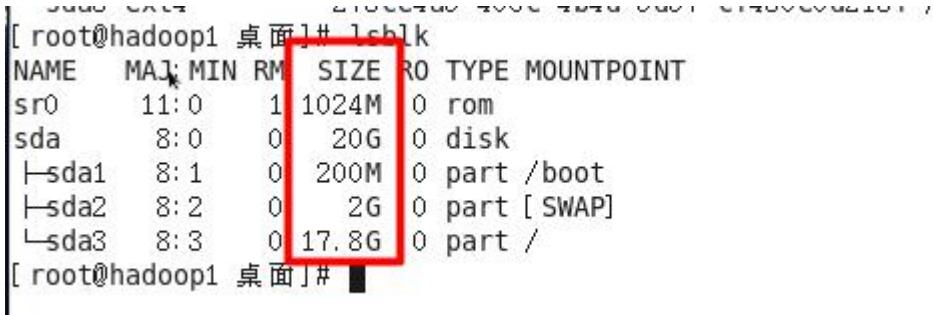
12.2.2 硬盘说明

- 1) Linux 硬盘分 IDE 硬盘和 SCSI 硬盘，目前基本上是 SCSI 硬盘
- 2) 对于 IDE 硬盘，驱动器标识符为“hdx~”，其中“hd”表明分区所在设备的类型，这里是指 IDE 硬盘了。“x”为盘号（a 为基本盘，b 为基本从属盘，c 为辅助主盘，d 为辅助从属盘），“~”代表分区，前四个分区用数字 1 到 4 表示，它们是主分区或扩展分区，从 5 开始就是逻辑分区。例，hda3 表示为第一个 IDE 硬盘上的第三个主分区或扩展分区，hdb2 表示为第二个 IDE 硬盘上的第二个主分区或扩展分区。
- 3) 对于 SCSI 硬盘则标识为“sdx~”，SCSI 硬盘是用“sd”来表示分区所在设备的类型的，其余则和 IDE 硬盘的表示方法一样。

12.2.3 使用 lsblk 指令查看当前系统的分区情况



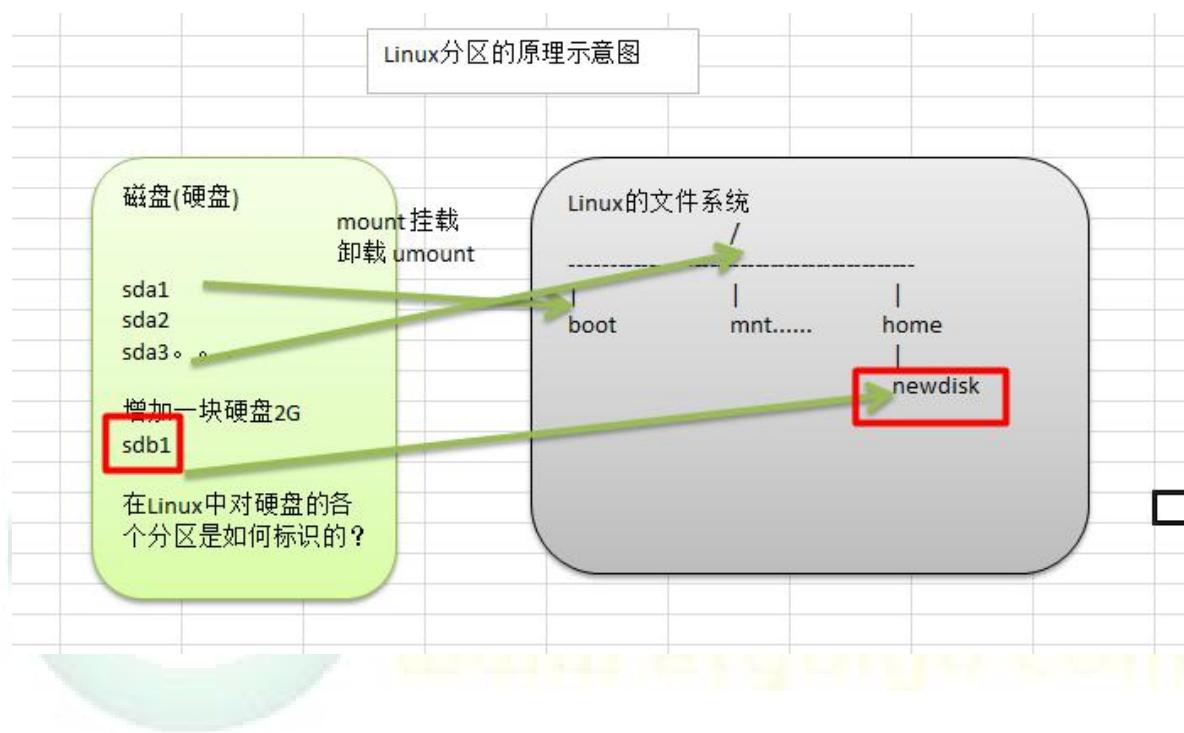
```
[root@hadoop1 桌面]# lsblk -f
NAME   FSTYPE LABEL UUID                                     MOUNTPOINT
sr0
└─sda
  ├─sda1 ext4   /boot  b3ece1f6-9547-4352-b6f5-a36486d09853
  ├─sda2 swap   [SWAP] 3320095d-1416-43f9-a9c5-a3758880e4d3
  └─sda3 ext4   /      278ce4a9-406c-4b4d-9d97-cf480e0d218f
```



```
[root@hadoop1 桌面]# lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sr0     11:0    1 1024M  0 rom
sda     8:0     0  20G  0 disk
└─sda1   8:1     0 200M  0 part /boot
└─sda2   8:2     0   2G  0 part [SWAP]
└─sda3   8:3     0 17.8G 0 part /
```

12.3 挂载的经典案例

需求是给我们的 Linux 系统增加一个新的硬盘，并且挂载到/home/newdisk



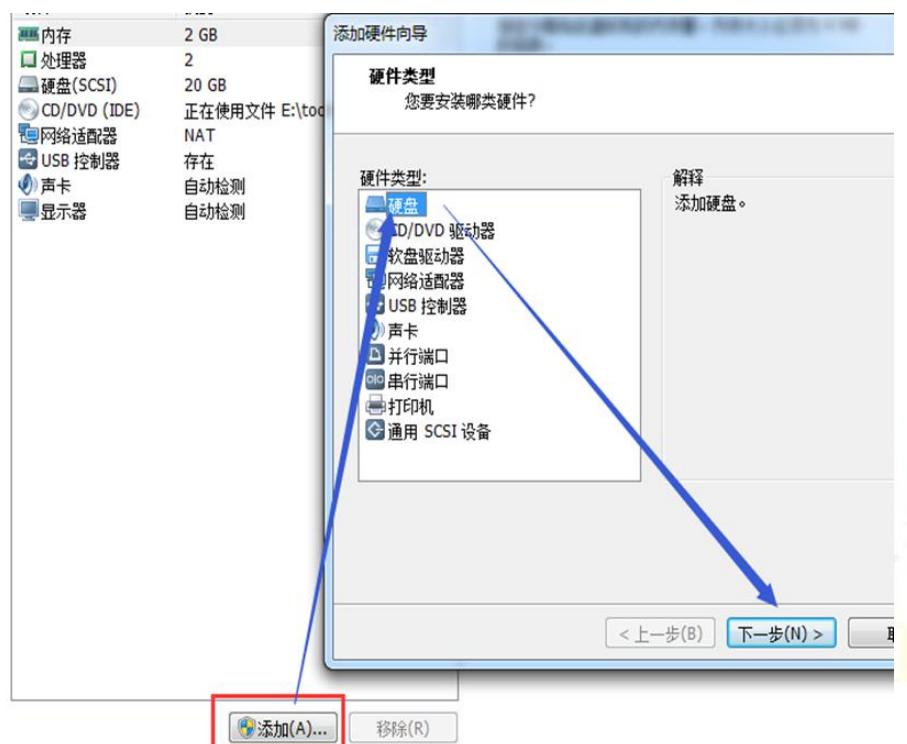
12.3.1 如何增加一块硬盘

- 1) 虚拟机添加硬盘
 - 2) 分区 fdisk /dev/sdb
 - 3) 格式化 mkfs -t ext4 /dev/sdb1
 - 4) 挂载 先创建一个 /home/newdisk ，挂载 mount /dev/sdb1 /home/newdisk
 - 5) 设置可以自动挂载(永久挂载，当你重启系统，仍然可以挂载到 /home/newdisk)。
vim /etc/fstab
- | | | | | |
|-----------|---------------|------|----------|-----|
| /dev/sdb1 | /home/newdisk | ext4 | defaults | 0 0 |
|-----------|---------------|------|----------|-----|

12.4 具体的操作步骤整理

12.4.1 虚拟机增加硬盘步骤 1

在【虚拟机】菜单中，选择【设置】，然后设备列表里添加硬盘，然后一路【下一步】，中间只有选择磁盘大小的地方需要修改，至到完成。然后重启系统（才能识别）！



12.4.2 虚拟机增加硬盘步骤 2

分区命令 fdisk /dev/sdb

开始对/sdb 分区

- m 显示命令列表
- p 显示磁盘分区 同 fdisk -l
- n 新增分区
- d 删除分区
- w 写入并退出

说明： 开始分区后输入 n，新增分区，然后选择 p，分区类型为主分区。两次回车默认剩余全

部空间。最后输入 w 写入分区并退出，若不保存退出输入 q。

```
Command (m for help): n
Command action
  e  extended
  p  primary partition (1-4)
p
Selected partition 4
First cylinder (2233-2610, default 2233): 回车, 默认全部剩余空间
Using default value 2233
Last cylinder, +cylinders or +size{K,M,G} (2233-2610, default 2610): 同上
Using default value 2610
```

12.4.3 虚拟机增加硬盘步骤 3

格式化磁盘

分区命令:mkfs -t ext4 /dev/sdb1

其中 ext4 是分区类型

12.4.4 虚拟机增加硬盘步骤 4

挂载: 将一个分区与一个目录联系起来,

•mount 设备名称 挂载目录

•例如: mount /dev/sdb1 /newdisk

•umount 设备名称 或者 挂载目录

•例如: umount /dev/sdb1 或者 umount /newdisk

12.4.5 虚拟机增加硬盘步骤 5

永久挂载: 通过修改/etc/fstab 实现挂载

添加完成后 执行 mount -a 即刻生效

```
# 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
#
/dev/sdb1          /home/newdisk      ext4    defaults        0 0
UUID=278ce4a9-106c-4b4d-9d07-cf480e0d218f /          ext4    defaults        1 1
UUID=b3ece1f6-9547-4352-b6f5-a36486d09853 /boot      ext4    defaults        1 2
UUID=3320095d-1416-43f9-a9c5-a3758880e4d3 swap      swap    defaults        0 0
tmpfs             /dev/shm           tmpfs   defaults        0 0
devpts            /dev/pts           devpts  gid=5, mode=620 0 0
sysfs             /sys               sysfs   defaults        0 0
proc              /proc              proc    defaults        0 0
```

12.5 磁盘情况查询

12.5.1 查询系统整体磁盘使用情况

基本语法

```
df -h
```

应用实例

查询系统整体磁盘使用情况

```
[root@hadoop1 ~]# df -lh
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda3       18G  3.3G  14G  20% /
tmpfs          996M 224K 996M   1% /dev/shm
/dev/sda1      190M  39M  142M  22% /boot
/dev/sdb1       2.0G  3.0M  1.9G   1% /home/newdisk
[root@hadoop1 ~]# umount /home/newdisk
[root@hadoop1 ~]# df -lh
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda3       18G  3.3G  14G  20% /
tmpfs          996M 224K 996M   1% /dev/shm
/dev/sda1      190M  39M  142M  22% /boot
[root@hadoop1 ~]#
```

12.5.2 查询指定目录的磁盘占用情况

• 基本语法

```
du -h /目录
```

查询指定目录的磁盘占用情况，默认为当前目录

-s 指定目录占用大小汇总

-h 带计量单位

-a 含文件

--max-depth=1 子目录深度

-c 列出明细的同时，增加汇总值

• 应用实例

查询 /opt 目录的磁盘占用情况，深度为 1

```
/dev/sda1      190M   39M  142M  22% /boot
[ root@hadoop1 home]# du -ach --max-depth=1 /opt
172K   /opt/home
4.0K    /opt/rh
214M   /opt/vmware-tools-distrib
152K   /opt/tmp
69M    /opt/VMwareTools-10.0.5-3228253.tar.gz
28M    /opt/金庸-射雕英雄传 txt精校版.txt
311M   /opt
311M   总用量
[ root@hadoop1 home]#
```

12.6 磁盘情况-工作实用指令

1) 统计/home 文件夹下文件的个数

```
drwx----- 4 500 500 4096 3月 18 04:08 xm
drwx----- 4 xq bandit 4096 3月 18 22:35 xq
drwx----- 5 zwj radang 4096 3月 18 16:30 zwj
[ root@hadoop1 home]# ls -l /home | grep "^-" | wc -l
12
[ root@hadoop1 home]#
```

2) 统计/home 文件夹下目录的个数

```
[ root@hadoop1 home]# ls -l /home | grep "^d" | wc -l
14
[ root@hadoop1 home]#
```

3) 统计/home 文件夹下文件的个数，包括子文件夹里的

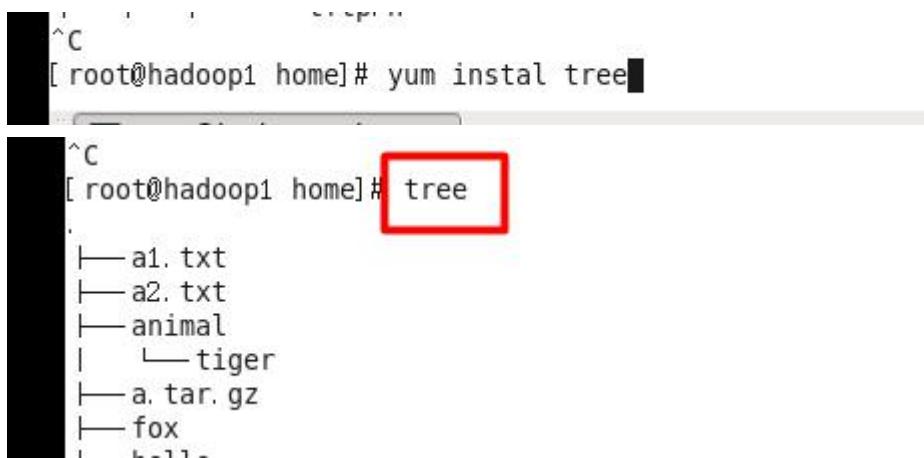
```
14
[ root@hadoop1 home]# ls -lR /home | grep "^-" | wc -l
25
[ root@hadoop1 home]#
```

4) 统计文件夹下目录的个数，包括子文件夹里的

```
[root@hadoop1 home]# ls -lR /home | grep "^\." | wc -l  
16  
[root@hadoop1 home]#
```

5) 以树状显示目录结构

```
[root@hadoop1 ~]# yum install tree  
[root@hadoop1 ~]# tree
```



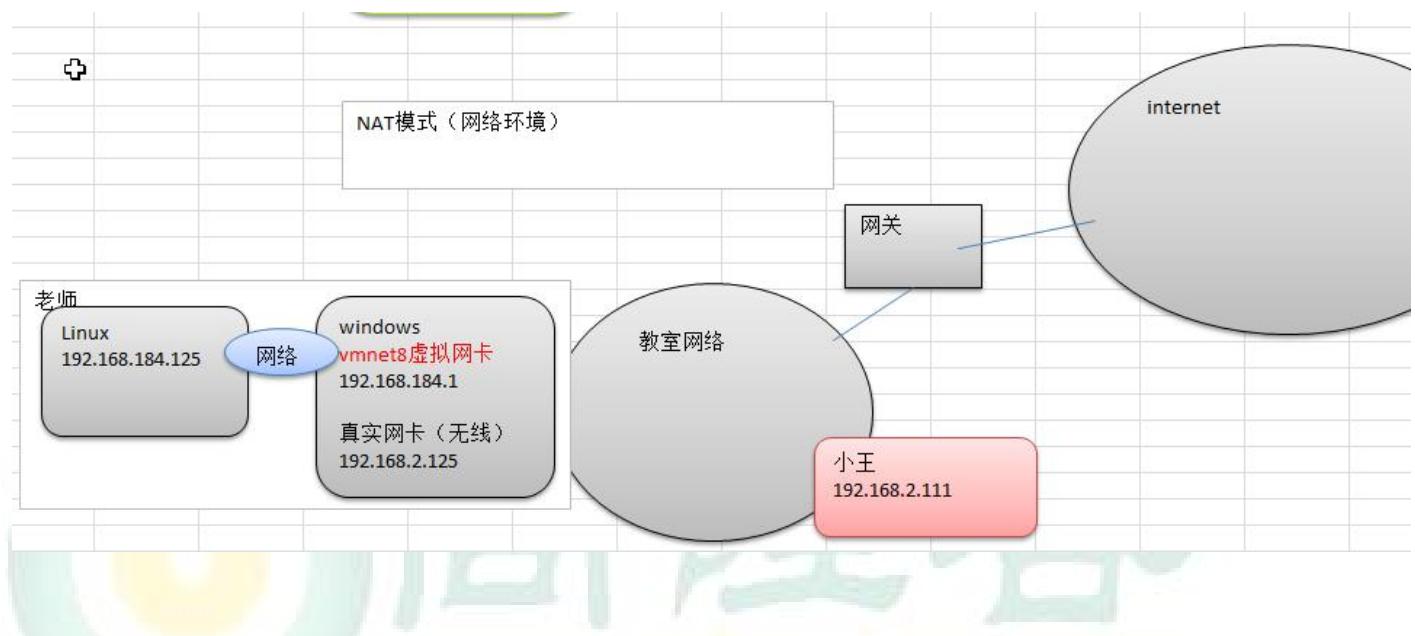
```
^C  
[root@hadoop1 ~]# tree  
  
[red box here]  
  
├── a1.txt  
├── a2.txt  
├── animal  
│   └── tiger  
├── a.tar.gz  
└── fox
```



第 13 章 实操篇 网络配置

13.1 Linux 网络配置原理图(含虚拟机)

目前我们的网络配置采用的是 NAT。

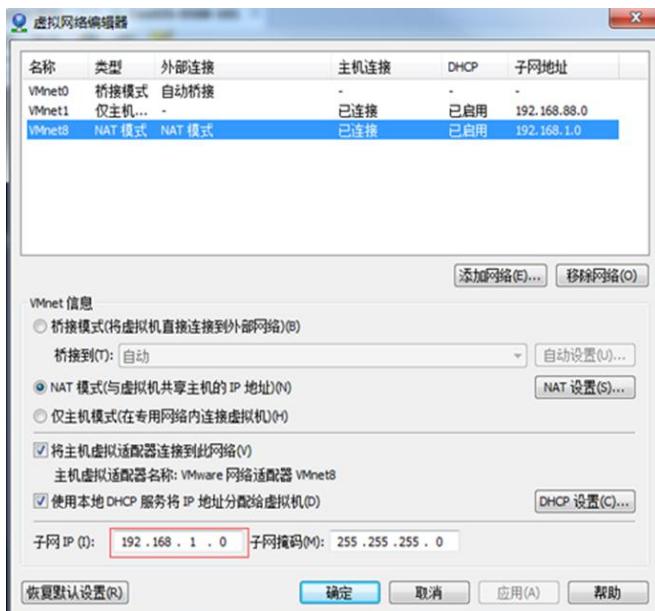


13.2 查看网络 IP 和网关

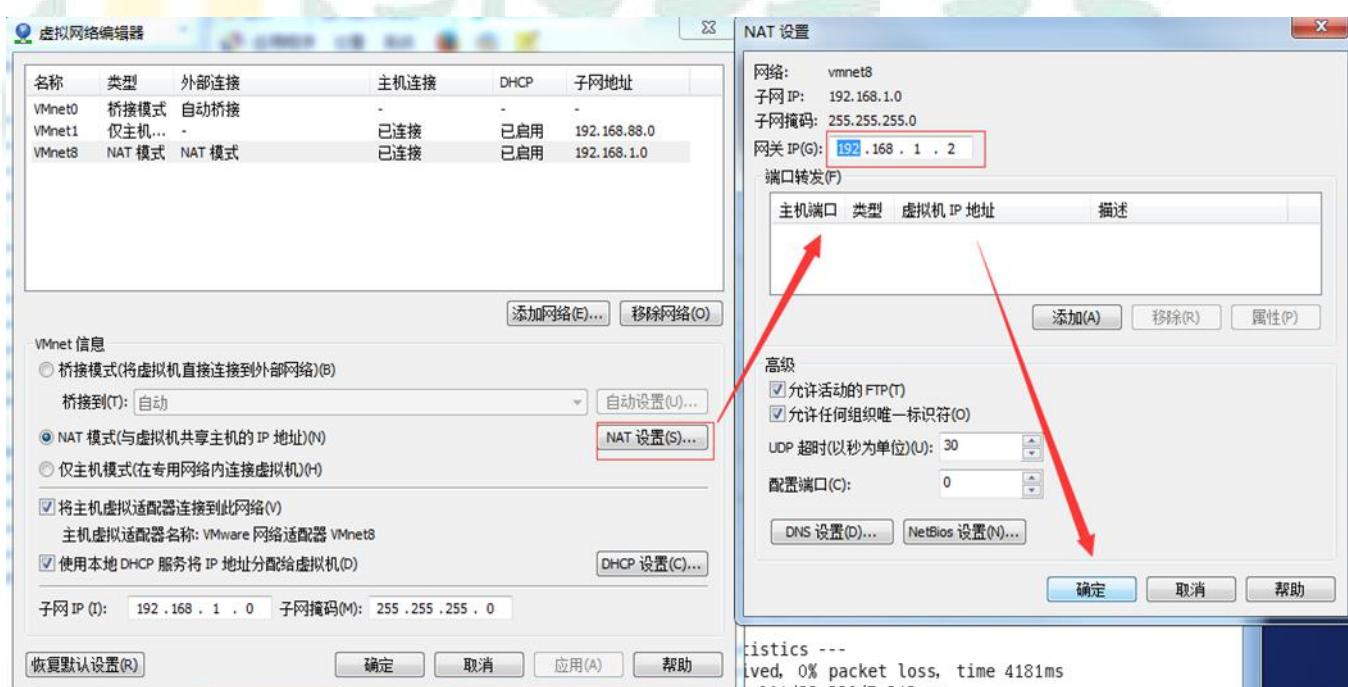
13.2.1 查看虚拟网络编辑器



13.2.2 修改 ip 地址(修改虚拟网络的 ip)

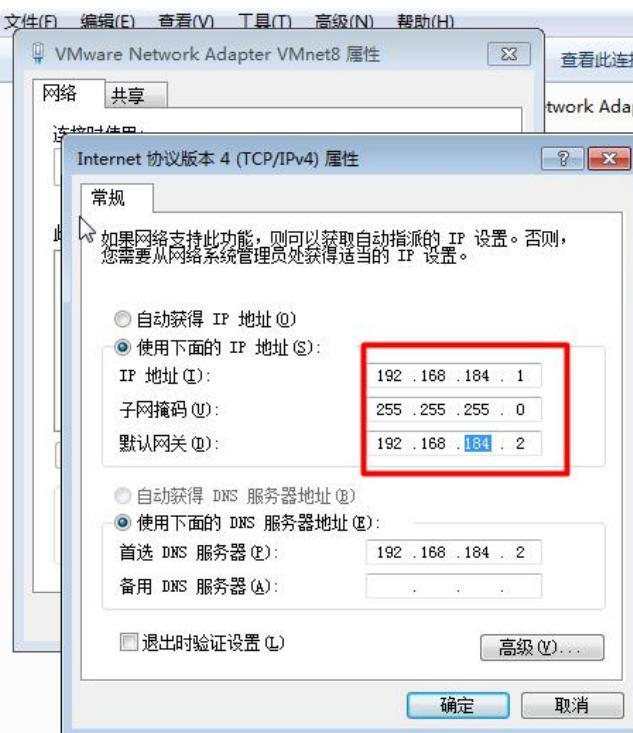


13.2.3 查看网关



13.2.4 查看 windows 环境的中 VMnet8 网络配置 (ipconfig 指令)

- 1) 使用 ipconfig 查看
- 2) 界面查看



13.3 ping 测试主机之间网络连通

13.3.1 基本语法

ping 目的主机 （功能描述：测试当前服务器是否可以连接目的主机）

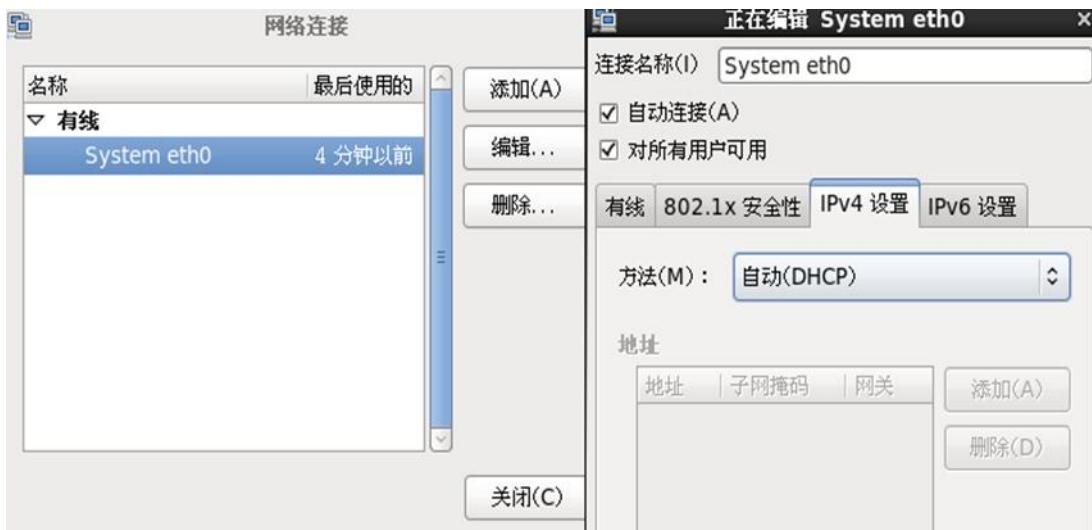
13.3.2 应用实例

测试当前服务器是否可以连接百度

```
[root@hadoop100 桌面]# ping www.baidu.com
```

13.4 linux 网络环境配置

13.4.1 第一种方法(自动获取)



缺点: linux 启动后会自动获取 IP, 缺点是每次自动获取的 ip 地址可能不一样。这个不适用于做服务器, 因为我们的服务器的 ip 需要是固定的。

13.4.2 第二种方法(指定固定的 ip)

说明

直接修改配置文件来指定 IP, 并可以连接到外网(程序员推荐), 编辑 vi
/etc/sysconfig/network-scripts/ifcfg-eth0

要求: 将 ip 地址配置的静态的, ip 地址为 192.168.184.130

```
DEVICE=eth0
TYPE=Ethernet
UUID=ff29a746-879b-4c89-953a-a8ca49987c06
ONBOOT=yes
root NM_CONTROLLED=yes
BOOTPROTO=static
DEFROUTE=yes
IPV4_FAILURE_FATAL=yes
IPV6INIT=no
NAME="System eth0"
IPADDR=192.168.184.130
GATEWAY=192.168.184.2
DNS1=192.168.184.2
PREFIX=24
HWADDR=00:0C:29:86:96:66
LAST_CONNECT=1521450896
~
```

启用boot配置成yes

以静态方式俩获取ip

指定ip

网关

dns和网关保持一致即可

修改后，一定要 重启服务

- 1) service network restart
- 2) reboot 重启系统

- ifcfg-eth0文件说明

```
I DEVICE=eth0      #接口名(设备,网卡)
HWADDR=00:0C:2x:6x:0x:xx #MAC地址
TYPE=Ethernet      #网络类型(通常是Ethemet)
UUID=926a57ba-92c6-4231-bacb-f27e5e6a9f44 #随机id
#系统启动的时候网络接口是否有效(yes/no)
ONBOOT=yes
#IP的配置方法[none|static|bootp|dhcp] (引导时不使用协议|静态分配IP|BOOTP协议|DHCP协议)
BOOTPROTO=static
#IP地址
IPADDR=192.168.184.130
#网关
GATEWAY=192.168.184.2
#域名解析器
DNS1=192.168.184.2
```

- 重启网络服务或者重启系统生效

service network restart 、 reboot

第 14 章 实操篇 进程管理

14.1 进程的基本介绍

- 1) 在 LINUX 中，每个执行的程序（代码）都称为一个进程。每一个进程都分配一个 ID 号。
- 2) 每一个进程，都会对应一个父进程，而这个父进程可以复制多个子进程。例如 www 服务器。
- 3) 每个进程都可能以两种方式存在的。前台与后台，所谓前台进程就是用户目前的屏幕上可以进行操作的。后台进程则是实际在操作，但由于屏幕上无法看到的进程，通常使用后台方式执行。
- 4) 一般系统的服务都是以后台进程的方式存在，而且都会常驻在系统中。直到关机才结束。

14.2 显示系统执行的进程

14.2.1 说明：

查看正在使用的指令是 `ps`, 一般来说使用的参数是 `ps -aux`

ps 显示的信息选项：

字段	说明
PID	进程识别号
TTY	终端机号
TIME	此进程所消 CPU 时间
CMD	正在执行的命令或进程名

`ps -a`: 显示当前终端的所有进程信息

`ps -u`: 以用户的格式显示进程信息

`ps -x`: 显示后台进程运行的参数

USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	COMMAND
root [root@hadoop1 ~]# ps -aux more										
root	1	0.1	0.0	19364	1552	?	Ss	17:18	0:05	/sbin/init
root	2	0.0	0.0	0	0	?	S	17:18	0:00	[kthread]
root	3	0.0	0.0	0	0	?	S	17:18	0:00	[migration/0]
root	4	0.0	0.0	0	0	?	S	17:18	0:00	[ksoftirqd/0]
root	5	0.0	0.0	0	0	?	S	17:18	0:00	[stopper/0]
root	6	0.0	0.0	0	0	?	S	17:18	0:00	[watchdog/0]
root	7	0.0	0.0	0	0	?	S	17:18	0:01	[migration/1]
root	8	0.0	0.0	0	0	?	S	17:18	0:00	[stopper/1]
root	9	0.0	0.0	0	0	?	S	17:18	0:00	[ksoftirqd/1]
root	10	0.0	0.0	0	0	?	S	17:18	0:00	[watchdog/1]
root	11	0.0	0.0	0	0	?	S	17:18	0:00	[migration/2]

14.2.2 ps 指令详解

- 1) 指令: ps -aux|grep xxx , 比如我看看有没有 sshd 服务
- 2) 指令说明
 - System V 展示风格
 - USER: 用户名称
 - PID: 进程号
 - %CPU: 进程占用 CPU 的百分比
 - %MEM: 进程占用物理内存的百分比
 - VSZ: 进程占用的虚拟内存大小 (单位: KB)
 - RSS: 进程占用的物理内存大小 (单位: KB)
 - TT: 终端名称, 缩写 .
 - STAT: 进程状态, 其中 S-睡眠, s-表示该进程是会话的先导进程, N-表示进程拥有比普通优先级更低的优先级, R-正在运行, D-短期等待, Z-僵死进程, T-被跟踪或者被停止等等
 - STARTED: 进程的启动时间
 - TIME: CPU 时间, 即进程使用 CPU 的总时间
 - COMMAND: 启动进程所用的命令和参数, 如果过长会被截断显示

14.2.3 应用实例

要求：以全格式显示当前所有的进程，查看进程的父进程。

```
[root@hadoop1 ~]# ps -ef | more
```

UID	PID	PPID	C	STIME	TTY	TIME	CMD
root	1	0	0	17:18	?	00:00:05	/sbin/init
root	2	0	0	17:18	?	00:00:00	[kthreadd]
root	3	2	0	17:18	?	00:00:00	[migration/0]
root	4	2	0	17:18	?	00:00:00	[ksoftirqd/0]
root	5	2	0	17:18	?	00:00:00	[stopper/0]
root	6	2	0	17:18	?	00:00:00	[watchdog/0]
root	7	2	0	17:18	?	00:00:01	[migration/1]

- ps -ef 是以全格式显示当前所有的进程
- -e 显示所有进程。-f 全格式。
- ps -ef|grep xxx
- 是 BSD 风格
- UID: 用户 ID
- PID: 进程 ID
- PPID: 父进程 ID
- C: CPU 用于计算执行优先级的因子。数值越大，表明进程是 CPU 密集型运算，执行优先级会降低；数值越小，表明进程是 I/O 密集型运算，执行优先级会提高
- STIME: 进程启动的时间
- TTY: 完整的终端名称
- TIME: CPU 时间
- CMD: 启动进程所用的命令和参数

思考题，如果我们希望查看 sshd 进程的父进程号是多少，应该怎样查询？

```
[root@hadoop1 ~]# ps -ef | grep sshd
root      3908      1  0 18:08 ?  I    00:00:00 /usr/sbin/sshd
root      3910  3908  0 18:08 ?  00:00:01 sshd: root@pts/1
root      4003  3914  0 18:44 pts/1  00:00:00 grep sshd
[root@hadoop1 ~]#
```

14.3 终止进程 kill 和 killall

14.3.1 介绍:

若是某个进程执行一半需要停止时，或是已消了很大的系统资源时，此时可以考虑停止该进程。使用 kill 命令来完成此项任务。

14.3.2 基本语法:

kill [选项] 进程号 (功能描述: 通过进程号杀死进程)

killall 进程名称 (功能描述: 通过进程名称杀死进程，也支持通配符，这在系统因负载过大而变得很慢时很有用)

14.3.3 常用选项:

-9 :表示强迫进程立即停止

14.3.4 最佳实践:

案例 1：踢掉某个非法登录用户

```
[root@hadoop1 桌面]# ps -aux | grep sshd
Warning: bad syntax, perhaps a bogus '-'? See /usr/share/doc/procps-3.2.8/FAQ
root      3908  0.0  0.0  66236  1200 ?          Ss   18:08  0:00 /usr/sbin/sshd
root      3910  0.0  0.2 102084  4156 ?          Ss   18:08  0:01 sshd: root@pts/1
jack     1010  0.6  0.0 102084  1844 ?          S    18:49  0:00 sshd: jack@pts/2
root      4041  2.0  0.0 103328   856 pts/0      S+   18:50  0:00 grep sshd
[root@hadoop1 桌面]# kill 4010
[root@hadoop1 桌面]#
```

案例 2：终止远程登录服务 sshd，在适当时候再次重启 sshd 服务

```
[root@hadoop1 桌面]# ps -aux | grep sshd
Warning: bad syntax, perhaps a bogus '-'? See /usr/share/doc/procps-3.2.8/FAQ
root      3908  0.0  0.0  66236  1200 ?          Ss   18:08  0:00 /usr/sbin/sshd
root      3910  0.0  0.2 102084  4156 ?          Ss   18:08  0:01 sshd: root@pts/1
jack     4010  0.6  0.0 102084  1844 ?          S    18:49  0:01 sshd: jack@pts/2
root      4041  2.0  0.0 103328   856 pts/0      S+   18:50  0:00 grep sshd
[root@hadoop1 桌面]# kill 4010
[root@hadoop1 桌面]#
[root@hadoop1 桌面]#
[root@hadoop1 桌面]# kill 3908
[root@hadoop1 桌面]# kill 3908
bash: kill: (3908) - 没有那个进程
[root@hadoop1 桌面]# ps -aux | grep sshd
Warning: bad syntax, perhaps a bogus '-'? See /usr/share/doc/procps-3.2.8/FAQ
root      3910  0.0  0.2 102084  4156 ?          Ss   18:08  0:01 sshd: root@pts/1
root      4043  6.0  0.0 103328   856 pts/0      S+   18:53  0:00 grep sshd
[root@hadoop1 桌面]#
```

kill后，就没有sshd服务

案例 3：终止多个 gedit 编辑器 【killall，通过进程名称来终止进程】



案例 4：强制杀掉一个终端

```
root      3914  0.0  0.0 108348  1828 pts/1    Ss+  18:08  0:00 /bin/bash
root      4090  0.0  0.0 108348  1820 pts/0    Ss+  18:57  0:00 /bin/bash
root      4115  0.3  0.0 108348  1816 pts/2    Ss   18:58  0:00 /bin/bash
root      4125  0.0  0.0 108348  848 pts/2    R+   18:58  0:00 grep bash
[ root@hadoop1 桌面]# kill 4090
[ root@hadoop1 桌面]# kill -9 4090
[ root@hadoop1 桌面]# ps aux | grep bash
Warning: bad syntax, perhaps a bogus '-'? See /usr/share/doc/procps-3.2.8/FAQ
root      3914  0.0  0.0 108348  1828 pts/1    Ss+  18:08  0:00 -bash
root      4115  0.0  0.0 108348  1828 pts/2    Ss   18:58  0:00 /bin/bash
root      4127  0.0  0.0 108348   692 pts/2    R+   18:59  0:00 /bin/bash
[ root@hadoop1 桌面]# kill -9 4115
```

14.4 查看进程树 pstree

14.4.1 基本语法:

`pstree [选项]`, 可以更加直观的来看进程信息

14.4.2 常用选项:

-p : 显示进程的 PID

-u : 显示进程的所属用户

14.4.3 应用实例:

案例 1: 请你树状的形式显示进程的 pid

```
[ root@hadoop1 桌面]# pstree -p
init(1)---ManagementAgent(1842)---{ManagementAgen}(1859)
          |           |
          |           +---{ManagementAgen}(1860)
          |
          +---NetworkManager(2195)
          +---VGAuthService(1730)
          +---abrt(2621)
          +---acpid(2274)
          +---atd(2648)
          +---auditd(2094)---{auditd}(2095)
```

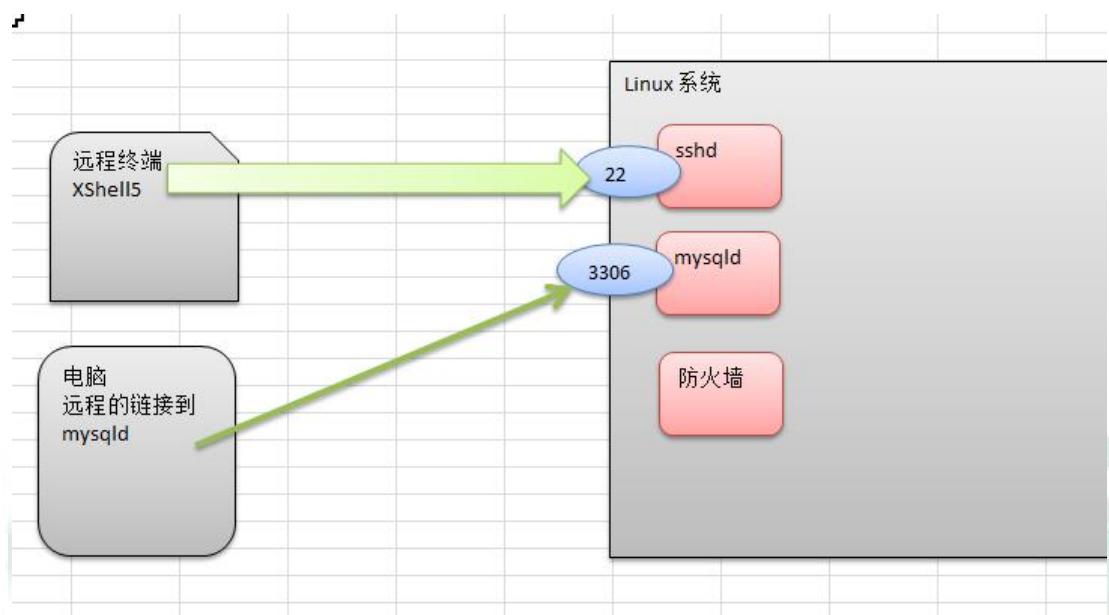
案例 2: 请你树状的形式进程的用户 id

`pstree -u` 即可。

14.5 服务(Service)管理

14.5.1 介绍:

服务(service) 本质就是进程，但是是运行在后台的，通常都会监听某个端口，等待其它程序的请求，比如(mysql , sshd 防火墙等)，因此我们又称为守护进程，是 Linux 中非常重要的知识点。【原理图】



14.5.2 service 管理指令:

service 服务名 [start | stop | restart | reload | status]

在 CentOS7.0 后 不再使用 service ,而是 systemctl

14.5.3 使用案例:

- 1) 查看当前防火墙的状况，关闭防火墙和重启防火墙。

```
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[ root@hadoop1 桌面]# service iptables status
表格 : filter
Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination
1    ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED, ESTABLISHED
2    ACCEPT     icmp --  0.0.0.0/0          0.0.0.0/0
3    ACCEPT     all  --  0.0.0.0/0          0.0.0.0/0
4    ACCEPT     tcp  --  0.0.0.0/0          0.0.0.0/0           state NEW tcp dpt:22
5    REJECT     all  --  0.0.0.0/0          0.0.0.0/0           reject-with icmp-host-prob
d

Chain FORWARD (policy ACCEPT)
num  target     prot opt source               destination
1    REJECT     all  --  0.0.0.0/0          0.0.0.0/0           reject-with icmp-host-prob
d

[ root@hadoop1 桌面]# service iptables stop
iptables: 将链设置为政策 ACCEPT: filter
[ 确定]
iptables: 清除防火墙规则 :
[ 确定]
iptables: 正在卸载模块 :
[ 确定]
[ root@hadoop1 桌面]# service iptables status
iptables: 未运行防火墙。
[ root@hadoop1 桌面]# service iptables start
iptables: 应用防火墙规则 :
[ 确定]
[ root@hadoop1 桌面]# service iptables status
本节 . filter
```

14.5.4 细节讨论：

- 1) 关闭或者启用防火墙后，立即生效。[telnet 测试 某个端口即可]

失去了跟主机的连接。

```
C:\Users\Administrator>telnet 192.168.184.130 22
```

- 2)这种方式只是临时生效，当重启系统后，还是回归以前对服务的设置。

如果希望设置某个服务自启动或关闭永久生效，要使用 chkconfig 指令，马上讲

14.5.5 查看服务名：

方式 1：使用 setup -> 系统服务 就可以看到。



方式 2: /etc/init.d/服务名称

```
[root@hadoop1 ~]# ls -l /etc/init.d/
```

列出系统有哪些服务

文件名	权限	拥有者	组	大小	修改时间	内容
abrt-ccpp	-rwxr-xr-x	1	root	root	1288	5月 12 2016
abrtd	-rwxr-xr-x	1	root	root	1628	5月 12 2016
abrt-oops	-rwxr-xr-x	1	root	root	1642	5月 12 2016
acpid	-rwxr-xr-x	1	root	root	1818	2月 17 2016

14.5.6 服务的运行级别(runlevel):

查看或者修改默认级别: vi /etc/inittab

Linux 系统有 7 种运行级别(runlevel): 常用的是级别 3 和 5

- 运行级别 0: 系统停机状态, 系统默认运行级别不能设为 0, 否则不能正常启动
- 运行级别 1: 单用户工作状态, root 权限, 用于系统维护, 禁止远程登陆
- 运行级别 2: 多用户状态(没有 NFS), 不支持网络
- 运行级别 3: 完全的多用户状态(有 NFS), 登陆后进入控制台命令行模式
- 运行级别 4: 系统未使用, 保留
- 运行级别 5: X11 控制台, 登录后进入图形 GUI 模式
- 运行级别 6: 系统正常关闭并重启, 默认运行级别不能设为 6, 否则不能正常启动

14.5.7 开机的流程说明



14.5.8 chkconfig 指令

介绍

通过 chkconfig 命令可以给每个服务的各个运行级别设置自启动/关闭

基本语法

1) 查看服务 chkconfig --list|grep xxx

```
[root@hadoop1 桌面]# root@hadoop1 桌面]# chkconfig --list
NetworkManager 0:关闭 1:关闭 2:启用 3:启用 4:启用 5:启用 6:?
abrt-ccpp 0:关闭 1:关闭 2:关闭 3:启用 4:关闭 5:启用 6:?
abrtd 0:关闭 1:关闭 2:关闭 3:启用 4:关闭 5:启用 6:?
acpid 0:关闭 1:关闭 2:启用 3:启用 4:启用 5:启用 6:?
```

```
wpa_supplicant 0:关闭 1:关闭 2:关闭 3:关闭 4:关闭 5:关闭 6:关闭
[root@hadoop1 桌面]# chkconfig --list | grep sshd
sshd 0:关闭 1:关闭 2:启用 3:启用 4:启用 5:启用 6:关闭
[root@hadoop1 桌面]#
```

2) chkconfig 服务名 --list

```
[root@hadoop1 桌面]# chkconfig iptables --list
iptables 0:关闭 1:关闭 2:启用 3:启用 4:启用 5:启用 6:关闭
[root@hadoop1 桌面]#
```

3) chkconfig --level 5 服务名 on/off

请将 sshd 服务在运行级别为 5 的情况下，不要自启动。

```
[root@hadoop1 桌面]# chkconfig --level 5 sshd off
[root@hadoop1 桌面]# chkconfig sshd --list
sshd      0:关闭  1:关闭  2:启用  3:启用  4:启用  5:关闭  6:关闭
[root@hadoop1 桌面]#
```

14.5.9 应用实例：

- 1) 案例 1： 请显示当前系统所有服务的各个运行级别的运行状态

```
bash> chkconfig --list
```

- 2) 案例 2： 请查看 sshd 服务的运行状态

```
bash> service sshd status
```

- 3) 案例 3： 将 sshd 服务在运行级别 5 下设置为不自动启动，看看有什么效果？

```
bash> chkconfig --level 5 sshd off
```

- 4) 案例 4： 当运行级别为 5 时，关闭防火墙。

```
bash> chkconfig --level 5 iptables off
```

- 5) 案例 5： 在所有运行级别下，关闭防火墙

```
bash> chkconfig iptables off
```

- 6) 案例 6： 在所有运行级别下，开启防火墙

```
bash> chkconfig iptables on
```

14.5.10 • 使用细节

- 1) chkconfig 重新设置服务后自启动或关闭，需要重启机器 reboot 才能生效。

14.6 动态监控进程

14.6.1 介绍：

top 与 ps 命令很相似。它们都用来显示正在执行的进程。Top 与 ps 最大的不同之处，在于 top 在执行一段时间可以更新正在运行的的进程。

14.6.2 基本语法:

top [选项]

14.6.3 选项说明:

选项	功能
-d 秒数	指定top命令每隔几秒更新，默认是3秒在top命令的交互模式当中可以执行的命令：
-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。

交互操作说明:

操作	功能
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

14.6.4 应用实例:

案例 1. 监视特定用户

top: 输入此命令，按回车键，查看执行的进程。

u: 然后输入“u”回车，再输入用户名，即可

```

root@hadoop1:~# top
当前时间          运行时间
top 20:51:22 up 3:32, 3 users, load average: 0.35, 0.24, 0.30
Tasks: 204 total, 1 running, 203 sleeping, 0 stopped, 0 zombie
Cpu(s): 0.8%us, 0.5%sy, 0.0%ni, 98.7%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Mem: 2038376k total, 783520k used, 1254856k free, 29636k buffers
Swap: 2097148k total, 0k used, 2097148k free, 389404k cached

PID USER      PR  NI    VIRT   RES   SHR S %CPU %MEM     TIME+   COMMAND
2695 root      20   0 220m  35m 12m S 6.6  1.8  2:54.04 xorg
3259 root      20   0 561m 17m 11m S 2.7  0.9  0:03.46 clock-applet
4165 root      20   0 340m 16m 11m S 2.0  0.8  0:33.42 gnome-terminal
1675 root      20   0 163m 7836 4480 S 1.0  0.4  0:38.58 vmtoolsd
4839 root      20   0 15136 1308 936 R 0.7  0.1  0:00.56 top
 20 root      20   0 0 0 0 S 0.3  0.0  0:02.66 events/1
 21 root      20   0 0 0 0 S 0.3  0.0  0:06.54 events/2
1842 root      20   0 203m 7380 4428 S 0.3  0.4  0:11.74 ManagementAgent

```

案例 2：终止指定的进程。

top: 输入此命令，按回车键，查看执行的进程。

k: 然后输入“k”回车，再输入要结束的进程 ID 号

```

top - 20:56:08 up 3:37, 3 users, load average: 0
Tasks: 204 total, 1 running, 203 sleeping, 0 st
Cpu(s): 1.9%us, 1.6%sy, 0.0%ni, 96.4%id, 0.0%wa
Mem: 2038376k total, 783876k used, 1254500k fr
Swap: 2097148k total, 0k used, 2097148k fr

PID to kill:
PID USER      PR  NI    VIRT   RES   SHR S %CPU %MEM
2695 root      20   0 220m  35m 12m S 13.6  1.8
3012 root      9 -11 439m 5148 3716 S 9.9  0.3
4165 root      20   0 340m 16m 11m S 8.6  0.8
 20 root      20   0 201m 10m 11m S 0.0  1.0

```

案例 3:指定系统状态更新的时间(每隔 10 秒自动更新，默认是 3 秒):

bash> top -d 10

14.6.5 查看系统网络情况 netstat(重要)

- 基本语法

```
netstat [选项]
```

```
netstat -anp
```

- 选项说明

- an 按一定顺序排列输出

- p 显示哪个进程在调用

- 应用案例

查看系统所有的网络服务

```
root@hadoop1 桌面]# netstat -anp | more
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address     State
am name
tcp        0      0 0.0.0.0:22              0.0.0.0:*          LISTEN
root@hadoop1 桌面]# netstat -anp | more
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address     State
am name
tcp        0      0 0.0.0.0:22              0.0.0.0:*          LISTEN
查看所有的网络服务
```

请查看服务名为 sshd 的服务的信息。

```
root@hadoop1 桌面]#
root@hadoop1 桌面]# netstat -anp | grep sshd
:cp      0      0 0.0.0.0:22              0.0.0.0:*          LISTEN      4366/sshd
:cp      0      0 192.168.184.130:22       192.168.184.1:52599 ESTABLISHED 3910/sshd
:cp      0      0 ::1:22                  ::*:             LISTEN      4366/sshd
inx 2      [ ]      DGRAM
root@hadoop1 桌面]#
```

第 15 章 实操篇 RPM 和 YUM

15.1 rpm 包的管理

15.1.1 介绍：

一种用于互联网下载包的打包及安装工具，它包含在某些 Linux 分发版中。它生成具有.RPM 扩展名的文件。RPM 是 RedHat Package Manager (RedHat 软件包管理工具) 的缩写，类似 windows 的 setup.exe，这一文件格式名称虽然打上了 RedHat 的标志，但理念是通用的。

Linux 的分发版本都有采用 (suse,redhat, centos 等等)，可以算是公认的标准了。

15.1.2 rpm 包的简单查询指令：

查询已安装的 rpm 列表 rpm -qa|grep xx

请查询看一下，当前的 Linux 有没有安装 firefox .



```
[root@hadoop1 桌面]# rpm -qa | grep firefox
firefox-45.0.1-1.el6.centos.x86_64
[root@hadoop1 桌面]#
```

软件名 版本 在适用于centos 6.x 的64系统

15.1.3 rpm 包名基本格式：

一个 rpm 包名：firefox-45.0.1-1.el6.centos.x86_64.rpm

名称:firefox

版本号：45.0.1-1

适用操作系统: el6.centos.x86_64

表示 centos6.x 的 64 位系统

如果是 i686、i386 表示 32 位系统，noarch 表示通用。。

15.1.4 rpm 包的其它查询指令：

rpm -qa :查询所安装的所有 rpm 软件包

rpm -qa | more [分页显示]

rpm -qa | grep X [rpm -qa | grep firefox]

```
gnome-mag-0.15.9-2.el6.x86_64
[root@hadoop1 桌面]# rpm -qa | grep jdk
[root@hadoop1 桌面]# rpm -qa | grep firefox
firefox-45.0.1-1.el6.centos.x86_64
[root@hadoop1 桌面]#
```

rpm -q 软件包名 :查询软件包是否安装

rpm -q firefox

rpm -qi 软件包名 : 查询软件包信息

```
firefox-45.0.1-1.el6.centos.x86_64
[ root@hadoop1 桌面]# rpm -qi firefox
Name        : firefox
Version     : 45.0.1
Release     : 1.el6.centos
时 49分04秒
Install Date: 2018年03月17日 星期六 21时17分02秒      Build Host: worker1.bsy
entos.org
Group       : Applications/Internet      Source RPM: firefox-45.0.1-1.el6.
tos.src.rpm
Size        : 133048560                  License: MPLv1.1 or GPLv2+ or
Lv2+
Signature   : RSA/SHA1, 2016年05月12日 星期四 18时50分20秒, Key ID 0946fca2c1
```

rpm -qi file

rpm -ql 软件包名 :查询软件包中的文件

rpm -ql firefox

```
Mozilla Firefox is an open-source web browser, designed for s
compliance, performance and portability.
[ root@hadoop1 桌面]# rpm -ql firefox
/etc/firefox/pref
/usr/bin/firefox
/usr/lib64/firefox
/usr/lib64/firefox/LICENSE
/usr/lib64/firefox/application.ini
/usr/lib64/firefox/browser/blocklist.xml
/usr/lib64/firefox/browser/chrome
```

rpm -qf 文件全路径名 查询文件所属的软件包

rpm -qf /etc/passwd

rpm -qf /root/install.log

```
bash: q: command not found
[ root@hadoop1 桌面]# rpm -qf /etc/passwd
setup-2.8.14-20.el6_4.1.noarch
[ root@hadoop1 桌面]#
```

查询某个文件属于
哪个rpm包

15.1.5 卸载 rpm 包：

- 基本语法

rpm -e RPM 包的名称

- 应用案例

- 1) 删除 firefox 软件包

```
# doop1 桌面]# rpm -qf /etc/passwd  
8.14-20.el6_4.1.noarch  
# doop1 桌面]# rpm -e firefox  
# doop1 桌面]#
```

- 细节问题

- 1) 如果其它软件包依赖于您要卸载的软件包，卸载时则会产生错误信息。

如： \$ rpm -e foo

removing these packages would break dependencies:foo is needed by bar-1.0-1

- 2) 如果我们就是要删除 foo 这个 rpm 包，可以增加参数 --nodeps ,就可以强制删除，但是一般不推荐这样做，因为依赖于该软件包的程序可能无法运行

如： \$ rpm -e --nodeps foo

带上 --nodeps 就是强制删除。

15.1.6 安装 rpm 包：

- 基本语法

rpm -ivh RPM 包全路径名称

- 参数说明

i=install 安装

v=verbose 提示

h=hash 进度条

- 应用实例

- 1) 演示安装 firefox 浏览器

步骤先找到 firefox 的安装 rpm 包,你需要挂载上我们安装 centos 的 iso 文件, 然后到/media/下去找 rpm 找。

```
cp firefox-45.0.1-1.el6.centos.x86_64.rpm /opt/
```

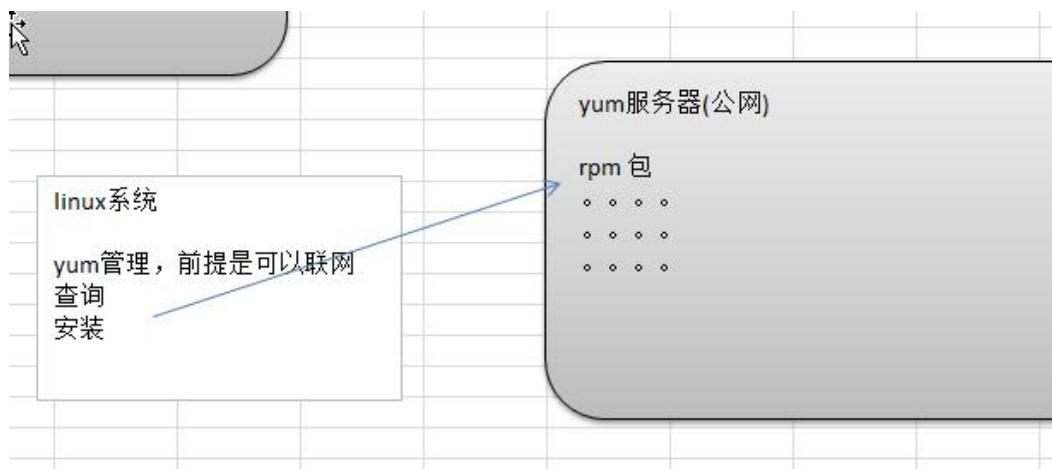
```
[root@hadoop1 Packages]# cp firefox-45.0.1-1.el6.centos.x86_64.rpm /opt/
[root@hadoop1 Packages]# cd /opt/
[root@hadoop1 opt]# ls
firefox-45.0.1-1.el6.centos.x86_64.rpm  VMwareTools-10.0.5-3228253.tar.gz
home                                         vmware-tools-distrib
rh                                           金庸 - 射雕英雄传 txt精校版.txt
tmp
[root@hadoop1 opt]# rpm -ivh firefox-45.0.1-1.el6.centos.x86_64.rpm
Preparing...                                           #### [100%]
 1:firefox                                         #### [100%]
[root@hadoop1 opt]#
[root@hadoop1 opt]#
```

OK

15.2 yum

15.2.1 介绍:

Yum 是一个 Shell 前端软件包管理器。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, 并且一次安装所有依赖的软件包。使用 yum 的前提是必须联网。



15.2.2 yum 的基本指令

- 查询 yum 服务器是否有需要安装的软件

 yum list|grep xx 软件列表

- 安装指定的 yum 包

 yum install xxx 下载安装

15.2.3 yum 应用实例：

案例：请使用 yum 的方式来安装 firefox

- 1) 先查看一下 firefox rpm 在 yum 服务器有没有

```
zsh-html.x86_64                               4.3.11-4.el6.centos
[root@hadoop1 opt]# yum list | grep firefox
firefox.x86_64                                45.0.1-1.el6.centos
da-CentOS-201605220104.x86_64/6.8
firefox.i686                                    52.7.0-1.el6.centos
firefox.x86_64                                52.7.0-1.el6.centos
[root@hadoop1 opt]#
```

- 2) 安装

 yum install firefox



会安装最新版本的软件。

成功！

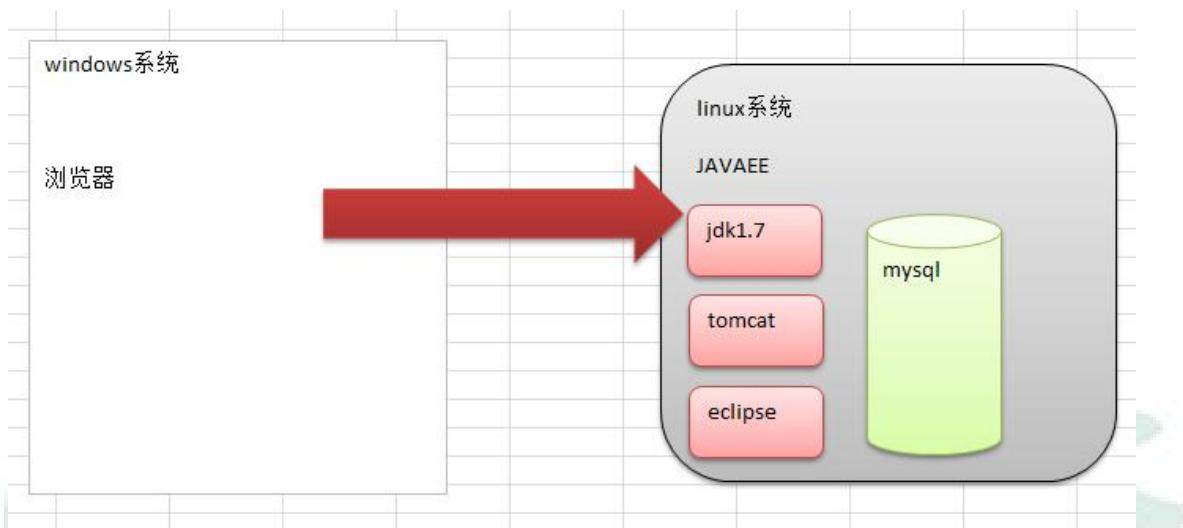
```
CentOS已安装:  
firefox.x86_64 0:52.7.0-1.el6.centos  
  
作为依赖被升级:  
nspr.x86_64 0:4.13.1-1.el6           nss.x86_  
nss-sysinit.x86_64 0:3.28.4-4.el6_9    nss-tool  
nss-util.x86_64 0:3.28.4-1.el6_9  
  
完毕!  
[root@hadoop1 opt]#
```



第 16 章 JavaEE 定制篇 搭建 JavaEE 环境

16.1 概述

16.1.1 示意图：



如果需要在 Linux 下进行 JavaEE 的开发，我们需要安装如下软件

- mysql-5.6.14.tar.gz
- jdk-7u79-linux-x64.gz
- eclipse-jee-mars-2-linux-gtk-x86_64.tar.gz
- apache-tomcat-7.0.70.tar.gz

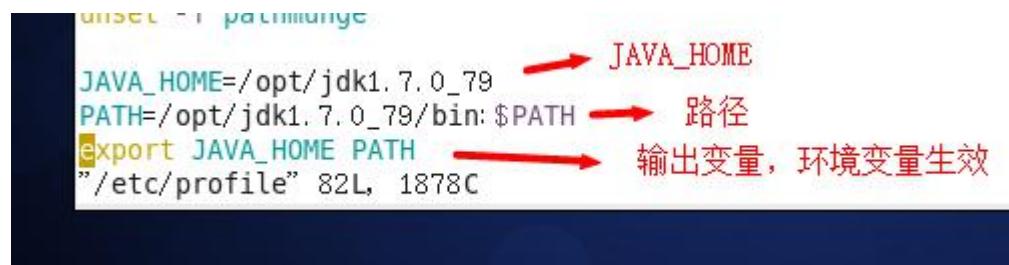
16.2 安装 JDK

16.2.1 看老师演示，一会整理笔记：

16.2.2 安装步骤

- 0) 先将软件通过 xftp5 上传到 /opt 下
- 1) 解压缩到 /opt

2) 配置环境变量的配置文件 vim /etc/profile



```
vim -i /etc/profile
JAVA_HOME=/opt/jdk1.7.0_79          → JAVA_HOME
PATH=/opt/jdk1.7.0_79/bin:$PATH      → 路径
export JAVA_HOME PATH                → 输出变量, 环境变量生效
"/etc/profile" 82L, 1878C
```

```
JAVA_HOME=/opt/jdk1.7.0_79
```

```
PATH=/opt/jdk1.7.0_79/bin:$PATH
```

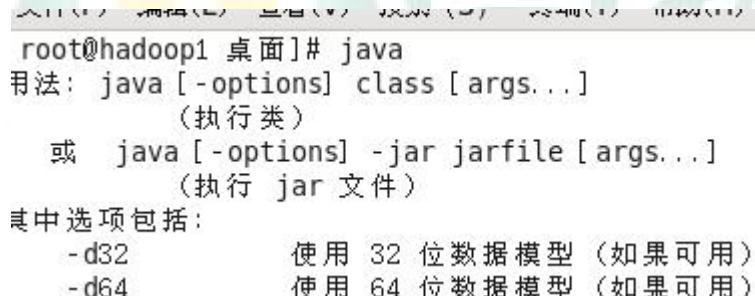
```
export JAVA_HOME PATH
```

3) 需要注销用户，环境变量才能生效。

如果是在 3 运行级别， logout

如果是在 5 运行级别，

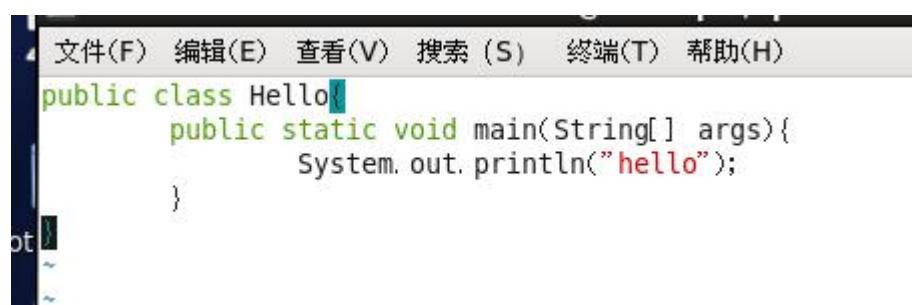
4) 在任何目录下就可以使用 java 和 javac



```
root@hadoop1 桌面]# java
用法: java [-options] class [args...]
          (执行类)
或   java [-options] -jar jarfile [args...]
          (执行 jar 文件)
其中选项包括:
-d32           使用 32 位数据模型 (如果可用)
-d64           使用 64 位数据模型 (如果可用)
```

16.2.3 测试是否安装成功

编写一个简单的 Hello.java 输出"hello,world!"



```
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
public class Hello{
    public static void main(String[] args){
        System.out.println("hello");
    }
}
```

```
hello  
[ root@hadoop1 opt]# javac Hello.java  
[ root@hadoop1 opt]# java Hello  
hello  
[ root@hadoop1 opt]#
```

16.3 安装 tomcat

16.3.1 步骤：

- 1) 解压缩到/opt

```
he-tomcat-7.0.70/webapps/manager/index.jsp  
he-tomcat-7.0.70/webapps/manager/status.xsd  
he-tomcat-7.0.70/webapps/manager/xform.xsl  
t@hadoop1 opt]# tar -zvxf apache-tomcat-7.0.70.tar.gz
```

- 2) 启动 tomcat ./startup.sh

先进入到 tomcat 的 bin 目录

```
[root@hadoop1 ~]# cd apache-tomcat-7.0.70/bin/  
[root@hadoop1 bin]# pwd  
/opt/apache-tomcat-7.0.70/bin  
[root@hadoop1 bin]# ls  
bootstrap.jar           daemon.sh          startup.sh  
catalina.bat             digest.bat        tomcat-juli.jar  
catalina.sh            digest.sh         tomcat-native.tar.gz  
catalina-tasks.xml       setclasspath.bat   tool-wrapper.bat  
commons-daemon.jar     setclasspath.sh    tool-wrapper.sh  
commons-daemon-native.tar.gz shutdown.bat    version.bat  
configtest.bat          shutdown.sh       version.sh  
configtest.sh           startup.bat
```

```
[root@hadoop1 bin]# ./startup.sh  
Using CATALINA_BASE: /opt/apache-tomcat-  
Using CATALINA_HOME: /opt/apache-tomcat-  
Using CATALINA_TMPDIR: /opt/apache-tomcat-  
Using JRE_HOME: /opt/jdk1.7.0_79
```

使用 Linux 本地的浏览器是可以访问到 tomcat

3) 开放端口 8080 ,这样外网才能访问到 tomcat

vim /etc/sysconfig/iptables

```
*filter
:INPUT ACCEPT [ 0:0]
:FORWARD ACCEPT [ 0:0]
:OUTPUT ACCEPT [ 0:0]
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 8080 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
COMMIT
~
~
```

放开8080端口

重启防火墙

```
[root@hadoop1 bin]# service iptables restart
iptables: 将链设置为政策 ACCEPT: filter
iptables: 清除防火墙规则:
iptables: 正在卸载模块:
iptables: 应用防火墙规则:
[root@hadoop1 bin]# service iptables status
表格 : filter
Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination
1    ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
2    ACCEPT     icmp --  0.0.0.0/0           0.0.0.0/0
3    ACCEPT     all  --  0.0.0.0/0           0.0.0.0/0
4    ACCEPT     tcp  --  0.0.0.0/0           0.0.0.0/0
5    ACCEPT     tcp  --  0.0.0.0/0           0.0.0.0/0
6    REJECT     all  --  0.0.0.0/0           0.0.0.0/0           reject-with icmp-host-prohibit
d

Chain FORWARD (policy ACCEPT)
```

说明8080端口已开放

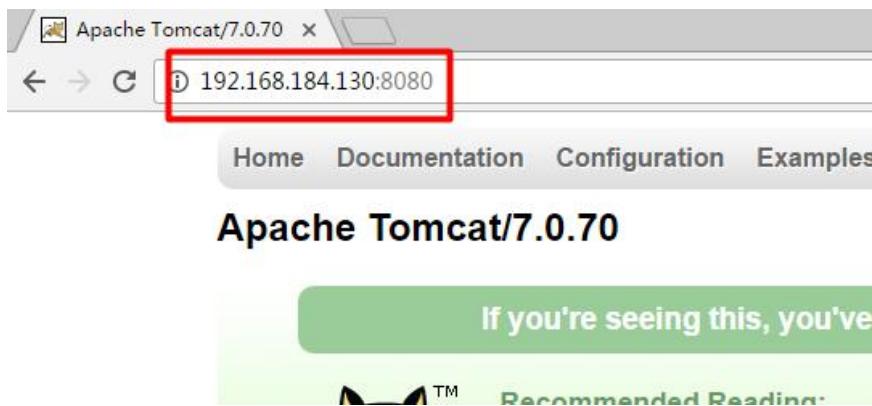
state NEW tcp dpt:8080

state NEW tcp dpt:22

reject-with icmp-host-prohibit

16.3.2 测试是否安装成功:

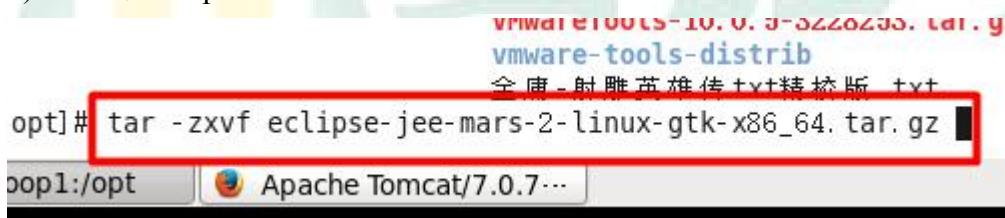
在 windows、Linux 下 访问 <http://linuxip:8080>



16.4 Eclipse 的安装

16.4.1 步骤：

1) 解压缩到/opt



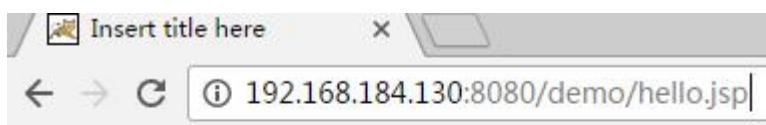
```
VMWARETOOLS-10.0.0-3220233.tar.gz
vmware-tools-distrib
全康-射雕英雄传txt精校版.txt
opt]# tar -zxvf eclipse-jee-mars-2-linux-gtk-x86_64.tar.gz
```

2) 启动 eclipse，配置 jre 和 server

启动方法 1：创建一个快捷方式

启动方法 2：进入到 eclipse 解压后的文件夹，然后执行 ./eclipse 即可

3) 编写 jsp 页面，并测试成功！



hello from Linux

16.5 mysql 的安装和配置

16.5.1 安装的步骤和文档

[说明: 因为 mysql 安装时间很长, 所以在授课时, 可以考虑最先安装 mysql]

相关的安装软件在课件

注意: 先删除一下 Mysql 相关的软件..



尚硅谷centos6.8下安装mysql.zip

16.5.2 说明

请同学们一定要按照老师的文档一步一步的安装成功。



第 17 章 大数据定制篇 Shell 编程

17.1 为什么要学习 Shell 编程

- 1)Linux 运维工程师在进行服务器集群管理时，需要编写 Shell 程序来进行服务器管理。
- 2)对于 JavaEE 和 Python 程序员来说，工作的需要，你的老大会要求你编写一些 Shell 脚本进行程序或者是服务器的维护，比如编写一个定时备份数据库的脚本。
- 3)对于大数据程序员来说，需要编写 Shell 程序来管理集群。

17.2 Shell 是什么

画一个示意图：



Shell 是一个命令行解释器，它为用户提供了一个向 Linux 内核发送请求以便运行程序的界面系统级程序，用户可以用 Shell 来启动、挂起、停止甚至是编写一些程序.

17.3 shell 编程快速入门-Shell 脚本的执行方式

17.3.1 脚本格式要求

- 1) 脚本以#!/bin/bash 开头

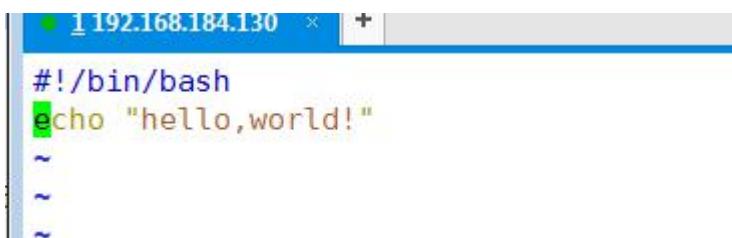
2) 脚本需要有可执行权限

17.3.2 编写第一个 Shell 脚本

- 需求说明

创建一个 Shell 脚本，输出 hello world!

看案例：



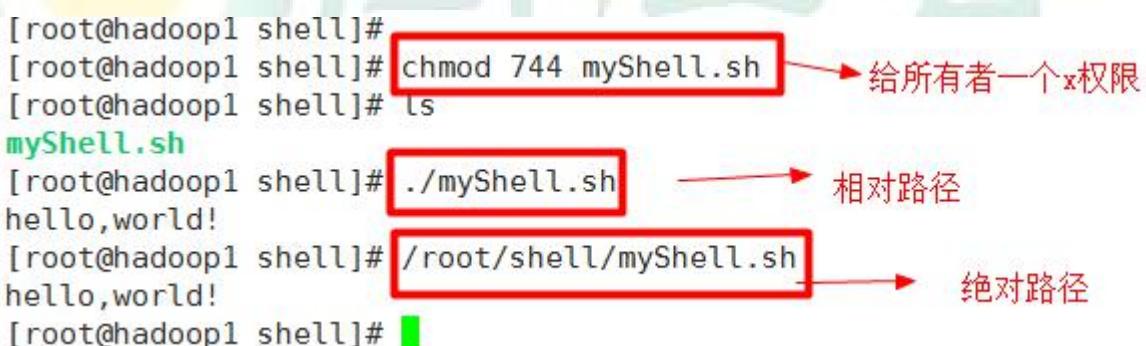
```
#!/bin/bash
echo "hello,world!"
```

17.3.3 脚本的常用执行方式

- 方式 1(输入脚本的绝对路径或相对路径)

1)首先要赋予 helloworld.sh 脚本的+x 权限

2)执行脚本



```
[root@hadoop1 shell]# chmod 744 myShell.sh → 给所有者一个x权限
[root@hadoop1 shell]# ls
myShell.sh
[root@hadoop1 shell]# ./myShell.sh → 相对路径
hello,world!
[root@hadoop1 shell]# /root/shell/myShell.sh → 绝对路径
hello,world!
[root@hadoop1 shell]#
```

- 方式 2(sh+脚本)，不推荐

说明：不用赋予脚本+x 权限，直接执行即可



```
[root@hadoop1 shell]#
[root@hadoop1 shell]# sh ./myShell.sh → 这样也可以不给 x权限, 但是不推荐
hello,world!
[root@hadoop1 shell]# sh /root/shell/myShell.sh
hello,world!
[root@hadoop1 shell]#
```

17.4 shell 的变量

17.4.1 Shell 的变量的介绍

- 1) Linux Shell 中的变量分为，**系统变量**和**用户自定义变量**。
- 2) 系统变量：\$HOME、\$PWD、\$SHELL、\$USER 等等

比如： echo \$HOME 等等..



终端截图显示了两条命令：echo "PATH=\$PATH" 和 echo "user=\$USER"。这两条命令被一个红色框包围，右侧有一个指向该框的箭头，标注为“输出系统变量”。背景中隐约可见“尚硅谷”字样。

```
#echo "PATH=$PATH"
echo "user=$USER"
```

- 3) 显示当前 shell 中所有变量：set

17.4.2 shell 变量的定义

- 基本语法
 - 1) 定义变量：变量=值
 - 2) 撤销变量：unset 变量
 - 3) 声明静态变量：readonly 变量，注意：不能 unset

- 快速入门

案例 1：定义变量 A

案例 2：撤销变量 A



```
A=100
echo "A=$A"
unset A
echo "A=$A"
```

案例 3：声明静态的变量 B=2，不能 unset



```
12
13
14 readonly A=99
15 echo "A=$A"
16 unset A
17 echo "A=$A"
set nu
```

案例 4：可把变量提升为全局环境变量，可供其他 shell 程序使用【一会举例。】

17.4.3 • 定义变量的规则

- 1) 变量名称可以由字母、数字和下划线组成，但是不能以数字开头。
- 2) 等号两侧不能有空格
- 3) 变量名称一般习惯为大写

17.4.4 • 将命令的返回值赋给变量（重点）

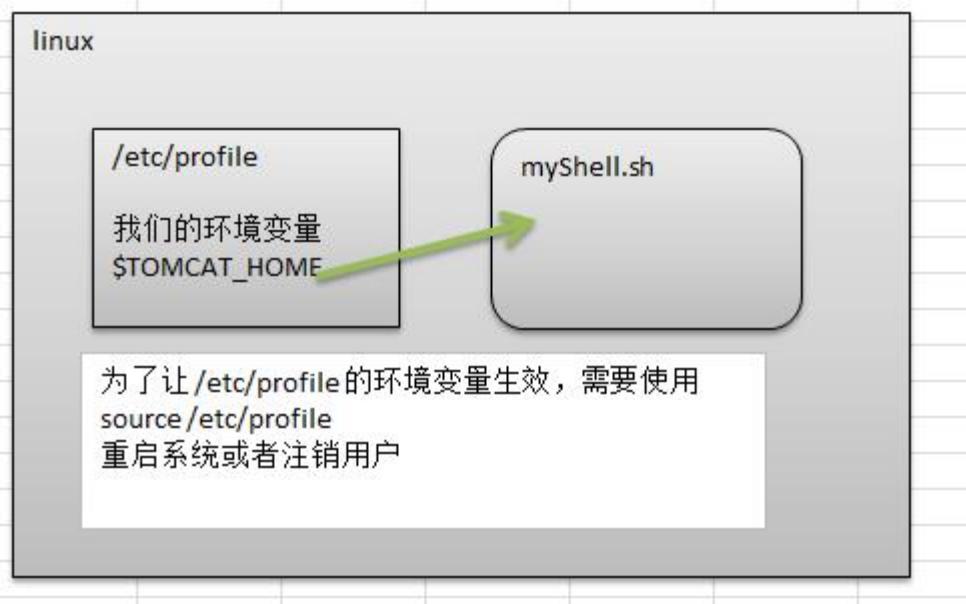
- 1) `A=`ls -la`` 反引号，运行里面的命令，并把结果返回给变量 A
- 2) `A=$(ls -la)` 等价于反引号

```
#echo "$A $MYNUM"  
RESULT=`ls -l /home`  
echo $RESULT  
echo ""  
MY_DATE=$(date)  
echo "date=$MY_DATE"  
"myShell.sh" 27L, 283C  
  
仅将文本发送到当前选项卡  
ssh://192.168.184.130:22
```

17.5 设置环境变量

17.5.1 基本语法

- 1) export 变量名=变量值 (功能描述: 将 shell 变量输出为环境变量)
- 2) source 配置文件 (功能描述: 让修改后的配置信息立即生效)
- 3) echo \$变量名 (功能描述: 查询环境变量的值)



17.5.2 快速入门

- 1) 在/etc/profile 文件中定义 TOMCAT_HOME 环境变量



```
# 定义一个自己的环境变量
TOMCAT_HOME=/opt/tomcat
export TOMCAT_HOME
"/etc/profile" 86L, 1978C
```

仅将文本发送到当前选项卡

- 2) 查看环境变量 TOMCAT_HOME 的值

```
echo $TOMCAT_HOME
```

- 3) 在另外一个 shell 程序中使用 TOMCAT_HOME



```
# 多行注释
:;<<!
RESULT=`ls -l /home`
echo $RESULT
echo ""
MY_DATE=$(date)
echo "date=$MY_DATE"
!

# 使用我们自定的环境变量
echo "tomcathome=$TOMCAT_HOME"
```

注意：在输出 TOMCAT_HOME 环境变量前，需要让其生效

```
source /etc/profile
```

17.6 位置参数变量

17.6.1 介绍

当我们执行一个 shell 脚本时，如果希望获取到命令行的参数信息，就可以使用到位置参数变量，比如：`./myshell.sh 100 200`，这个就是一个执行 shell 的命令行，可以在 myshell 脚本中获取到参数信息

17.6.2 基本语法

`$n`（功能描述：n 为数字，`$0` 代表命令本身，`$1-$9` 代表第一到第九个参数，十以上的参数需要用大括号包含，如 `${10}`）

`$*`（功能描述：这个变量代表命令行中所有的参数，`$*` 把所有的参数看成一个整体）

`$@`（功能描述：这个变量也代表命令行中所有的参数，不过`$@` 把每个参数区分对待）

`$#`（功能描述：这个变量代表命令行中所有参数的个数）

17.6.3 位置参数变量应用实例

案例：编写一个 shell 脚本 `positionPara.sh`，在脚本中获取到命令行的各个参数信息

```
#!/bin/bash
#获取到各个参数
echo "$0 $1 $2"
echo "$*"
echo "$@"
echo "参数个数=$#"
```

```
[root@hadoop1 shell]# vim positionPara.sh
[root@hadoop1 shell]# ./positionPara.sh 30 60
./positionPara.sh 30 60
30 60
30 60
参数个数=2
[root@hadoop1 shell]#
```

17.7 预定义变量

17.7.1 基本介绍

就是 shell 设计者事先已经定义好的变量，可以直接在 shell 脚本中使用

17.7.2 基本语法

\$\$ (功能描述：当前进程的进程号（PID）)

\$! (功能描述：后台运行的最后一个进程的进程号（PID）)

\$? (功能描述：最后一次执行的命令的返回状态。如果这个变量的值为 0，证明上一个命令正确执行；如果这个变量的值为非 0（具体是哪个数，由命令自己来决定），则证明上一个命令执行不正确了。)

17.7.3 应用实例

在一个 shell 脚本中简单使用一下预定义变量

```
#!/bin/bash
echo "当前的进程号=$$"
#后台的方式运行 myShell.sh
./myShell.sh &
echo "最后的进程的号=$!"
echo "执行的值=$?"
```

17.8 运算符

17.8.1 基本介绍

学习如何在 shell 中进行各种运算操作。

17.8.2 基本语法

1) “\$((运算式))” 或 “[运算式]”

2) expr m + n

注意 expr 运算符间要有空格

3) expr m - n

4) expr *, /, % 乘, 除, 取余

• 应用实例

案例 1：计算 $(2+3) \times 4$ 的值

1) \$((运算式))

```
#!/bin/bash

#第一方式$()

RESULT1=$(((2+3)*4))
echo "result1=$RESULT1"
```

2) [运算式]

```
#第二种方式[]
RESULT2=$((2+3)*4) → 推荐
echo "result2=$RESULT2"
```

3) expr

```
#使用expr
TEMP=`expr 2 + 3`
RESULT3=`expr $TEMP \* 4`
echo "result3=$RESULT3"
```

案例 2：请求出命令行的两个参数[整数]的和

```
#求出两个参数的和  
SUM=$[ $1+$2 ]  
echo "SUM=$SUM"  
"demo.sh" 21L, 279C
```

17.9 条件判断

判断语句

17.9.1 •基本语法

[condition] (注意 condition 前后要有空格)

#非空返回 true, 可使用\$?验证 (0 为 true, >1 为 false)

17.9.2 •应用实例

[atguigu] 返回 true

[] 返回 false

[condition] && echo OK || echo notok 条件满足, 执行后面的语句

17.9.3 •常用判断条件

1)两个整数的比较

= 字符串比较

-lt 小于

-le 小于等于

-eq 等于

-gt 大于

-ge 大于等于

-ne 不等于

2) 按照文件权限进行判断

-r 有读的权限 [-r 文件]

-w 有写的权限

-x 有执行的权限

3) 按照文件类型进行判断

-f 文件存在并且是一个常规的文件

-e 文件存在

-d 文件存在并是一个目录

17.9.4 应用实例

案例 1: "ok"是否等于"ok"

判断语句:

```
#案例1: "ok"是否等于"ok"
[
if [ "ok100" = "ok" ]
then
    echo "equal"
fi
~
```

案例 2: 23 是否大于等于 22

判断语句:

```
#案例2: 23是否大于等于22
```

```
if [ 23 -gt 22 ]
then
    echo "大于"
fi
```

案例 3: /root/install.log 目录中的文件是否存在

判断语句:

```
#案例3: /root/shell/aaa.txt 目录中的文件是否存在
if [ -e /root/shell/aaa.txt ]
then
    echo "存在"
fi
```

"judge.sh" 21L, 284C

仅将文本发送到当前选项卡

17.10 流程控制

17.10.1 if 判断

- 基本语法

```
if [ 条件判断式 ];then
```

 程序

```
fi
```

或者

```
if [ 条件判断式 ]
```

```
    then
```

 程序

```
    elif [条件判断式]
```



```
then
```

```
    程序
```

```
fi
```

注意事项：(1) [条件判断式]，中括号和条件判断式之间必须有空格 (2) 推荐使用第二种方式

- 应用实例

案例：请编写一个 shell 程序，如果输入的参数，大于等于 60，则输出 "及格了"，如果小于 60，则输出 "不及格"

```
#!/bin/bash

#案例：请编写一个shell程序，如果输入的参数，大于等于60，则输出 "及格了"，如果小于60，则输出 "不及格"

if [ $1 -ge 60 ]
then
    echo "及格了"
elif [ $1 -lt 60 ]
then
    echo "不及格"
fi
```

17.10.2 case 语句

- 基本语法

```
case $变量名 in
```

```
    "值 1")
```

如果变量的值等于值 1，则执行程序 1

```
    ::
```

```
    "值 2")
```

如果变量的值等于值 2，则执行程序 2

```
    ::
```

…省略其他分支…

```
*)
```

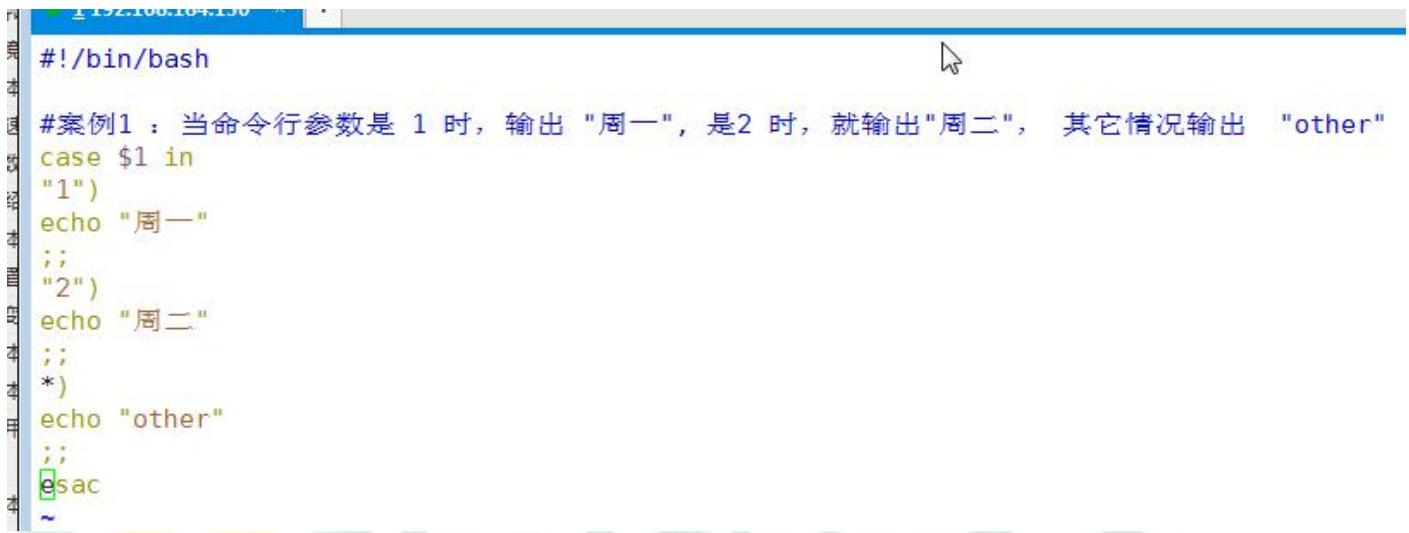
如果变量的值都不是以上的值，则执行此程序

```
;;
```

```
esac
```

- 应用实例

案例 1：当命令行参数是 1 时，输出 "周一"，是 2 时，就输出"周二"， 其它情况输出 "other"



```
#!/bin/bash

#案例1：当命令行参数是 1 时，输出 "周一"，是2 时，就输出"周二"， 其它情况输出 "other"
case $1 in
"1")
echo "周一"
;;
"2")
echo "周二"
;;
*)
echo "other"
;;
esac
~
```

17.10.3 for 循环

- 基本语法 1

```
for 变量 in 值1 值2 值3…
```

```
do
```

```
    程序
```

```
done
```

- 应用实例

案例 1：打印命令行输入的参数 【会使用到\$* \$@】

```
#!/bin/bash
#案例1：打印命令行输入的参数

#使用 $*
for i in "$*"
do
    echo "the num is $i"
done
echo "====="
#使用 ${@}
for j in "${@}"
do
    echo "the num is $j"
done
```

• 基本语法 2

for ((初始值;循环控制条件;变量变化))

do

程序

done

• 应用实例

案例 1：从 1 加到 100 的值输出显示

```
#!/bin/bash
#案例1：从1加到100的值输出显示

#定义一个变量
SUM=0
for((i=1;i<=100;i++))
do
    SUM=$[SUM+$i]
done
echo "sum=$SUM"
```

17.10.4 while 循环

- 基本语法 1

```
while [ 条件判断式 ]
```

```
do
```

```
    程序
```

```
done
```

- 应用实例

案例 1：从命令行输入一个数 n，统计从 1+..+n 的值是多少？

```
#!/bin/bash

#[案例1：从命令行输入一个数n，统计从 1+..+n 的值是多少

SUM=0
i=0
while [ $i -le $1 ]
do
    SUM=$[SUM+$i]
    i=$[i+1]
done
echo "sum= $SUM"
~
```

17.11 read 读取控制台输入

17.11.1 基本语法

read(选项)(参数)

选项：

-p: 指定读取值时的提示符;

-t: 指定读取值时等待的时间（秒），如果没有在指定的时间内输入，就不再等待了。。

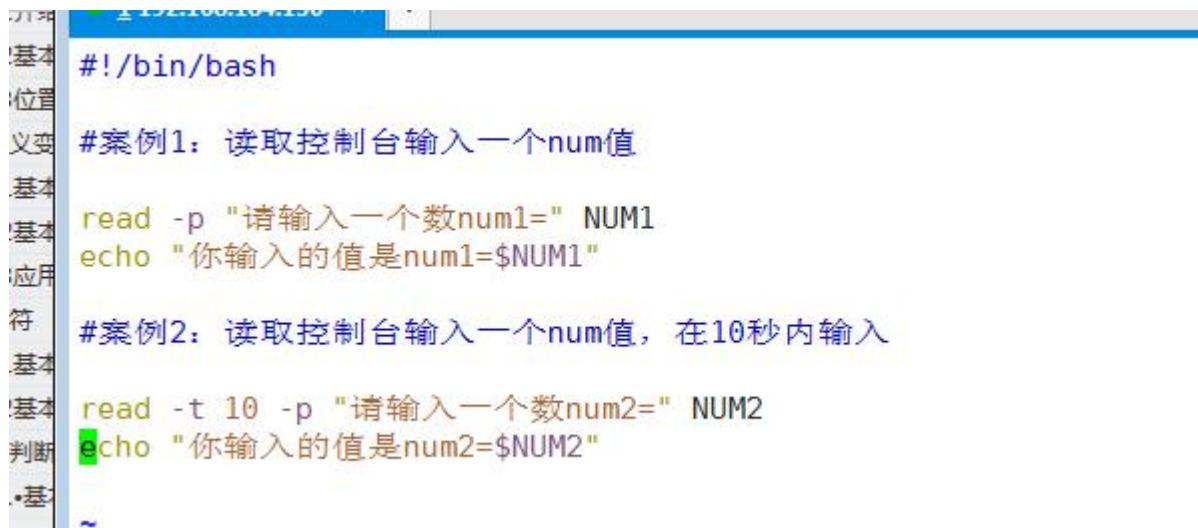
参数

变量: 指定读取值的变量名

17.11.2 应用实例

案例 1: 读取控制台输入一个 num 值

案例 2: 读取控制台输入一个 num 值，在 10 秒内输入。



```
#!/bin/bash

#案例1: 读取控制台输入一个num值
read -p "请输入一个数num1=" NUM1
echo "你输入的值是num1=$NUM1"

#案例2: 读取控制台输入一个num值, 在10秒内输入
read -t 10 -p "请输入一个数num2=" NUM2
echo "你输入的值是num2=$NUM2"
```

17.12 函数

17.12.1 函数介绍

shell 编程和其它编程语言一样，有系统函数，也可以自定义函数。系统函数中，我们这里就介绍两个。

17.12.2 系统函数

• basename 基本语法

功能：返回完整路径最后 / 的部分，常用于获取文件名

basename [pathname] [suffix]

basename [string] [suffix] （功能描述： basename 命令会删掉所有的前缀包括最后一个（ ‘/’ ）字符，然后将字符串显示出来。

选项：

suffix 为后缀，如果 suffix 被指定了， basename 会将 pathname 或 string 中的 suffix 去掉。

- dirname 基本语法

功能：返回完整路径最后 / 的前面的部分，常用于返回路径部分

dirname 文件绝对路径 （功能描述：从给定的包含绝对路径的文件名中去除文件名（非目录的部分），然后返回剩下的路径（目录的部分））

17.12.3 • 应用实例

案例 1：请返回 /home/aaa/test.txt 的 "test.txt" 部分

```
[root@hadoop1 shell]#  
[root@hadoop1 shell]# basename /home/aaa/test.txt  
test.txt  
[root@hadoop1 shell]# basename /home/aaa/test.txt .txt  
test  
[root@hadoop1 shell]#
```

案例 2：请返回 /home/aaa/test.txt 的 /home/aaa

```
tes  
[root@hadoop1 shell]# dirname /home/aaa/test.txt  
/home/aaa  
[root@hadoop1 shell]# dirname /home/aaa/ccc/jj/ooo/test.txt  
/home/aaa/ccc/jj/ooo  
[root@hadoop1 shell]#
```

17.12.4 自定义函数

- 基本语法

```
[ function ] funname()  
{
```

```
Action;  
[return int;]  
}
```

调用直接写函数名: `funname [值]`

- 应用实例

案例 1: 计算输入两个参数的和 (`read`) , `getSum`

```
#!/bin/bash  
  
#案例1: 计算输入两个参数的和 (read) , getSum  
  
function getSum(){  
  
    SUM=$[$n1+$n2]  
    echo "和是=$SUM"  
}  
  
read -p "请输入第一个数n1" n1  
read -p "请输入第二个数n2" n2  
  
#调用getSum  
getSum $n1 $n2
```

17.13Shell 编程综合案例

需求分析

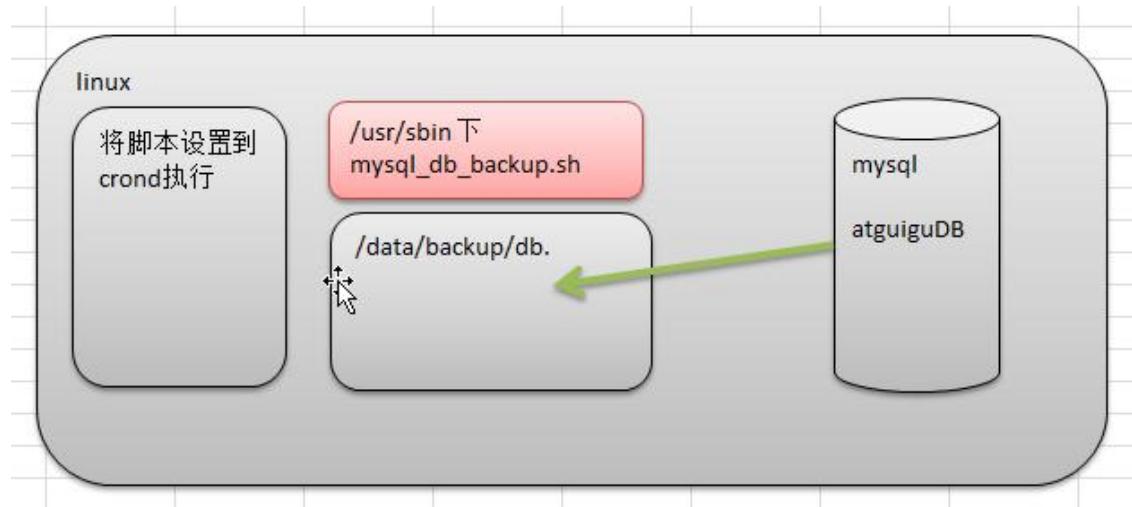
- 1) 每天凌晨 2:10 备份 数据库 `atguiguDB` 到 `/data/backup/db`
- 2) 备份开始和备份结束能够给出相应的提示信息
- 3) 备份后的文件要求以备份时间为文件名，并打包成 `.tar.gz` 的形式，比如：

`2018-03-12_230201.tar.gz`

- 4) 在备份的同时，检查是否有 10 天前备份的数据库文件，如果有就将其删除。

编写一个 shell 脚本。

思路分析：



代码实现：

```
[root@hadoop1 2018_03_20_203317]#  
[root@hadoop1 2018_03_20_203317]# vim /usr/sbin/mysql_db_backup.sh  
  
echo "=====开始备份======"  
echo "=====备份的路径是 $BACKUP/$DATETIME.tar.gz"  
  
#主机  
HOST=localhost  
#用户名  
DB_USER=root  
#密码  
DB_PWD=root  
#备份数据库名  
DATABASE=atguiguDB  
#创建备份的路径  
#如果备份的路径文件夹存在，就使用，否则就创建  
[ ! -d "$BACKUP/$DATETIME" ] && mkdir -p "$BACKUP/$DATETIME"  
#执行mysql的备份数据库的指令  
mysqldump -u${DB_USER} -p${DB_PWD} --host=$HOST $DATABASE | gzip > $BACKUP/$DATETIME/$DATETIME.sql.gz  
#打包备份文件  
cd $BACKUP  
tar -zcvf $DATETIME.tar.gz $DATETIME  
#删除临时目录  
rm -rf $BACKUP/$DATETIME  
  
#删除10天前的备份文件  
find $BACKUP -mtime +10 -name "*.tar.gz" -exec rm -rf {} \;  
echo "=====备份文件成功======"  
"/usr/sbin/mysql_db_backup.sh" 36L, 942C
```

```
要在当前会话中添加新命令，点击左侧的前头按钮。  
● 1 192.168.184.130 × +  
10 2 * * * /usr/sbin/mysql_db_backup.sh  
~  
~  
~  
~  
→ 在crontab中的配置
```



第 18 章 Python 定制篇 开发平台 Ubuntu

18.1 Ubuntu 的介绍

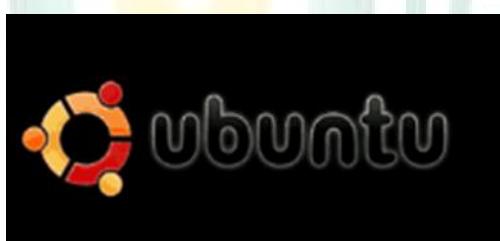
Ubuntu（友帮拓、优般图、乌班图）是一个以桌面应用为主的开源 GNU/Linux 操作系统，Ubuntu 是基于 GNU/Linux，支持 x86、amd64（即 x64）和 ppc 架构，由全球化的专业开发团队（Canonical Ltd）打造的。

专业的 Python 开发者一般会选择 Ubuntu 这款 Linux 系统作为生产平台。

温馨提示：

Ubuntu 和 Centos 都是基于 GNU/Linux 内核的，因此基本使用和 Centos 是几乎一样的，它们的各种指令可以通用，同学们在学习和使用 Ubuntu 的过程中，会发现各种操作指令在前面学习 CentOS 都使用过。只是界面和预安装的软件有所差别。

Ubuntu 下载地址：<http://cn.ubuntu.com/download/>



18.2 Ubuntu 的安装

18.2.1 安装的步骤



尚硅谷Python安装Ubuntu.zip

18.2.2 设置 Ubuntu 支持中文

默认安装的 ubuntu 中只有英文语言，因此是不能显示汉字的。要正确显示汉字，需要安装中文语言包。

安装中文支持步骤：

1. 单击左侧图标栏打开 System Settings (系统设置) 菜单，点击打开 Language Support (语言支持) 选项卡。
2. 点击 Install / Remove Languages，在弹出的选项卡中下拉找到 Chinese(Simplified)，即中文简体，在后面的选项框中打勾。然后点击 Apply Changes 提交，系统会自动联网下载中文语言包。（保证 ubuntu 是联网的）。
3. 这时“汉语（中国）”在最后一位因为当前第一位是“English”，所以默认显示都是英文。我们如果希望默认显示用中文，则应该将“汉语（中国）”设置为第一位。设置方法是拖动，鼠标单击“汉语（中国）”，当底色变化（表示选中了）后，按住鼠标左键不松手，向上拖动放置到第一位。
4. 设置后不会即刻生效，需要下一次登录时才会生效。



18.3 Ubuntu 的 root 用户

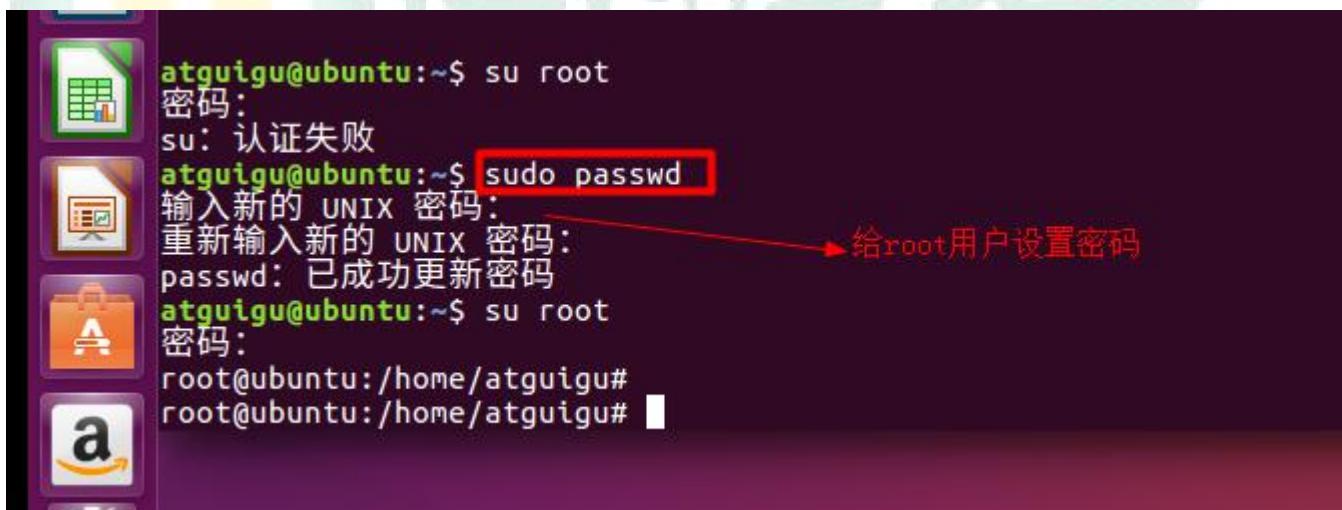
18.3.1 介绍

安装 ubuntu 成功后，都是普通用户权限，并没有最高 root 权限，如果需要使用 root 权限的时候，通常都会在命令前面加上 sudo 。有的时候感觉很麻烦。

我们一般使用 su 命令来直接切换到 root 用户的，但是如果没给 root 设置初始密码，就会抛出 su : Authentication failure 这样的问题。所以，我们只要给 root 用户设置一个初始密码就好了。

18.3.2 给 root 用户设置密码并使用

- 1) 输入 sudo passwd 命令，输入一般用户密码并设定 root 用户密码。
- 2) 设定 root 密码成功后，输入 su 命令，并输入刚才设定的 root 密码，就可以切换成 root 了。
提示符\$代表一般用户，提示符#代表 root 用户。
- 3) 输入 exit 命令，退出 root 并返回一般用户
- 4) 以后就可以使用 root 用户了



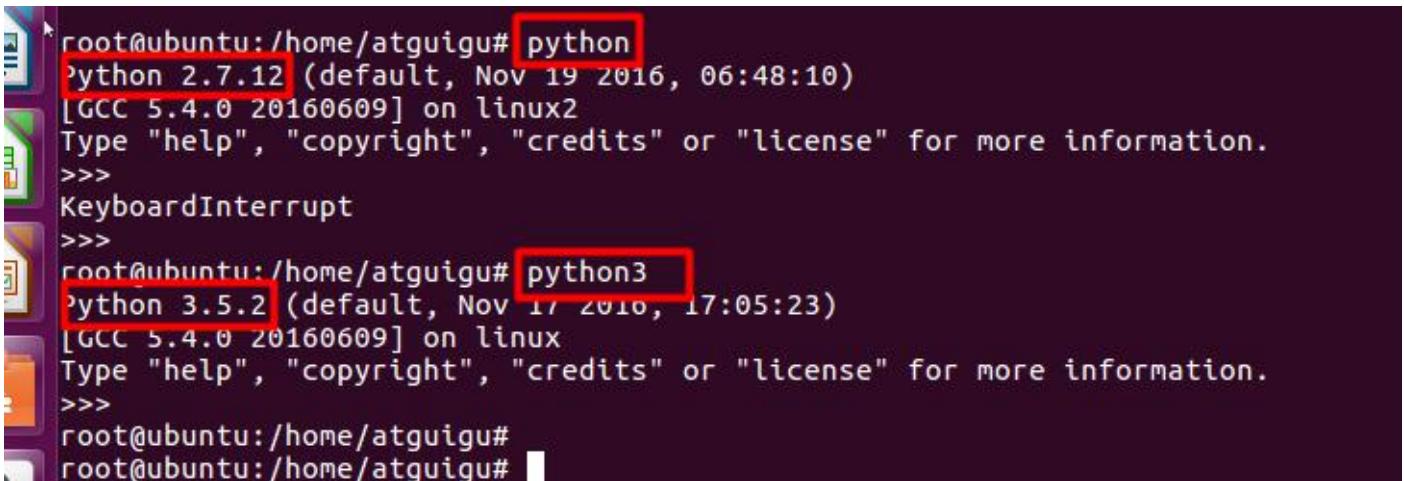
The screenshot shows a terminal window on an Ubuntu desktop. The user has run the command `sudo passwd`. A red box highlights this command, with a red arrow pointing from it to the text "给root用户设置密码" (Set root user password). The terminal output is as follows:

```
atguigu@ubuntu:~$ su root  
密码:  
su: 认证失败  
atguigu@ubuntu:~$ sudo passwd  
输入新的 UNIX 密码:  
重新输入新的 UNIX 密码:  
passwd: 已成功更新密码  
atguigu@ubuntu:~$ su root  
密码:  
root@ubuntu:/home/atguigu#  
root@ubuntu:/home/atguigu#
```

18.4 Ubuntu 下开发 Python

18.4.1 说明

安装好 Ubuntu 后，默认就已经安装好 Python 的开发环境[Python2.7 和 Python3.5]。



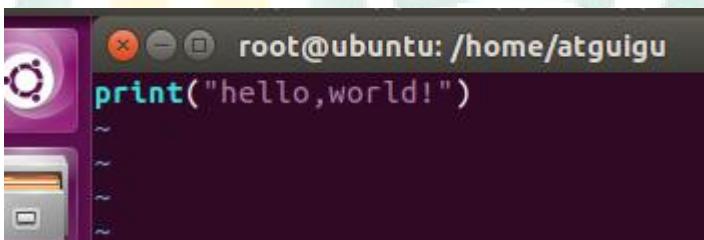
```
root@ubuntu:/home/atguigu# python
python 2.7.12 (default, Nov 19 2016, 06:48:10)
[GCC 5.4.0 20160609] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
KeyboardInterrupt
>>>
root@ubuntu:/home/atguigu# python3
python 3.5.2 (default, Nov 17 2016, 17:05:23)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
root@ubuntu:/home/atguigu#
root@ubuntu:/home/atguigu#
```

18.4.2 在 Ubuntu 下开发一个 Python 程序

- 1) vim hello.py [编写 hello.py]

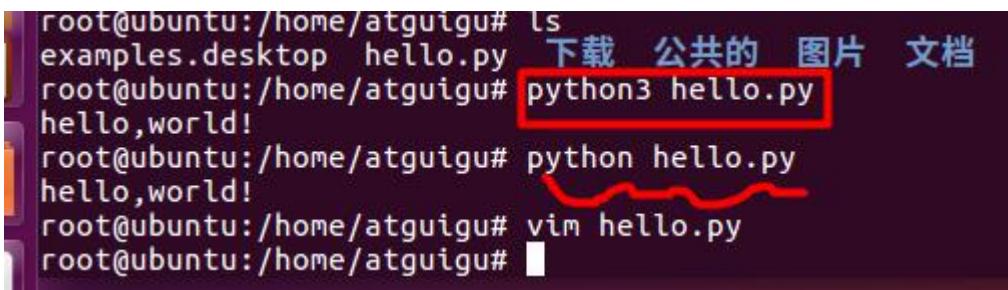
提示：如果 Ubuntu 没有 vim 我们可以根据提示信息安装一个 vim

```
apt install vim
```



```
root@ubuntu:/home/atguigu# print("hello,world!")
~
```

- 2) python3 hello.py [运行 hello.py]



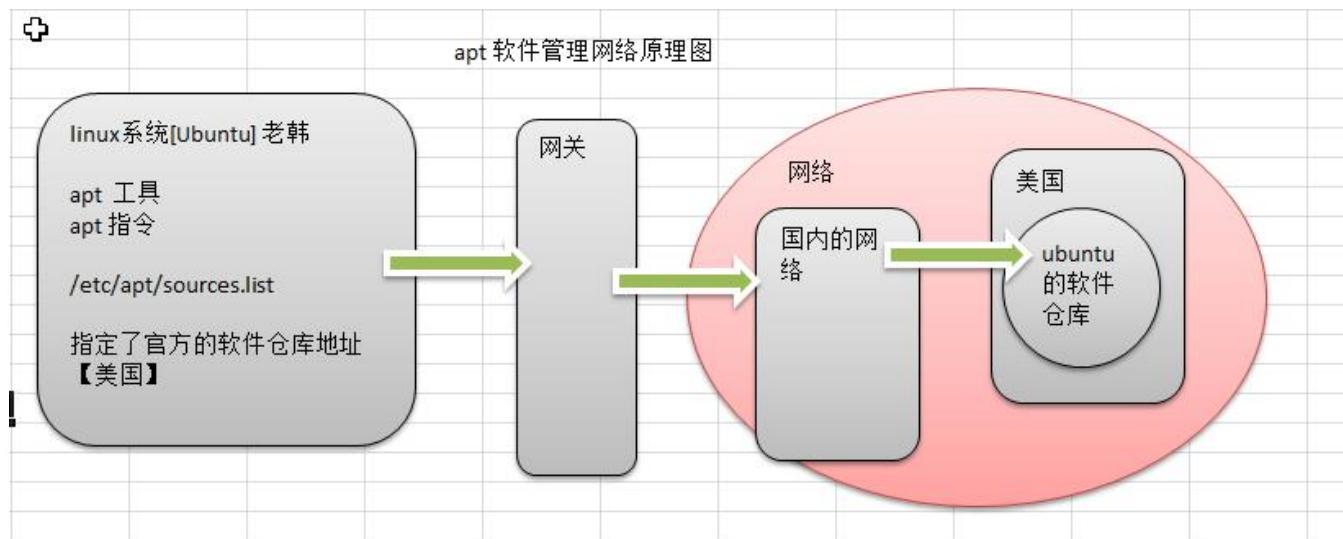
```
root@ubuntu:/home/atguigu# ls
examples.desktop  hello.py  下载  公共的  图片  文档
root@ubuntu:/home/atguigu# python3 hello.py
hello,world!
root@ubuntu:/home/atguigu# python hello.py
hello,world!
root@ubuntu:/home/atguigu# vim hello.py
root@ubuntu:/home/atguigu#
```

第 19 章 Python 定制篇 apt 软件管理和远程登录

19.1 apt 介绍

apt 是 Advanced Packaging Tool 的简称，是一款安装包管理工具。在 Ubuntu 下，我们可以使用 apt 命令可用于软件包的安装、删除、清理等，类似于 Windows 中的软件管理工具。

unbuntu 软件管理的原理示意图：



19.2 Ubuntu 软件操作的相关命令

sudo apt-get update 更新源

sudo apt-get install package 安装包

sudo apt-get remove package 删除包

sudo apt-cache search package 搜索软件包

sudo apt-cache show package 获取包的相关信息，如说明、大小、版本等

sudo apt-get install package --reinstall 重新安装包

sudo apt-get -f install 修复安装

sudo apt-get remove package --purge 删除包，包括配置文件等

sudo apt-get build-dep package 安装相关的编译环境

sudo apt-get upgrade 更新已安装的包

sudo apt-get dist-upgrade 升级系统

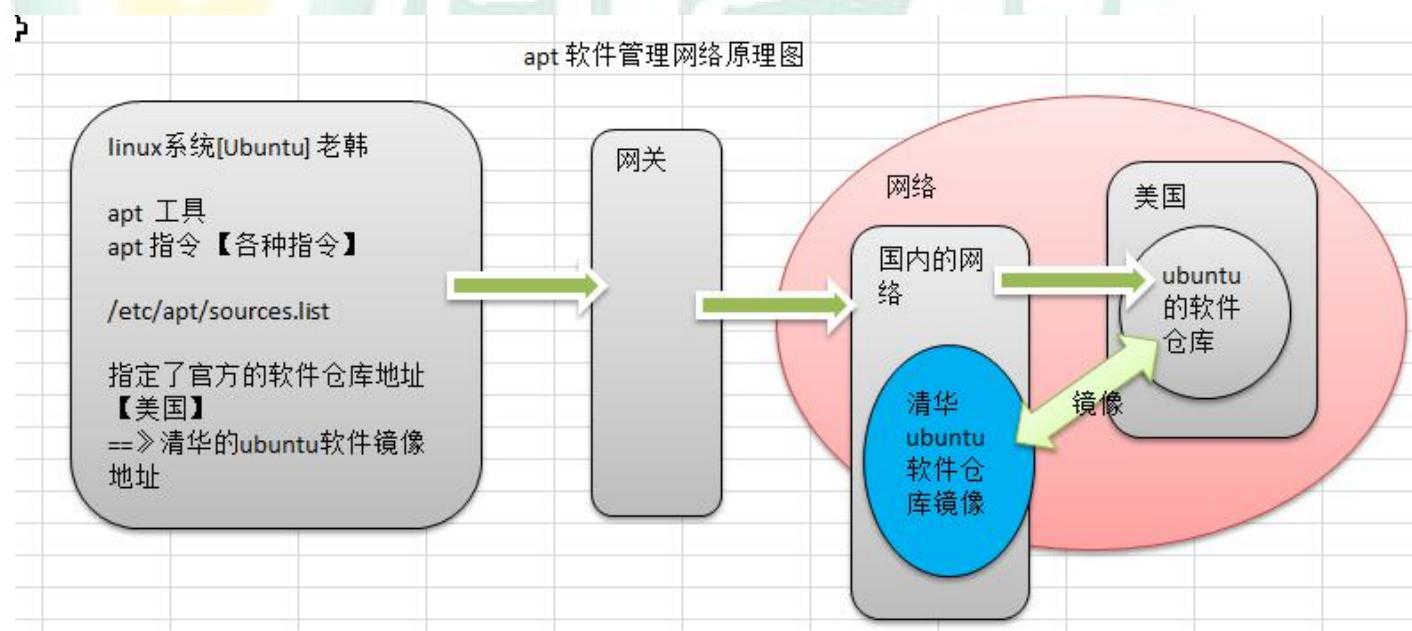
sudo apt-cache depends package 了解使用该包依赖那些包

sudo apt-cache rdepends package 查看该包被哪些包依赖

sudo apt-get source package 下载该包的源代码

19.3 更新 Ubuntu 软件下载地址

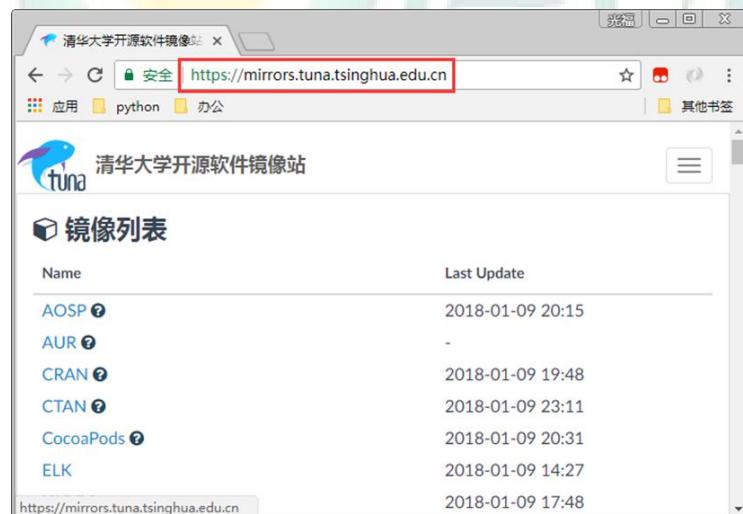
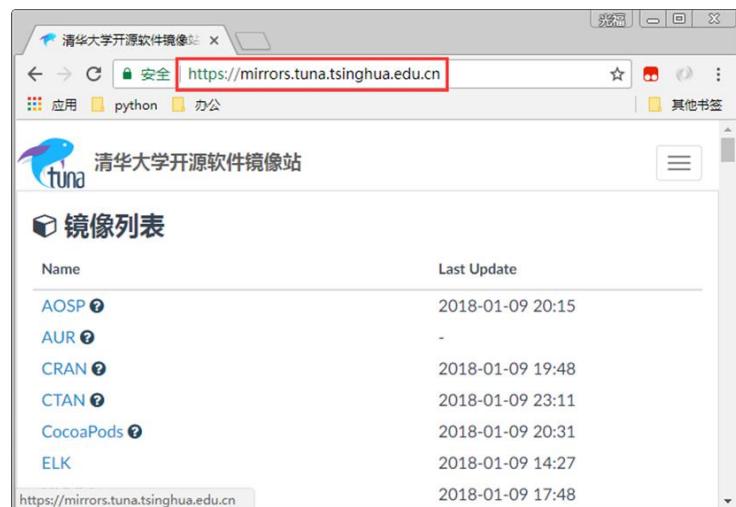
19.3.1 原理示意图

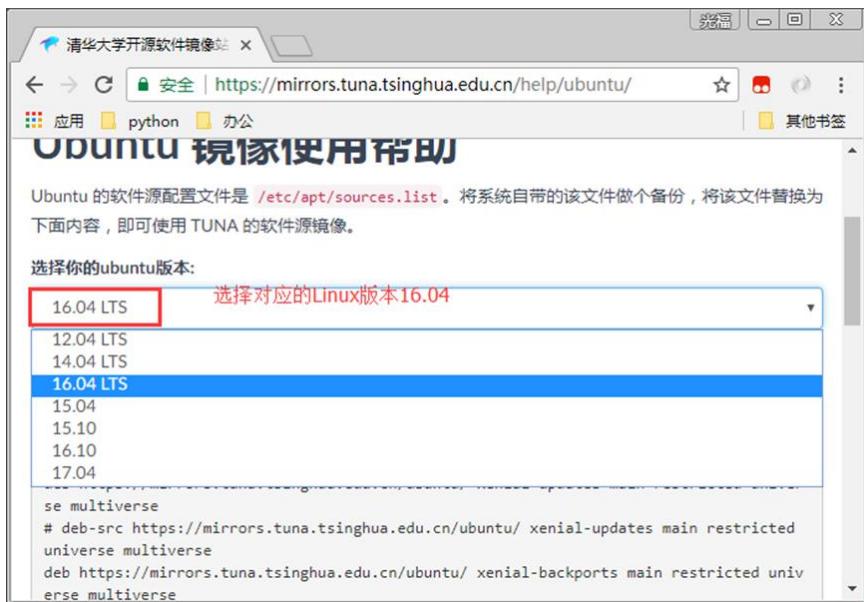


19.3.2 寻找国内镜像源

<https://mirrors.tuna.tsinghua.edu.cn/>

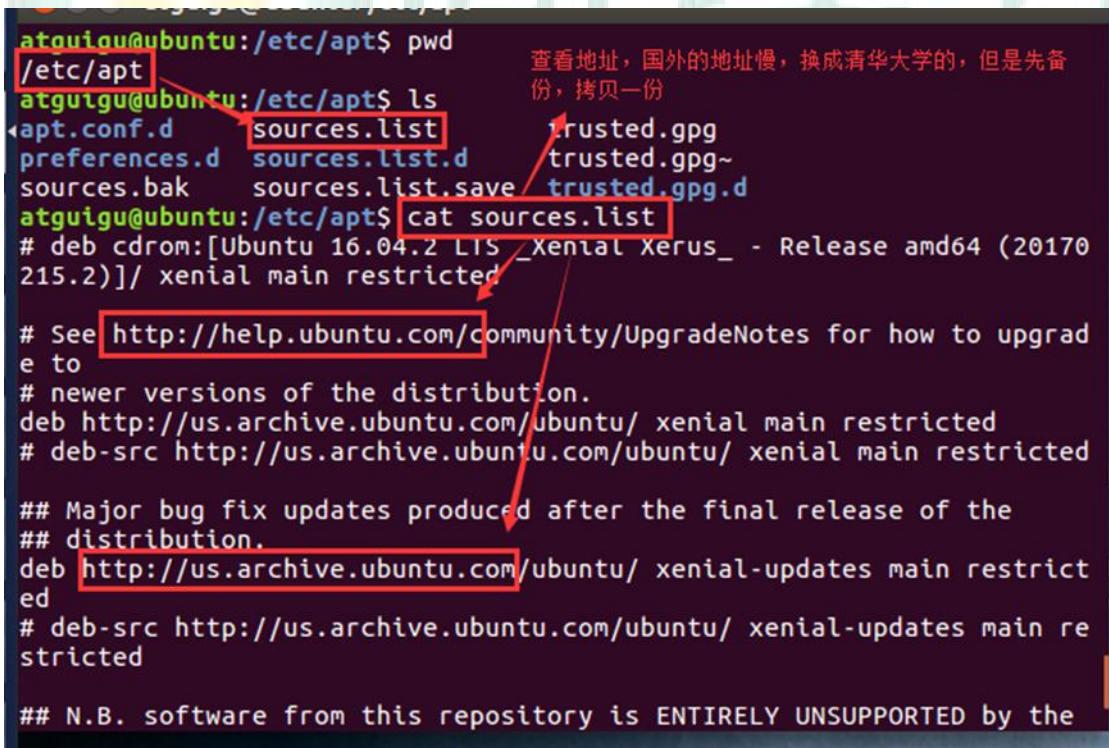
所谓的镜像源：可以理解为提供下载
软件的地方，比如 Android 手机上可以
下载软件的安卓市场；iOS 手机上可
以下载软件的 AppStore





19.3.3 备份 Ubuntu 默认的源地址

```
sudo cp /etc/apt/sources.list /etc/apt/sources.list.backup
```



```
atguigu@ubuntu:/etc/apt$ pwd
/etc/apt
atguigu@ubuntu:/etc/apt$ ls
apt.conf.d      sources.list      trusted.gpg
preferences.d   sources.list.d    trusted.gpg~
sources.bak     sources.list.save trusted.gpg.d
atguigu@ubuntu:/etc/apt$ cat sources.list
# deb cdrom:[Ubuntu 16.04.2 LTS _Xenial Xerus_ - Release amd64 (20170215.2)]/ xenial main restricted
# See http://help.ubuntu.com/community/UpgradeNotes for how to upgrade to
# newer versions of the distribution.
deb http://us.archive.ubuntu.com/ubuntu/ xenial main restricted
# deb-src http://us.archive.ubuntu.com/ubuntu/ xenial main restricted

## Major bug fix updates produced after the final release of the
## distribution.
deb http://us.archive.ubuntu.com/ubuntu/ xenial-updates main restricted
# deb-src http://us.archive.ubuntu.com/ubuntu/ xenial-updates main restricted

## N.B. software from this repository is ENTIRELY UNSUPPORTED by the
## distribution.
```

查看地址，国外的地址慢，换成清华大学的，但是先备份，拷贝一份

```
atguigu@ubuntu:/etc/apt$ ls
apt.conf.d      sources.list      trusted.gpg
preferences.d    sources.list.d    trusted.gpg~
sources.bak     sources.list.save  trusted.gpg.d  拷贝一份
atguigu@ubuntu:/etc/apt$ cp sources.list sources.list.backup
cp: 无法创建普通文件'sources.list.backup': 权限不够
atguigu@ubuntu:/etc/apt$ sudo cp sources.list sources.list.backup
atguigu@ubuntu:/etc/apt$ ls
apt.conf.d      sources.list      sources.list.save  trusted.gpg.d
preferences.d    sources.list.backup  trusted.gpg
sources.bak     sources.list.d    trusted.gpg~
atguigu@ubuntu:/etc/apt$                                备份成功，现在可以修改sources.list了
```

19.3.4 更新源服务器列表

```
preferences.d  sources.list  sources.list.d  trusted.gpg.d
atguigu@ubuntu:/etc/apt$ echo '' > sources.list
bash: sources.list: 权限不够
atguigu@ubuntu:/etc/apt$ sudo echo '' > sources.list
bash: sources.list: 权限不够
atguigu@ubuntu:/etc/apt$ su root
密码:                                                 切换成root
root@ubuntu:/etc/apt# echo '' > sources.list          将 sources.list 清空
root@ubuntu:/etc/apt# vim sources.list               在 vim sources.list, 我们将清华镜像
root@ubuntu:/etc/apt# exit                           地址拷贝到该文件
exit
atguigu@ubuntu:/etc/apt$ sudo apt-get update        更新ubuntu的软件列表
获取:5 https://mirrors.tuna.tsinghua.edu.cn/ubuntu xenial/main amd64 Packages
,201 kB
获取:6 https://mirrors.tuna.tsinghua.edu.cn/ubuntu xenial/main i386 Packages [
```

19.4 Ubuntu 软件安装, 卸载的最佳实践

19.4.1 案例说明: 使用 apt 完成安装和卸载 vim 软件, 并查询 vim 软件的信息:

```
sudo apt-get remove vim
```

```
atguigu@ubuntu:/etc/apt$ sudo apt-get remove vim
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
下列软件包是自动安装的并且现在不需要了:
  vim-runtime
使用'sudo apt autoremove'来卸载它(它们)。
下列软件包将被【卸载】:
  vim
升级了 0 个软件包, 新安装了 0 个软件包, 要卸载 1 个软件包, 有 502 个
级。
解压缩后将会空出 2,458 kB 的空间。
您希望继续执行吗? [Y/n] Y
(正在读取数据库 ... 系统当前共安装有 178840 个文件和目录。)
正在卸载 vim (2:7.4.1689-3ubuntu1.2) ...
update-alternatives: 使用 /usr/bin/vim.tiny 来在自动模式中提供 /usr/
update-alternatives: 使用 /usr/bin/vim.tiny 来在自动模式中提供 /usr/
ew)
update-alternatives: 使用 /usr/bin/vim.tiny 来在自动模式中提供 /usr/
update-alternatives: 使用 /usr/bin/vim.tiny 来在自动模式中提供 /usr/
sudo apt-get install vim
```

```
请尝试: sudo apt install <选定的软件包>
atguigu@ubuntu:/etc/apt$ sudo apt-get install vim
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
建议安装:
  ctags vim-doc vim-scripts
下列【新】软件包将被安装:
  vim
升级了 0 个软件包, 新安装了 1 个软件包, 要卸载 0 个软件包, 有
级。
需要下载 1,036 kB 的归档。
```

```
sudo apt-cache show vim
```

```
atguigu@ubuntu:/etc/apt$ sudo apt-cache show vim
Package: vim
Architecture: amd64
Version: 2:7.4.1689-3ubuntu1.2
Priority: optional
Section: editors
```

19.5 使用 ssh 远程登录 Ubuntu

19.5.1 ssh 介绍

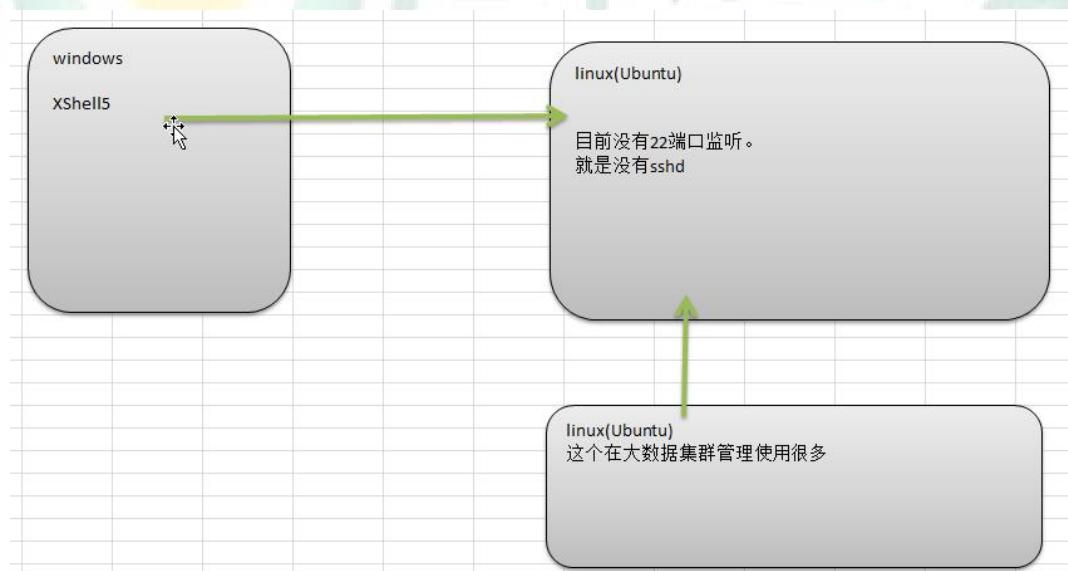
SSH 为 Secure Shell 的缩写，由 IETF 的网络工作小组 (Network Working Group) 所制定；SSH 为建立在应用层和传输层基础上的安全协议。

SSH 是目前较可靠，专为远程登录会话和其他网络服务提供安全性的协议。常用于远程登录，以及用户之间进行资料拷贝。几乎所有 UNIX 平台—包括 HP-UX、Linux、AIX、Solaris、Digital UNIX、Irix，以及其他平台，都可运行 SSH。

使用 SSH 服务，需要安装相应的服务器和客户端。客户端和服务器的关系：如果，A 机器想被 B 机器远程控制，那么，A 机器需要安装 SSH 服务器，B 机器需要安装 SSH 客户端。

和 CentOS 不一样，Ubuntu 默认没有安装 SSHD 服务，因此，我们不能进行远程登录。

19.5.2 原理示意图：



19.6 使用 ssh 远程登录 Ubuntu

19.6.1 安装 SSH 和启用

```
sudo apt-get install openssh-server
```

执行上面指令后，在当前这台 Linux 上就安装了 SSH 服务端和客户端。

```
service sshd restart
```

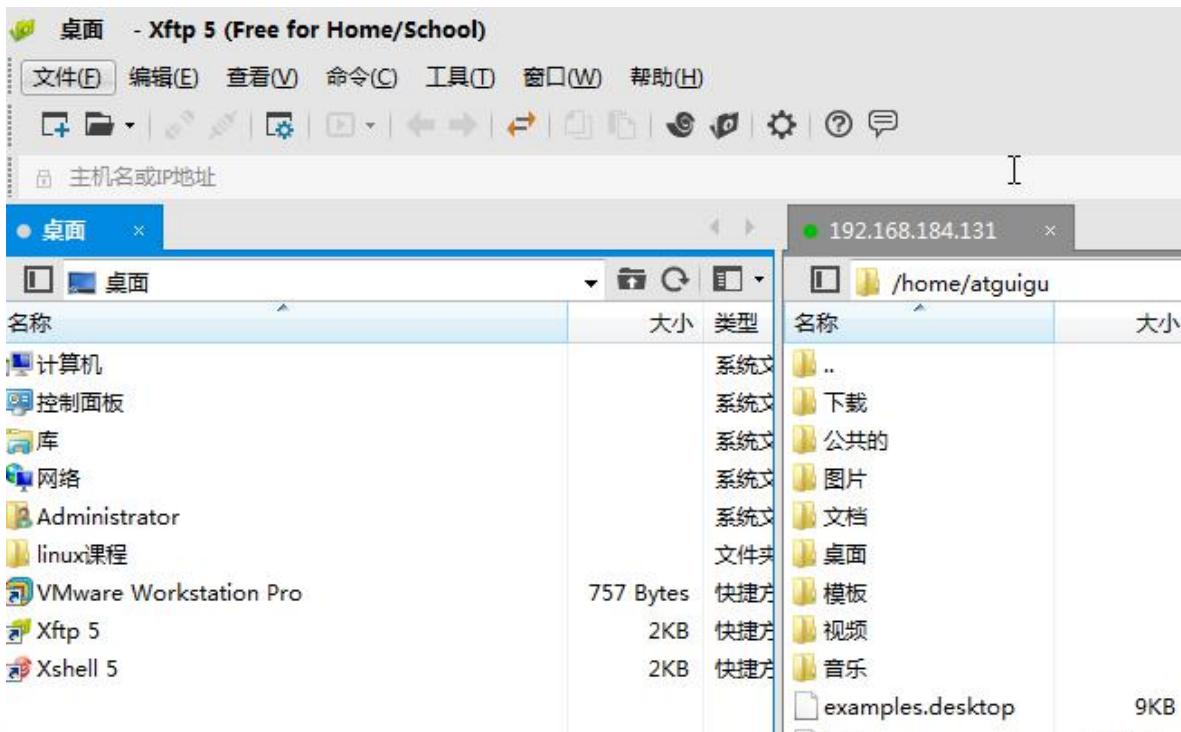
执行上面的指令，就启动了 sshd 服务。会监听端口 22

Proto	Recv-Q	Send-Q	Local Address	Foreign Address
PID/Program name				
tcp	0	0	127.0.1.1:53	0.0.0.0:*
-				
tcp	0	0	0.0.0.0:22	0.0.0.0:*
-				
tcp	0	0	127.0.0.1:631	0.0.0.0:*
-				
tcp6	0	0	:::22	:::*
-				

19.6.2 在 Windows 使用 XShell5/XFTP5 登录 Ubuntu

前面我们已经安装了 XShell5，直接使用即可。

注意：使用 atguigu 用户登录，需要的时候再 su - 切换成为 root 用户



19.6.3 从 linux 系统客户机远程登陆 linux 系统服务器

首先，我们需要在 linux 的系统客户机也要安装 openssh-server

- 基本语法：

ssh 用户名@IP

例如： ssh atguigu@192.168.188.131

使用 ssh 访问，如访问出现错误。可查看是否有该文件 `~/.ssh/known_hosts` 尝试删除该文件解决。

- 登出

登出命令： exit 或者 logout



尚硅谷 Linux 课程

