# CentOS虚拟机基础知识

对于一些初学者，我们建议把centos安装到虚拟机，进行学习，等学到一定阶段，可进行物理安装，包括内核源码编译安装，因为在虚拟机上学习有很多好处，所以这里首先要学习虚拟机的一些相关知识。

## 1.1、[VMware Workstation的虚拟网络](http://blog.csdn.net/adultf/article/details/7290999)

<http://blog.csdn.net/adultf/article/details/7290999> 资料地址

### 1.1.1 基本概念

调制解调器(Modem):

实现模拟信号与数字信号间的相互转换，可以忽略成一条网线。

路由器:

提供路由、DHCP、NAT等<功能/服务>的专门设备，连接不同的网络(包括局域网与WAN之间的连接，提供互联网连接)。

网关:

网关实质上是一个网络通向其他网络的IP地址。比如有网络A和网络B，

网络A的IP地址范围为“192.168.1.1~192.168.1.254”，

子网掩码为255.255.255.0；

网络B的IP地址范围为“192.168.2.1~192.168.2.254”，

子网掩码为255.255.255.0。

在没有路由器的情况下，两个网络之间是不能进行TCP/IP通信的，即使是两个网络连接在同一台交换机（或集线器） 上，TCP/IP协议也会根据子网掩码（255.255.255.0）判定两个网络中的主机处在不同的网络里。而要实现这两个网络之间的通信，则必须通过网关。如果网络A中的主机发现数据包的目的主机不在本地网络中，就把数据包转发给它自己的网关，再由网关转发给网络B的网关，网络B的网关再转发给网络B 的某个主机。只有设置好网关的IP地址，TCP/IP协议才能实现不同网络之间的相互通信。那么这个IP地址是哪台机器的IP地址呢？网关的IP地址是具有路由功能的设备的IP地址，具有路由功能的设备有路由器、启用了路由协议的服务器（实质上相当于一台路由器）、代理服务器（也相当于一台路由器）。

默认网关:

如果搞清了什么是网关，默认网关也就好理解了。就好像一个房间可以有多扇门一样，一台主机可以有多个网关。默认网关的意思是一台主机如果找不到可用的网关，就把数据包发给默认指定的网关，由这个网关来处理数据包。现在主机使用的网关，一般指的是默认网关。默认网关一般填写255.255.255.0。配置默认网关可以在 IP 路由表中创建一个默认路径。需要特别注意的是：默认网关必须是电脑自己所在的网段中的IP地址，而不能填写其他网段中的IP地址。

#### **1.1.1 网关举例说明：**

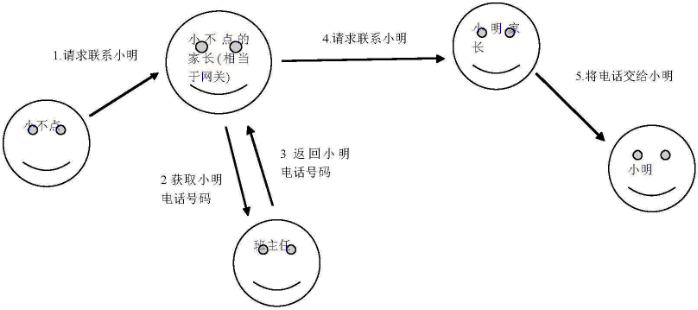
如下图2.1。假设你的名字叫小不点(很小)，你住在一个大院子里，你的邻居有很多小伙伴，父母是你的网关。当你想跟院子里的某个小伙伴玩，只要你在院子里大喊一声他的名字，他听到了就会回应你，并且跑出来跟你玩。但是你家长不允许你走出大门，你想与外界发生的一切联系，都必须由父母（网关）用电话帮助你联系。假如你想找你的同学小明聊天，小明家住在很远的另外一个院子里，他家里也有父母（小明的网关）。但是你不知道小明家的电话号码，不过你的班主任老师有一份你们班全体同学的名单和电话号码对照表，你的老师就是你的DNS服务器。于是你在家里和父母有了下面的对话：

小不点：妈妈(或爸爸),我想找班主任 查一下小明的电话号码行吗？

家长：好，你等着。（接着你家长给你的班主任挂了一个电话，问清楚了小明的电话）问到了，他家的号码是 211.99.99.99。

小不点：太好了！妈(或爸),我想找小明，你再帮我联系一下小明吧。

家长：没问题。（接着家长向电话局发出了请求接通小明家电话的请求，最后一关当然是被转接到了小明家家长那里，然后他家长把电话给转到小明）。就这样你和小明取得了联系。



 交换机/多接口网桥:

交换机和网桥的功能类似，基本上可以认为是一样的。如果真要说差异在哪里：两点，一是接口数量不同，交换机比网桥的接口数量多；二是功能的实现方式不同，功能是指交换机和网桥使用桥接技术，实现方式不同是指交换机通过专门的<集成电路/芯片>来实现桥接功能，而网桥则没有专门的处理芯，所以交换机比网桥更快更智能。

接口(interface):

路由器，交换机上那种直接插线的就称为接口。

---------------- 以上偏向硬件，以下偏向软件 ---------------

端口(port):

大家都知道标识WWW服务的是80，标识FTP服务的是21，这些数字就称为端口。比如www.baidu.com的IP是119.75.216.30， 那么访问www.baidu.com实质上就是访问了119.75.216.30的80端口119.75.216.30:80。

路由服务器:

提供路由<功能/服务>的服务器。如在Windows 2003 Server上加入路由服务器角色就能提供路由<功能/服务>，相当于一台路由器提供的路由<功能/服务>。

DHCP服务器:

（Dynamic Host Configuration Protocol, 动态主机配置协议） 提供DHCP<功能/服务>的服务器，是一个局域网的网络协议，使用UDP协议工作，主要有两个用途：1. 给内部网络或网络服务供应商自动分配IP地址；2. 给用户或者内部网络管理员作为对所有计算机作中央管理的手段。如在Windows 2003 Server上加入DHCP服务器角色就能提供DHCP<功能/服务>，相当于一台路由器提供的DHCP<功能/服务>。

DHCP的前身是BOOTP。BOOTP 原本是用于无磁盘主机连接的网络上面的：网络主机使用BOOT ROM而不是磁盘启动并连接上网络，BOOTP则可以自动地为那些主机设定TCP/IP环境。但BOOTP有一个缺点：在设定前须事先获得客户端的硬件地址，而且，与IP的对应是静态的。换而言之，BOOTP非常缺乏 "动态性"，若在有限的IP资源环境中，BOOTP的一一对应会造成非常严重的资源浪费。DHCP可以说是 BOOTP的增强版本，它分为两个部份：一个是服务器端，而另一个是客户端。所有的IP网络设定数据都由DHCP服务器集中管理，并负责处理客户端的DHCP要求；而客户端则会使用从服务器分配下来的IP环境数据。比较起BOOTP，DHCP透过"租约"的概念，有效且动态的分配客户端的 TCP/IP设定，而且，作为兼容考虑，DHCP 也完全照顾了 BOOTP Client的需求。DHCP 的分配形式为：首先，必须至少有一台 DHCP服务器工作在网络上面，它会监听网络的DHCP请求，并与客户端磋商TCP/IP 的设定环境。

DHCP有三种IP分配方式：1. Manual Allocation (人工分配)：获得的IP也叫静态地址，网络管理员为某些少数特定的在网计算机或者网络设备绑定固定IP地址，且地址不会过期；2. Automatic Allocation (自动分配)：一旦DHCP 客户端第一次成功的从DHCP 服务器端租用到 IP 地址之后，就永远使用这个地址；3. Dynamic Allocation (动态分配)：当DHCP客户端第一次从DHCP 服务器端租用到IP 地址之后，并非永久的使用该地址，只要租约到期，客户端就得释放(release)这个IP 地址，以给其它工作站使用。当然，客户端可以比其它主机更优先的更新（renew)租约，或是租用其它的 IP 地址。动态分配显然比手动分配更加灵活，尤其是当您的实际IP 地址不足的时候。

NAT服务器:

（Network Address Translation, 网络地址转换）提供NAT<功能/服务>的服务器：就是把在内部网络中使用的IP地址转换成外部网络中使用的IP地址，把不可路由的IP地址转化成可路由的 IP地址，对外部网络隐蔽内部网。NAT不仅完美地解决了lP地址不足的问题，而且还能够有效地避免来自网络外部的攻击，隐藏并保护网络内部的计算机。如在Windows 2003 Server上加入NAT服务器角色就能提供NAT<功能/服务>，相当于一台路由器提供的NAT<功能/服务>。

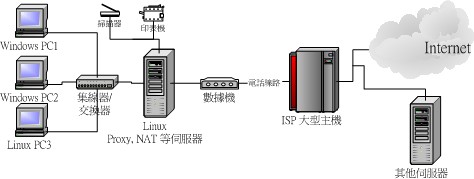
①如下图,这个client 的gateway 设定为NAT主机，所以当要连上Internet 的时候，该封包就会被送到NAT 主机啦，这个时候的封包Header 之source IP 为192.168.1.100 ；

②而透过这个NAT 主机，她会将client 的对外联机封包的source IP ( 192.168.1.100 ) 伪装成 ppp0 ( 假设为拨接情况)这个接口所具有的公有IP 啰，因为是公有IP 了，所以这个封包就可以连上Internet 了！同时 NAT 主机会记忆这个联机的封包是由哪一个( 192.168.1.100 ) client 端传送来的；

③由Internet 传送回来的封包，当然由NAT 主机来接收了，这个时候， NAT 主机会去查询原本记录的路由信息，并将目标 IP 由 ppp0 上面的公共 IP 改回原来的192.168.1.100 ；

④最后则由 NAT 主机将该封包传送给原先发送封包的 Client ！



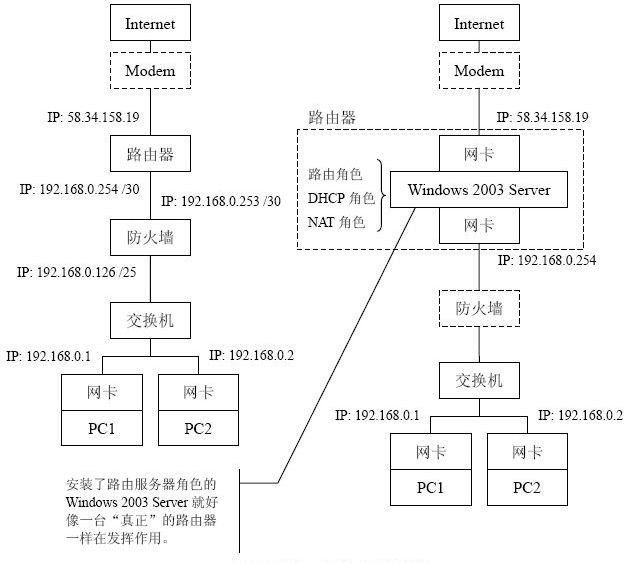


### 1.1.2 硬件设备与软件(虚拟)设备

先举个例子，见下图，路由器和带双网卡并安装路由及防火墙<服务/功能>的服务器，组建一个多主机网络。其中安装在Windows 2003 Server上的路由<服务器角色/服务/功能>不正是一台“虚拟路由器”吗？使得Windows 2003 Server像一台真的路由器一样在这个网络中发挥作用。实际上，如果在这台Windows 2003 Server上再装上软件防火墙，用这样的软件方式也不正是虚拟出了一个网路上确实存在的硬件防火墙一样吗？

所以，从功能出发来看，<物理/实际/硬件>设备和<虚拟/软件>设备是同一的，可以将<虚拟/软件>设备看成是实实在在的硬件设备。但是，从设计和性能出发，两者有质的不同，前者是由专门设计的电子集成电路和微程序来实现功能，而后者是由软件编程交由CPU处理来完成，前者的性能远胜过后者。

当这些概念清晰以后，再来看虚拟机以及虚拟设备就一目了然了。我们大可以将以软件方式虚拟出的硬件设备大大方方的放在我们的网络结构中，就像他们真实存在一样。



### 1.1.3 VMware Workstation的虚拟网络设置

不同虚拟交换机应用在不同的联网模式下，vmware提供了Bridged、NAT、host-only、custom四种模式，下面分别介绍其具体分配：

#### 1.1.3.1 VMware虚拟硬件介绍：

VMnet0：这是VMware用于虚拟桥接网络下的虚拟交换机；

VMnet1：这是VMware用于虚拟Host-Only网络下的虚拟交换机；

VMnet8：这是VMware用于虚拟NAT网络下的虚拟交换机；

VMnet2~VMnet7及VMnet9~VMnet19：是VMware用于虚拟自定义custom网络下的虚拟交换机；

VMware Network Adapter VMnet1：这是宿主机用于与Host-Only虚拟网络进行通信的宿主机使用的虚拟网卡；

VMware Network Adapter VMnet8：这是宿主机用于与NAT虚拟网络进行通信的宿主机使用的虚拟网卡；

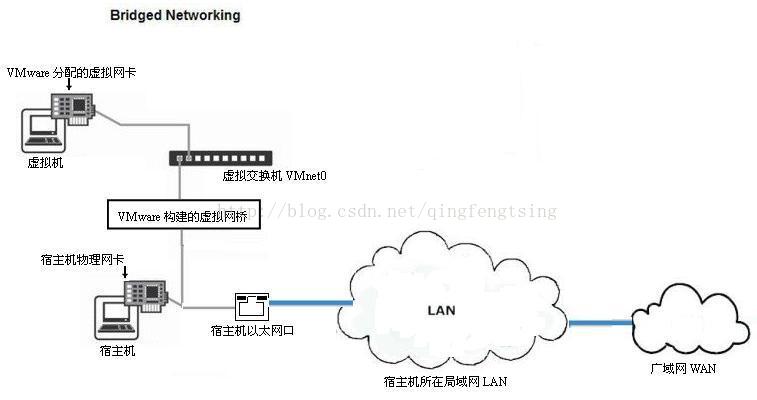
eth0：是虚拟机第一块网卡，相应的ethX就是虚拟机的第X块网卡。

#### 1.1.3.2 Bridged桥接模式：

VMware在桥接模式下，虚拟机使用VMware为该虚拟机分配的虚拟网卡，宿主机使用自身的物理网卡（有线或无线都行），并且默认使用虚拟交换机VMnet0来连接虚拟机的虚拟网卡和宿主机的物理网卡。在此模式下没有局域网动态地址分配DHCP服务器，也没有网络地址转换ＮＡＴ服务器，虚拟交换机没有连接DHCP服务器和NAT服务器。宿主机的网口（插网线的那个口）与宿主机物理网卡相连，同时也就和虚拟机的虚拟网卡相连，也就是和虚拟交换机相连，所以虚拟机相当于在宿主机所在局域网内的一个单独的主机，他的行为和宿主机是同等地位的，没有依存关系。原理图如下：

使用桥接模式的虚拟系统和宿主机器的关系，就像连接在同一个Hub上的两台电脑。想让它们相互通讯，你就需要为虚拟系统配置IP地址和子网掩码，否则就无法通信。

如果你想利用VMware在局域网内新建一个虚拟服务器，为局域网用户提供网络服务，就应该选择桥接模式。

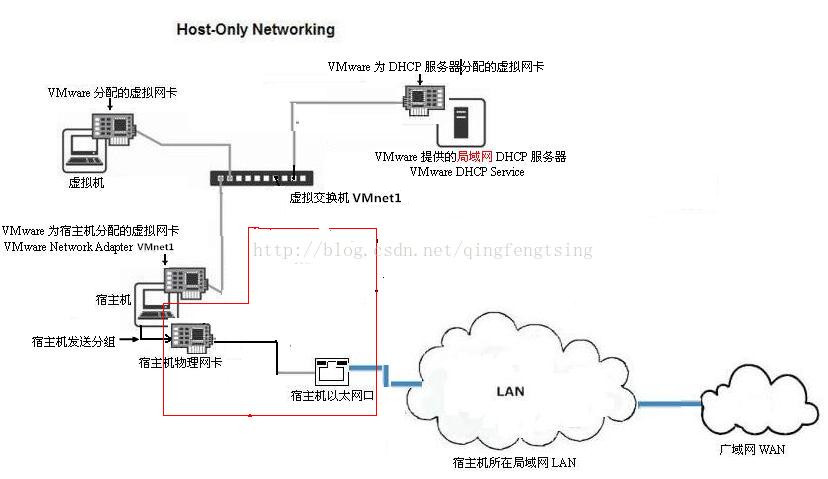


#### 1.1.3.3 Host-Only方式

在某些特殊的网络调试环境中，要求将真实环境和虚拟环境隔离开，这时你就可采用Host-only模式。在Host-only模式中，所有的虚拟系统是可以相互通信的，但虚拟系统和真实的网络是被隔离开的。虚拟系统的TCP/IP配置信息 (如IP地址、网关地址、DNS服务器等)，都是由VMnet1(Host-only)虚拟网络的DHCP服务器来动态分配的。如果你想利用VMware 创建一个与网内其他机器相隔离的虚拟系统，进行某些特殊的网络调试工作，可以选择Host-only模式。

在Host-Only网络中，Host-Only网络被用来设计成一个与外界隔绝的网络，其实Host-Only网络和NAT网络非常相似，唯一不同的地方就是在Host-Only网络中，没有用到NAT服务，没有服务器为VMnet1虚拟交换机做路由，它当然就没有办法访问Internet啦，可是如果我的Host要和虚拟机通信呢？怎么办？对了！当然就要用到VMware Network Adapter VMnet1这块虚拟网卡了。

事实上，如果我们足够BT，也可以在Host上来以VMware Network Adapter VMnet1虚拟网卡来做路由，比如，我们可以用Windows 2000的RRAS来做，这样的话，处于Host-Only网络下的Guest（虚拟机）就又可以上网了，他们只需要使用route add命令把自己的Default Gateway改成VMware Network Adapter VMnet1网卡即可，不过这样不推荐，也没有必要。如果仅仅想让虚拟机访问互联网，nat方式是最简单的，基本上不需要做什么操作，就能自动连接上。



注意：其实跟nat模式的图片是类似的，只是少了nat服务。

所以host-only上不了外网，只能实现主机的VMnet1网卡和虚拟机的虚拟网卡通信。

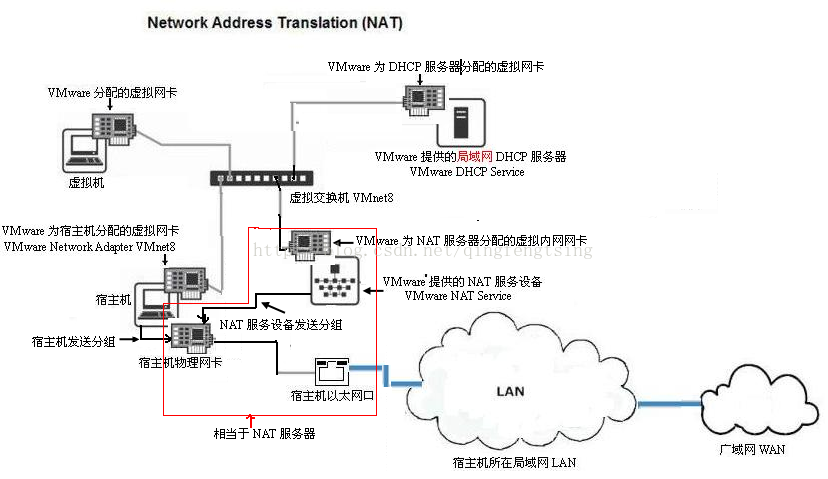
#### 1.1.3.4 NAT网络地址转换模式：

在NAT网络中，会使用到VMnet8虚拟交换机，Host上的VMware Network Adapter VMnet8虚拟网卡被连接到VMnet8交换机上，来与虚拟机进行通信，但是VMware Network Adapter VMnet8虚拟网卡仅仅是用于和VMnet8虚拟交换机网段通信用的，它并不为VMnet8网段提供路由功能，处于虚拟NAT网络下的虚拟机是使用虚拟的NAT服务器连接的Internet的。

这时候，你的虚拟机和Host就可以实现互访了，并且如果你的Host此时已经连接到了Internet，那么你的虚拟机也就可以连上Internet 了。那么VMware Network Adapter VMnet8虚拟网卡在这里扮演了一个什么角色呢？它仅仅是为Host和NAT虚拟网络下的虚拟机通信提供一个接口，所以，即便Disable掉这块虚拟网卡，虚拟机仍然是可以上网的，只是Host无法再访问VMnet8网段而已，当然也就无法访问虚拟机了。

这种方式的时候，主机需要开启**vmdhcp**和**vmnat**服务。

使用NAT模式，就是让虚拟系统借助NAT(网络地址转换)功能，通过宿主机器所在的网络来访问公网。也就是说，使用NAT模式可以实现在虚拟系统里访问互联网。由于NAT服务协议对外部网络隐蔽内部网络，因此虚拟系统可以通过NAT服务器访问宿主机所在局域网中的其他真实主机，而反过来却不行，即宿主机以及宿主机所在网络的其他主机（更何况其它外网）不能通过NAT服务器访问虚拟机上的操作系统。（但是可以通过修改相关路由表，以让虚拟机和外网中除宿主机以外的其他主机通过VMnet1/8虚拟网卡互访。如4.3.4.图中的红色字体。对宿主机网络来说，VMnet1/8网卡被视为通往虚拟网络的网关，反之亦然）采用NAT模式最大的优势是虚拟系统接入互联网非常简单，你不需要进行任何其他的配置，只需要宿主机器能访问互联网即可。 如果你想利用VMware安装一个新的虚拟系统，在虚拟系统中不用进行任何手工配置就能直接访问互联网，建议你采用NAT模式。



注意：红色的方框是nat服务器，nat服务器有两个网卡一个是虚拟内网网卡，一个是宿主机的物理网卡。禁用VMware Network Adapter VMnet8，虚拟机仍然可以上网，ping通主机，但是主机ping不通虚拟机的网卡。

备注：

VMnet8存在的意义在于实现主机和虚拟机的通信。弥补了NAT协议中外网不能访问局域网的缺点。

### 1.1.4 NAT模式的虚拟网络设置

虚拟网络组件

1.虚拟<网卡/网络适配器>：安装一个虚拟PC时自动安装一块虚拟<网卡/网络适配器>，可添加多块虚拟网卡。

2.虚拟交换机：安装VM时就自动安装了20台虚拟交换机。

3.虚拟网桥：只在虚拟网卡连接到VMnet0 (default Bridged)虚拟交换机上才应用虚拟网桥。

4．虚拟DHCP服务器：在虚拟网卡连接到VMnet1或VMnet8虚拟交换机时，自动安装运行虚拟DHCP服务器并连接至这台虚拟交换机。

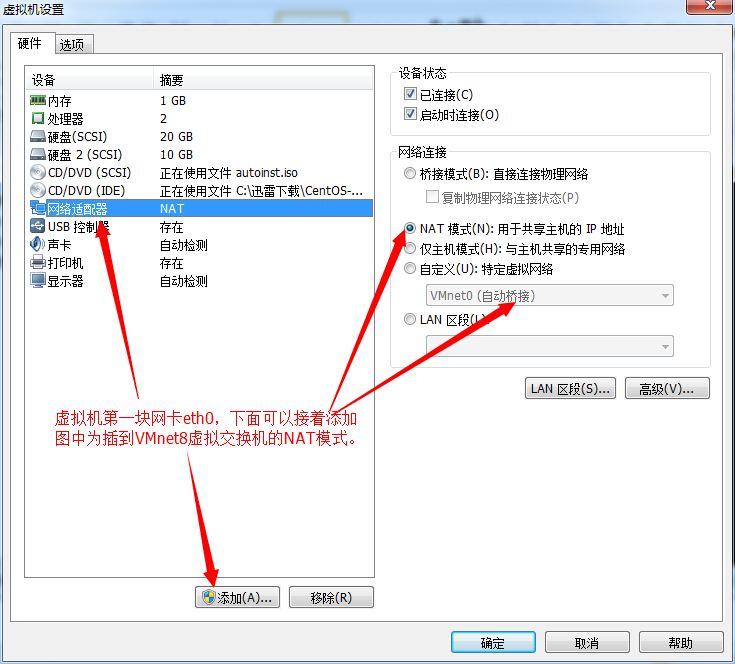
5.虚拟NAT服务器：只在虚拟网卡连接到VMnet8 (NAT)虚拟交换机时才自动安装运行虚拟NAT服务器，并连接至这台虚拟交换机。

6.主机虚拟<网卡/网络适配器>：相当于一块确实存在的网卡，是连接主机和虚拟交换机用的网卡。

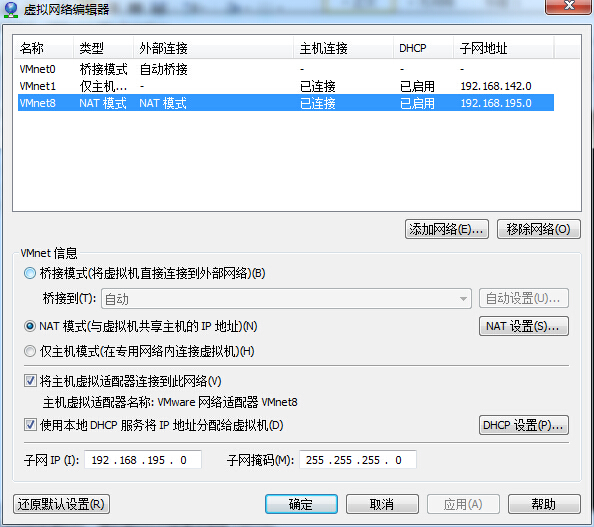
#### 1.1.4.1 VMware和Network Adapter VMnet8设置

看了上面这些信息后，相信你已经对NAT模式比较清楚了，其实就是VMware提供了软件模拟硬件，包括虚拟机网卡，虚拟机交换机，虚拟机网关，主机网卡等，当然这些都是虚拟的，了解了相关的概念后，再配置NAT模式，就相对简单了。

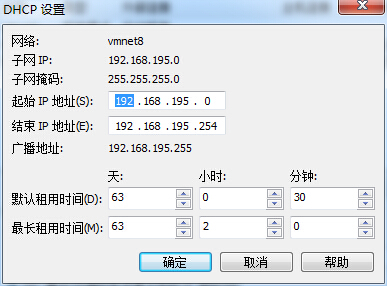
配置VMware的NAT模式，实际上非常简单，第一步首先进入VMware的虚拟机设置，选择网络适配器，选择NAT模式如下图：



第二步，不要启动虚拟机，进入编辑中虚拟网络编辑器，选择VMnet8交换机，右边是自定义模式的交换机，可以通过添加网络来添加。注意最下边的子网IP，添加你要分配的虚拟网段，如192.18.195.0，子网掩码通常都是255.255.255.0 如下图



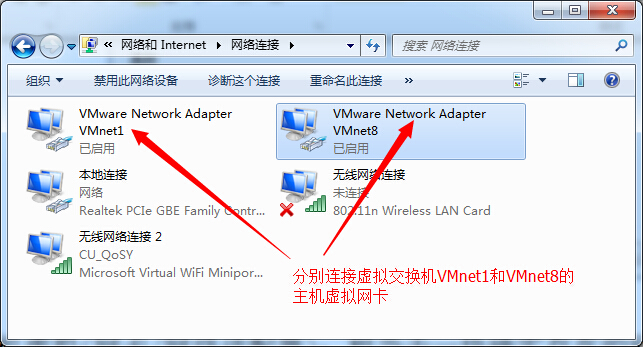
第三步设置DHCP服务器，分配ip号段，同时添加租用时间，最长63天。



第四步，设置NAT服务器，添加网关。



第五步，到此，VMware部分设置完了，开始设置主机的虚拟网卡，VMware为主机提供了2块虚拟网卡VMware Network Adapter VMnet1和VMware Network Adapter VMnet8，这里我们用默认的第二块，为其分配IP地址和子网掩码，DNS等。其地址应设置与虚拟交换机同网段。



到此，主机和虚拟机的工作模式配置全部完成，下面是进行虚拟机内部网卡的配置，根据不同的操作系统，进行相应的设置，已达到主机和虚拟机之间的通信，以及虚拟机和外网的通信。

## 1.2、CentOS虚拟机网卡eth0设置

在Linux中通过修改网络配置文件的方式来配置网络，因此不仅需要知道配置哪个文件，还要知道文件中每个配置参数的功能。不同版本的linux中配置文件的名称和路径略有不同，但功能是一样的，下面就CentOS为例，为大家介绍，网络配置文件的设置。Linux中网络相关的主要的几个配置文件为：

/ect/hosts #配置主机名（域名）和IP地址的对应

/etc/sysconfig/network #配置主机名和网关

/etc/resolv.conf #配置DNS服务器

/etc/sysconfig/network-scripts/ifcfg-eth0 #eth0配置文件

在一个计算机系统中，可以有多个网卡，分别对应多个网络接口配置文件，在/etc/sysconfig/network-scripts/目录下，依次编号的文件是ifcfg-eth0，ifcfg-eth1，...，ifcfg-eth<X>。常用的是ifcfg-eth0，表示第一块网卡eth0的配置文件。

下面就这些配置文件做详细的说明：

### 1.2.1 ifcfg-eth0设置

该文件位于/etc/sysconfig/network-scripts/ifcfg-eth0

DEVICE=eth0 #物理设备文件名称

HWADDR=00:0c:29:8f:f6:ba #eth0的mac地址

TYPE=Ethernet #配置文件接口类型，通常为Ethernet

UUID=7caaef42-471f-41d0-b59c-fd945f6040e6 #设备UUID

ONBOOT=yes #系统启动时是否激活该设备（yes/no）

NM\_CONTROLLED=yes #是否由Network Manager控制该网络接口。

#修改保存后立即生效，无需重启。

#被其坑过几次，建议一般设为no。

BOOTPROTO=none #系统启动地址协议

#none：不使用启动地址协议

#bootp：BOOTP协议

#dhcp：DHCP动态地址协议

#static：静态地址协议

IPADDR=192.168.195.3 #给虚拟机分配ip地址

NETMASK=255.255.255.0 #子网掩码

GATEWAY=192.168.195.1 #网关地址

DNS1=192.168.195.1 # DNS地址。

#当PEERDNS为yes时

#会被写入/etc/resolv.conf中。

DNS2=8.8.8.8

IPV6INIT=yes #是否支持IPv6

USERCTL=no #用户权限控制。yes：非root用户允许控制该网络接口，no则否定。

BROADCAST=192.168.195.255 #通常广播地址

PEERDNS=yes #是否指定DNS。如果使用DHCP协议，默认为yes。

#yes：如果DNS设置，修改/etc/resolv.conf中的DNS

#no：不修改/etc/resolv.conf中的DNS

MTU=1500     #最大传输单位的设置

注意事项：

这个文件是用来设置网卡参数，这个文件内的所有设置，基本上就是bash的变量设置规则，请注意每个变量（左边的英文）都应该要大写，否则我们的scropt会误判。几个重要的项目设置说明：

DEVICE ：这个设置值后面接的设备代号必须与文件名（ifcfg-eth0）的设备号相同，否则会显示找不到设备名（网络的设备文件名为ethX，X从0开始eth0为第一个网络设备以此类推）

BOOTPROTO ：启动该网络接口时，使用那种协议，none(引导时不使用协议)，static(静态分配，手动设置IP时用)，dhcp(DHCP协议，自动设置IP时用)，bootp(bootp协议)。

GATEWAY ： 代表整个主机系统的默认网关，不要有重复设置的情况发生。也就是说当你有ifcfg-eth0,ifcfg-eth1多个文件时(即多个网卡)，只在其中一个文件里设置GATEWAY即可。

还有二个参数比较特殊，就是ipv6和peerdns，ipv6如果设置no，有可能不能进行桥接，peerdns最好配合真实的dns设置，否则容易导致dns设置失败，无法访问外网。

### 1.2.2 network设置

该文件位于/etc/sysconfig/network

NETWORKING=yes        #yes启动网络no关闭网络

HOSTNAME=hostname      #主机名

GATEWAY=192.168.1.1    #默认网关

这个文件的主要功能是设置主机名(HostName)及能否启动网络(Network) 更改了主机名后重启系统后才能生效。

### 1.2.3 resolv.conf设置

该文件位于 /etc/resolv.conf

nameserver 175.191.127.2        #主DNS

nameserver 175.191.127.6        #次DNS

这个文件是用来设置DNS IP(域名解析服务器)的文件，常常有人提到，我已经ping到外部计算机的公共IP 了，为何输入网址无法连机？通常发生错误的原因就是文件里的设置不正确。

－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－

**ＲＨＥＬ　６.０的ＤＮＳ不能直接在这个文件配置，要在ifcfg-eth0哪里配置。DNS1=\*.\*.\*.\***

### 1.2.4 hosts设置

该文件位于/etc/hosts

127.0.0.1   hostname(主机名)  localhost.domain.domain  localhost（别名）

这个文件可以记录计算机的IP对应的主机的名称或主机的别名。

### 1.2.5 eth0自动更名为eth1的处理办法

当使用linux更换了网卡，或者将虚拟机从一台电脑拷贝到另一台电脑时，由于设备网卡mac地址改变，但是系统配置文件

/etc/udev/rules.d/70-persistent-net.rules中仍然保留了老网卡的内容，新网卡则被识别为eth1。所以会出现eth0变为 eth1的情况，其实eth0变为eth1也没什么问题，但是变得不完全，导致上不了网。解决办法：

1、在配置文件/etc/udev/rules.d/70-persistent-net.rules中将表示新网卡的行注释掉，然后将表示老网卡的行中mac地址改成新网卡的mac地址，重启即可。

2、若/etc/sysconfig/network-script/ifcfg-eth0中有指定了mac地址，则将此mac地址改为新网卡的mac地址，或者干脆删掉这一行。

然后重启系统或者service network restart

## 1.3、VMware Workstation分辨率问题

这个问题，通常在硬件上直接安装系统是不会出现的，只要你能正确安装驱动就没有问题了，但这个问题在虚拟机上就不是那么顺利了，因为虚拟机显卡的模拟不完善，导致没有合适的驱动来解决这个问题，尽管，安装一些教程，设置了，但全屏还是有问题，下面就来探讨这个问题。

这本来应该不是一个问题，但由于虚拟显卡驱动的问题，使这个问题变得非常难解决，网上给出的方案，似乎并不是都有效，而且做法非常多，没有一个统一的做法，这是让人不能接受的，所以，有必要对这个问题，研究一下。

首先你要做的两件事，一个是安装VMware tools，一个是安装显卡驱动VMsvga2\_v1.2.5\_OS\_10.9.pkg，是的就按照这个名字搜索下载就好了。

这样首先，你就可以调整分辨率，但不见得有适合你的分辨率，网上给出的教程，有两个方法，来解决合适分辨率的问题，但这两个方法我都试过了，都不能完美解决问题，也许和具体环境有关系，但毕竟这是最糟糕的情况了。

第一种方法是从VMware下手，更改其默认分辨率。



上图中，大家已经看到了，3D图形加速不支持，这个目前，还没有办法解决，应该也是目前虚拟机显卡存在的最大问题。然后在监视器一栏中，选择指定监视器设置，更改其最大分辨率，通过测试，要修改成比实际你要用的大的分辨率，这个例子中，实际我们使用的是1360x768，但是这个地方我们写的确实1920x1080，这样在mac中，我们发现才会出现1360x768，如果直接写1360x768，显示器设置中最高只有1280x768，当然如果，你直接修改VMware的vmx文件，应该是同样的效果。

svga.autodetect = "FALSE"

svga.maxWidth = "1920"

svga.maxHeight = "1080"

svga.vramSize = "134217728"

mks.enable3d = "TRUE"

svga.graphicsMemoryKB = "524288"

这里还有一个问题，就是，不知道这个1360x768是不是我们修改了mac的启动配置文件，这个没有测试，分析应该有关，因为，这里出现的分辨率并不全，所以。。。。。。。。



而且还有比较奇怪的事，就是我在mac启动文件设置的分辨率是1280x1024，也不是这个1366x768，而且这个分辨率也没有出现，所以这个文件从结果来看，根本就没有起作用。我是通过缩放，实现了全屏，这样的结果，有点让人费解，不过对于这个大BUG能有这样的结果，还是很满意的。



也就是说，从实际结果来看，显卡驱动和虚拟机tools这两个必须安装，然后指定一个比实际分辨率大的分辨率，通过缩放就可以实现了，当然，如果正好有你要的分辨率，那么恭喜你，你可以直接使用，而不用设置这些了，如果没有，就按照我的测试方法，应该也可以。

但还有一种可能就和显卡是否有关，还不知道，因为，虚拟机的显卡，毕竟是通过物理显卡实现的，不过，没有必要弄清楚这些了。反正是一顿乱搞，实现了全屏。

## 1.4、ntfs-3g插件的使用

网上的教程几乎都有或多或少的问题，连LMAP的视频中都有问题，这里就基础安装CentOS会出现的几个问题说明一下。下载这个插件，解压，然后安装比较简单，没什么难点。就是执行两个命令就可以了。现在，你可以直接用cifs协议了。后边会讲。

解压缩用

tar -zxvf \*\*\*\*\*\*.taz

进入解压目录执行

./configure && make && make install

就可以安装了。

但我们安装的CentOS是基础安装，所以没有安装gcc，也就没法make了，所以还必须安装gcc，这个东西，如果后安装，其实也不是麻烦，而是相应的配置比较费劲，你需要了解yum方式安装，通常我们不建议直接安装gcc安装包，因为它需要很多依赖包，而是通过yum这种软件仓库的方式安装比较方便。

Yum（全称为 Yellow dog Updater, Modified）是一个在Fedora和RedHat以及SUSE、CentOS中的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，无须繁琐地一次次下载、安装。

Yum分为两种安装方式，一种是远程服务器安装，这个视本地网络和服务器端速度而定，通常速度比较慢。一般都是采用本地服务器安装。

### 1.4.1repo配置文件介绍

### 1.4.2设置本地镜像作为yum源

默认软件包来源是从网络上下载进行安装的，现在我们要做的是修改来源为本地进行安装。

【设置本地镜像作为yum源】

1、mkdir /mnt/cdrom

2 mount /dev/cdrom /mnt/cdrom  #挂着镜像，光盘

3 mkdir /mnt/cdrom-iso

4 cp -Rf /mnt/cdrom/\* /mnt/cdrom-iso #将光盘内的内容copy到某个路径下

5 vi /etc/yum.repos.d/CentOS-debuginfo.repo #编辑repo文件

添加一下内容：

[CentOS\_6\_iso]

name=CentOS\_6\_iso

baseurl=file:///mnt/cdrom-iso/ #本地源

gpgcheck=0

enable=1

gpgkey=file:///mnt/cdrom-iso/RPM-GPG-KEY-CentOS-6 #该处的gpgkey文件应该改成iso镜像中的文件名

这里需要注意一下，yum.repos.d目录下通常有多个repo文件，我们只保留要执行的一个，将其他备份或者移出，通常用mv或者cp命令。

### 1.4.3实例：创建自己的repo文件，安装gcc

1.3.1首先挂载光驱，复制光盘内容

mkdir /mnt/cdrom

mkdir /mnt/cdrom-iso

mount /etc/sr0 /mnt/cdrom

cp –Rf /mnt/cdrom/\* /mnt/cdrom-iso #复制光驱全部内容

umount /etc/sr0 #卸载光驱

1.3.2创建自己的repo文件

首先把/etc/yum.repos.d/目录下所有文件备份移出。

执行

mkdir /mnt/repo-back #建立备份目录

mv /etc/yum.repos.d/\* /mnt/repo-back #备份repo文件

然后建立自己的repo配置文件,注意这个文件只能在这个目录中。

新建一个文件，名为：caojun.repo 内容如下：

[caojun]

name=caojun

baseurl=file:///mnt/cdrom-iso \\查找路径

enable=1 \\是否检查GPG-KEY，0为不检查，1为检查

gpgchack=0 \\是否启用yum源，0为不启用，1为启用

touch caojun.repo

vi /etc/yum.repos.d/caojun.repo

建好后存盘退出。

通常这样本地源的yum就配置好了，可以安装软件了。

1.3.3 安装gcc

既然源已经配置好了，可以先执行查找gcc:

yum search gcc

当然我们的是安装光盘，通常不需要查找的。

直接执行

yum –y install gcc

通常会出现下面的错误

Public key for \*.rpm is not installed

解决方法：此时要导入rpm的签名信息即可

以root登录，执行下面命令

# rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release

根据我的Linux版本是CentOS 6.5

于是我执行下面命令

#rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-6

或者直接在caojun.repo中加入

gpgkey=file:///mnt/cdrom-iso/RPM-GPG-KEY-CentOS-6

这时在执行yum –y install gcc 就没有问题了。

### 1.4.4 安装ntfs-3g

gcc安装完成后就可以，编译安装ntfs-3g了

执行

./configure && make && make install

这时在插入移动硬盘就可以识别了，当通常只能读，不能写，呵呵，

废了尼玛半天劲，居然只能读，哈哈。

### 1.4.5使用NTFS-3G

正确安装后，我们需要通过ntfs-3g来加载NTFS分区。

获得NTFS分区设备名,执行下面的命令

fdisk -l | grep NTFS

结果如下所示：

/dev/sdb1   1   10443   83883366  7  HPFS/NTFS

其中第一个字段/dev/sdb1即为所需要的分区设备名  
建立装载点

          mkdir /mnt/ntfs-p1   #建立个目录

如果需要，请给予特定的访问权限

          chmod 755 /mnt/ntfs-p1

临时装载NTFS分区

可以使用下面的命令以读写方式临时装载一个NTFS分区到装载点

mount  -t ntfs-3g <NTFS Partition>  <Mount Point>

其中：

<NTFS Partition> ------ NTFS所在分区的设备名，如/dev/sdb1

<Mount Point>  ------ 装载点，如/mnt/ntfs-p1

例如：

     mount  -t ntfs-3g /dev/sdb1 /mnt/ntfs-p1

或者:  ntfs-3g /dev/sdb1 /mnt/ntfs-p1

系统启动时装载NTFS分区

编辑/etc/fstab

          vi /etc/fstab

在文件最后增加如下格式的行  
<NTFS Partition>  <Mount Point>  ntfs-3g  defaults  0  0

其中：

<NTFS Partition> ------ NTFS所在分区的设备名，如/dev/sdb1

<Mount Point> ------ 装载点，如/mnt/ntfs-p1

例如：

     /dev/sdb1  /mnt/ntfs-p1  ntfs-3g  defaults  0  0

保存文件后重启系统或简单的执行下面的命令，即可装载NTFS分区到指定的装载点

     mount  -a

## 1.5、cifs协议

上边介绍的ntfs系统访问方法，由于一些版权问题，只能读，不能写，显然不能满足我们的要求，所以我们有必要学习cifs协议，来进行linux无缝访问windows系统的ntfs系统。

### 1.5.1、smb协议的增强版—cifs协议

NAS存储系统中目前支持都只NFS协议与CIFS协议，困惑与samba、ftp的区别，百度之，得如下结论，samba项目是基于SMB协议的逆向项目，为了使得非Windows资源可以供Windows操作系统使用，而SMB协议即为目前的CIFS协议。

ftp是一种文件传输协议，通过命令交互，获取服务器保存的文件数据，传输ftp命令以及文件数据。

SMB最初是IBM的贝瑞·费根鲍姆（Barry Feigenbaum）研制的，其目的是将DOS操作系统中的本地文件接口“中断13”改造为网络文件系统。后来微软对这个发展进行了重大更改，这个更改后的版本也是最常见的版本。微软将SMB协议与它和3Com一起发展的网络管理程序结合在一起，并在Windows for Workgroups和后来的Windows版本中不断加入新的功能。

　　SMB一开始的设计是在NetBIOS协议上运行的（而NetBIOS本身则运行在NetBEUI、IPX/SPX或TCP/IP协议上），Windows 2000引入了SMB直接在TCP/IP上运行的功能。在这里我们必须区分SMB协议和运行在这个协议上的SMB业务，以及NetBIOS和使用SMB作为认证隧道的DCE/RPC业务。此外我们还要区分主要（但不仅仅）直接使用NetBIOS数据报的“网络邻居”协议。

　　1996年，约于升阳推出WebNFS的同时[1]，微软提出将SMB改称为Common Internet File System[2]。此外微软还加入了许多新的功能，比如符号链接、硬链接、提高文件的大小。微软还试图支持直接联系，不依靠NetBIOS，不过这个试图依然处于尝试阶段，并需要继续完善。微软向互联网工程工作小组提出了部分定义作为互联网草案[3]。不过这些提案现在均已过期。

　　由于SMB协议对于与占主要地位的Microsoft Windows平台通讯时的重要性，而目前该平台使用的SMB协议与初始的版本相比有巨大的改变，因此Samba项目就是被创立来逆向工程来提供一个与SMB软件兼容的自由软件，使得非微软操作系统也能够使用它。

　　在Windows Vista中微软又推出了Server Message Block 2.0。[1][2]

SMB（Server Message Block）通信协议是微软（Microsoft）和英特尔(Intel)在1987年制定的协议，主要是作为Microsoft网络的通讯协议。SMB 是在会话层（session layer）和表示层（presentation layer）以及小部分应用层（application layer）的协议。SMB使用了NetBIOS的应用程序接口 （Application Program Interface，简称API）。另外，它是一个开放性的协议，允许了协议扩展——使得它变得更大而且复杂；大约有65个最上层的作业，而每个作业都超过120个函数，甚至Windows NT也没有全部支持到，最近微软又把 SMB 改名为 CIFS（Common Internet File System），并且加入了许多新的特色。

　　SMB协议是基于TCP－NETBIOS下的，一般端口使用为139，445，关于NETBIOS协议这里就不介绍了。

　　NetBios协议头可以是:

　　typedef struct

　　{

　　unsigned char msg\_type;

　　unsigned char flags;

　　unsigned short length;

　　}

　　netbios\_ss\_hdr\_t;

（IBM SMB：Server Message Block protocol）

　　服务器信息块（SMB）协议是一种IBM协议，用于在计算机间共享文件、打印机、串口等。SMB 协议可以用在因特网的TCP/IP协议之上，也可以用在其它网络协议如IPX和NetBEUI 之上。

　　SMB 一种客户机/服务器、请求/响应协议。通过 SMB 协议，客户端应用程序可以在各种网络环境下读、写服务器上的文件，以及对服务器程序提出服务请求。此外通过 SMB 协议，应用程序可以访问远程服务器端的文件、以及打印机、邮件槽（mailslot）、命名管道（named pipe）等资源。

　　在 TCP/IP 环境下，客户机通过 NetBIOS over TCP/IP（或 NetBEUI/TCP 或 SPX/IPX）连接服务器。一旦连接成功，客户机可发送 SMB 命令到服务器上，从而客户机能够访问共享目录、打开文件、读写文件，以及一切在文件系统上能做的所有事情。

从 Windows 95 开始，Microsoft Windows 操作系统（operating system）都包括了客户机和服务器 SMB 协议支持。Microsoft 为 Internet 提供了 SMB 的开源版本，即通用 Internet 文件系统 （CIFS）。与现有 Internet 应用程序如文件传输协议（FTP）相比， CIFS 灵活性更大。对于 UNIX 系统，可使用一种称为 Samba 的共享软件。

CIFS (Common Internet File System)通用Internet文件系统,属于微软公司的一种私有协议。

CIFS是一个新提出的协议，它使程序可以访问远程Internet计算机上的文件并要求此计算机的服务。CIFS使用客户/服务器模式。客户程序请求远在服务器上的服务器程序为它提供服务。服务器获得请求并返回响应。CIFS是公共的或开放的SMB协议版本，并由Microsoft使用。SMB协议现在是局域网上用于服务器文件访问和打印的协议。象SMB协议一样，CIFS在高级运行，而不象TCP/IP协议那样运行在底层。CIFS可以看做是现丰应用程序协议如文件传输协议和超文本传输协议的一个实现。

CIFS可以使您达到以下功能：

1.访问服务器本地文件并读写这些文件

2.与其它用户一起共享一些文件块

3.在断线时自动恢复与网络的连接 (--interop, 往往会有error信息提示, 不了解可能以为是bug)

4.使用西欧字符文件名 (--支持ASCII字符集, 西欧字符也在内..命名比MS OS下有个性多了:-))

一般来说，CIFS使用户得到比FTP更好的对文件的控制。它提供潜在的更直接地服务器程序接口，这比使用HTTP协议的浏览器更好。

CIFS是开放组标准而且已经被作为Internet应用程序标准被提交到IETF。

通用网际文件系统（CIFS）是微软服务器消息块协议（SMB）的增强版本，是计算机用户在企业内部网和因特网上共享文件的标准方法。CIFS 通过定义一种与应用程序在本地磁盘和网络文件服务器上共享数据的方式相兼容的远程文件访问协议使之能够在因特网上进行协作。CIFS 在 TCP/IP 上运行，利用因特网上的全球域名服务系统（DNS）增强其可扩展性，同时为因特网上普遍存在的慢速拨号连接优化。CIFS 利用重定向包可以通过网络发送至远端设备，而重定向器也利用 CIFS 向本地计算机的协议栈发出请求。

CIFS 提供如下关键特点：

l．文件访问的完整性：CIFS 支持一套通用的文件操作：打开、关闭、读，写以及搜索。CIFS 也支持文件和记录的锁定和解锁。CIFS 允许多个客户端访问和更新同一个文件，它通过提供文件共享和文件锁定功能来避免发生冲突。

2.为慢速链接优化：CIFS 已被优化过，使之能在慢速拨号线路上良好运行，结果就是为使用调制解调器访问因特网的用户提供了改善的性能。

3.安全性：CIFS 服务器既支持匿名传输，也支持对于指定文件的安全的、需要验证的访问。同时，也易于管理文件和目录的安全策略。

4.高性能和可扩展性：CIFS 服务器和操作系统高度集成，为最大化系统性能而优化。CIFS 支持 Windows 95 之后的所有微软平台。它也支持其它流行的操作系统，如 UNIX、VMS、Macintosh 、IBM LAN server等。

5.使用统一码（Unicode）文件名：文件名可以使用任何字符集，而不局限于为英语或西欧语言设计的字符集。

6.全局文件名：用户不必挂载远程文件系统也能直接查阅到全局有效名称，而不是只有本地意义的那些名称。

协议结构

CIFS 和 SMB 定义了许多客户端和服务器端的命令和消息。这些命令和消息大致可分为如下几类：

1、连接建立消息：包括开始或结束一个到服务器上共享资源的重定向连接的命令。

2、命名空间和文件处理消息：重定向器利用这个消息获得对服务器上文件的访问并对其进行读写操作。

3、打印消息：重定向器利用此消息向服务器上的打印队列发送数据和获得打印队列的状态信息。

4、其它消息：重定向器利用这些消息向邮槽和命名管道写入信息。

### 1.5.2、实例应用

那么cifs协议在centos中的具体应用，就是让centos直接访问windows共享文件系统，并且可直接访问windows文件系统，而不再受ntfs制约了。格式如下：

mount -t cifs -o username=administrator,password= ,rw //192.168.195.66/share /mnt

当然你可以直接写在fstab中，来跟随系统直接启动，格式当然要按照fstab来写了。

vi /etc/fstab,在文件的后边添加一行就好了。

//192.168.195.1/share/ /mnt/share cifs auto,username=omy,password= 0 0

这样就可以了。

这样还是有个缺点，即用户名和密码都以明文的方式暴露了，还可以这样：编辑/etc/fstab，按如下格式添加：

//server/share\_folder /mount\_point cifs credential=/root/credential

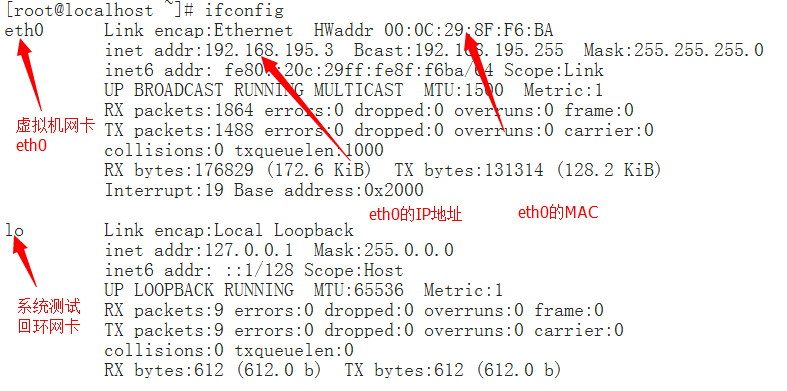
而/root/credential中内容如下：

useername=your\_name

password=your\_password

## 1.6、ifconfig

这个命令其实主要是用来查询虚拟系统中网卡的配置的，当然也可以用它来设置网卡的ip和子网掩码等，同事可以关闭和开启指定网卡，而不用重启系统。



查询系统网卡和IP可以直接执行

ifconfig或者ifconfig –a

设置网卡参数

ifconfig 网络端口 IP地址 hw MAC地址 netmask 掩码地址 broadcast 广播地址 [up/down]

ifconfig eth0 192.168.195.3 [hw 00:0C:29:8F:F6:BA netmask 255.255.255.0 broadcast 192.168.195.255 up]

上面是一条完整的ifconfig设置。包括了IP，Mac 子网掩码 广播地址 启动参数。

用ifconfig设置的网卡参数会立即生效，但系统重启不会保留。下面举几个例子，看看如何使用。

＊ 实例一：    
比如我们用ifconfig 来调试 eth0网卡的地址   
[root@linuxchao ~]#ifconfig eth0 down   
[root@linuxchao ~]#ifconfig eth0 192.168.1.99 broadcast 192.168.1.255 netmask 255.255.255.0   
[root@linuxchao ~]#ifconfig eth0 up   
[root@linuxchao ~]#ifconfig eth0   
eth0 Link encap:Ethernet HWaddr 00:03:0D:27:86:41   
inet addr:192.168.1.99 Bcast:192.168.1.255 Mask:255.255.255.0   
inet6 addr: fe80::203:dff:fe27:8641/64 Scope:Link   
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1   
RX packets:618 errors:0 dropped:0 overruns:0 frame:0   
TX packets:676 errors:0 dropped:0 overruns:0 carrier:0   
collisions:0 txqueuelen:1000   
RX bytes:491238 (479.7 KiB) TX bytes:86286 (84.2 KiB)   
Interrupt:5 Base address:0x8c00   
  
注解： 上面的例子我们解说一下；   
  
第一行：ifconfig eth0 down 表示如果eth0是激活的，就把它DOWN掉。此命令等同于 ifdown eth0；   
第二行：用ifconfig 来配置 eth0的IP地址、广播地址和网络掩码；   
第三行：用ifconfig eth0 up 来激活eth0 ； 此命令等同于 ifup eth0   
第四行：用 ifconfig eth0 来查看 eth0的状态；   
  
当然您也可以用直接在指令IP地址、网络掩码、广播地址的同时，激活网卡；要加up参数；比如下面的例子；   
[root@linuxchao ~]#ifconfig eth0 192.168.1.99 broadcast 192.168.1.255 netmask 255.255.255.0 up

＊ 实例二：在这个例子中，我们要学会设置网络IP地址的同时，学会设置网卡的物理地址（MAC地址）；   
  
比如我们设置网卡eth1的IP地址、网络掩码、广播地址，物理地址并且激活它；   
[root@linuxchao ~]#ifconfig eth1 192.168.1.252 hw ether 04:64:03:00:12:51 netmask 255.255.255.0 broadcast 192.168.1.255 up   
或   
[root@linuxchao ~]#ifconfig eth1 hw ether 04:64:03:00:12:51   
[root@linuxchao ~]#ifconfig eth1 192.168.1.252 netmask 255.255.255.0 broadcast 192.168.1.255 up   
  
其中 hw 后面所接的是网络接口类型， ether表示乙太网， 同时也支持 ax25 、ARCnet、netrom等，详情请查看 man ifconfig ；   
  
  
3.如何用ifconfig 来配置虚拟网络接口    
  
有时我们为了满足不同的需要还需要配置虚拟网络接口，比如我们用不同的IP地址来架运行多个HTTPD服务器，就要用到虚拟地址；这样就省却了同一个IP地址，如果开设两个的HTTPD服务器时，要指定端口号。   
  
虚拟网络接口指的是为一个网络接口指定多个IP地址，虚拟接口是这样的 eth0:0 、 eth0:1、eth0:2 ... .. eth1N。当然您为eth1 指定多个IP地址，也就是 eth1:0、eth1:1、eth1:2 ... ...以此类推；   
  
其实用ifconfig 为一个网卡配置多个IP地址，就用前面我们所说的ifconfig的用法，这个比较简单；看下面的例子；   
[root@linuxchao ~]#ifconfig eth1:0 192.168.1.251 hw ether 04:64:03:00:12:51 netmask 255.255.255.0 broadcast 192.168.1.255 up   
或   
[root@linuxchao ~]#ifconfig eth1 hw ether 04:64:03:00:12:51   
[root@linuxchao ~]#ifconfig eth1 192.168.1.251 netmask 255.255.255.0 broadcast 192.168.1.255 up   
  
注意：指定时，要为每个虚拟网卡指定不同的物理地址；   
  
在 Redhat/Fedora 或与Redhat/Fedora类似的系统，您可以把配置网络IP地址、广播地址、掩码地址、物理地址以及激活网络接口同时放在一个句子中，写入/etc/rc.d/rc.local中。比如下面的例子；   
ifconfig eth1:0 192.168.1.250 hw ether 00:11:00:33:11:44 netmask 255.255.255.0 broadcast 192.168.1.255 up   
ifconfig eth1:1 192.168.1.249 hw ether 00:11:00:33:11:55 netmask 255.255.255.0 broadcast 192.168.1.255 up   
  
解说：上面是为eth1的网络接口，设置了两个虚拟接口；每个接口都有自己的物理地址、IP地址... ...   
  
  
4.如何用ifconfig 来激活和终止网络接口的连接    
  
激活和终止网络接口的用 ifconfig 命令，后面接网络接口，然后加上 down或up参数，就可以禁止或激活相应的网络接口了。当然也可以用专用工具ifup和ifdown 工具；   
[root@linuxchao ~]#ifconfig eth0 down   
[root@linuxchao ~]#ifconfig eth0 up   
[root@linuxchao ~]#ifup eth0   
[root@linuxchao ~]#ifdown eth0   
  
对于激活其它类型的网络接口也是如此，比如 ppp0，wlan0等；不过只是对指定IP的网卡有效。   
  
注意：对DHCP自动分配的IP，还得由各个发行版自带的网络工具来激活；当然得安装dhcp客户端；这个您我们应该明白；   
  
比如Redhat/Fedora   
[root@linuxchao ~]#/etc/init.d/network start   
  
Slackware 发行版；   
[root@linuxchao ~]#/etc/rc.d/rc.inet1\

## 1.7、即时生效与永久生效

A、修改ip地址  
即时生效:  
# ifconfig eth0 192.168.0.20 netmask 255.255.255.0  
启动生效:  
修改/etc/sysconfig/network-scripts/ifcfg-eth0  
  
  
B、修改default gateway  
即时生效:  
# route add default gw 192.168.0.254  
启动生效:  
修改/etc/sysconfig/network-scripts/ifcfg-eth0  
  
  
C、修改dns  
修改/etc/resolv.conf  
修改后可即时生效，启动同样有效  
  
  
D、修改host name  
即时生效:  
# hostname fc2  
启动生效:  
修改/etc/sysconfig/network

# 二、CentOS系统知识

## 2.1、CentOS常用命令

作为linux的一个重要分支，CentOS支持绝大多数linux标准命令，其实这里作为通用教程，我们大多介绍linux的标准命令，如果是CentOS独有命令，我会特殊说明。当然，我们只会介绍常用的命令，也会包括一些特殊用法，但，这些一定是经常用到的东西。

**man ls | col -b > man.txt**

这个col命令可以把帮助内容重定向到一个文本文件，然后你可以查看文本就可以了，这样更方便阅读。

如果选择用命令行模式登录Linux的话，那么看到的第一个Linux命令就是login：

安装和登录命令：

login、shutdown、halt、reboot、install、mount、umount、chsh、exit、last；

文件处理命令：

file、mkdir、grep、dd、find、cp、mv、ls、diff、cat、ln；

系统管理相关命令：

df、top、free、quota、at、lp、adduser、groupadd、kill、crontab；

网络操作命令：

ifconfig、ip、ping、netstat、telnet、ftp、route、rlogin、rcp、finger、mail、 nslookup；

系统安全相关命令：

passwd、su、umask、chgrp、chmod、chown、chattr、sudo ps、who；

其它命令：

tar、unzip、gunzip、unarj、mtools、man、unendcode、uudecode。

### 2.1.0 文件处理命令

这些类unix系统、linux系统，之所以易于使用，功能强大，那是因为它们有为数众多的命令和参数，实现很多匪夷所思的功能，这也是这些系统令人着迷的重要原因，命令行永远都比图形界面高效，但前提是你必须学会使用它。这里给大家介绍系统中重要且常用的命令和参数，不是完整的介绍命令的手册，学会这些，你就可以非常高效的工作在命令行下，而不是一部百科全书，并且大多数的参数，也许对你并没有什么卵用，如果要了解这个命令支持所有的参数，你可以通过man查询它的帮助文档。这里不在解释。

#### 2.1.0.1、ls

这个命令是系统用来显示文件目录信息的，通常来说是所有系统中使用频率最高的命令，通常一些文档中对它的的介绍并不好，只是简单的罗列了它的一些参数，而通常这些参数并不能完全满足我们的需求，比如我只想看文件或者目录信息，你会发现，它并没有直接提供这样的参数，我们这里要说的正是这些普通教程里没有的，但对于我们非常重要和有用的用法。

语法格式：

#ls ［参数］ [目录或文件名]

参数：

-a 显示所有文件，包含隐藏文件；包括“.“和“..“

-A 显示所有文件，包含隐藏文件，不包括“.“和“..“目录；

-B 不列出文件名以~结尾的文件；

-C 按列列出文件；

-d 仅可以查看目录的属性参数及信息；

-F 这个参数加上后，能给目录添加上/，给可执行文件加上\*，给符号链接加上@ 等等，来标记文件名，目的是让用户方便查看； 并且严格按照默认排序显示。

-f 忽略大小写，显示所有文件，排序上没有大小写了。与参数a的区别是，参数a是按照默认排序的，就是数字、A~Z、a~z这种顺序显示的。

-l 长格式输出

-h 习惯人们可阅读的方式来显示文件或目录的大小，比如K、M或G等；

-p 在目录的后面附加/标记；

-r 逆序列出文件；

-S 文件的大小排序

-t 按文件最后访问（或修改）的时间排序，最新的排在最前面；

-X 根据扩展名进行排序

-U 按原始排序，

-R 递归显示，也就是说，当我们列某个目录的下的所有文件和目录时，能把它的下级目录的内容显示出来。

--help 获得帮助；

在这里，位于[]之内的都是可选的。如果您只输入ls ，就会列出用户所处当前目录的所有文件，用短格式输出，但不会列出隐藏文件。这些参数使我们经常用到的，通常已经够用了，如果觉得不够，自己查询帮助。unix系统参数通常可以连用，就是这样:

ls -laih

我们这里并不是给大家泛泛的介绍参数，如果那样，就不需要整理文档了，既然是留作学习的日志，那么通常我们要介绍的更为实用的一些实际用法，当然它的一般用法还是有必要先说一下的，然后介绍一些man帮助中没有，但我们实际应用中还要用到的用法，因为无论它有多么多的参数，实际上，你能用到的东西，才是我们真正要掌握的，好了，言归正传，下面看一下ls的一些更为实用的用法：

##### 1、只显示文件：

通常这样的实例，在教科书中是没有的，而它却是我们常用到的，有时候一个目录下文件和目录比较多，我们通常不想看到多余的目录，而ls并没有提供只显示文件或者目录的参数，那我们如何做到呢？

ls -la |grep -v "^d"

ls -la |grep "^-"

这是通过管道符来过滤掉多余的信息，前面的是过滤掉首字母是d的行，也就是目录，后边的用法是只显示首字母是"-"的行，也就是文件，这两种方法其实表达的意思都是一样的，这样的用法也把你对命令行操作带到一个新高度，就是根据你自己的需要，随便组合，来实现你的要求，这也是命令行强大的地方。

ls -F |grep -v /$

看到了吧，这就是命令行的牛X之处，你可以随意发挥，只要你认为它可以正常工作就可以了，没有什么固定的模式，

如果要多列显示过滤后的文件，目前还没有找到方法。

通过grep、awk这样的管道命令，目前只能做到单列输出。

##### 2、只显示目录

这个其实和对文件的操作有相似之处，我们具体看一下：

ls -d \*/

这是最简单也是最有效的多列只显示目录的方法

ls -la |grep -v "^-"

ls -la |grep "^d"

ls -F |grep /$

这三条命令都是单列显示目录的方法，区别只是头2条是包含了详细信息的，第三条则只是显示目录名称。

这里边我们给出的方法都是没有瑕疵的，网上很多有很多方法，其实是有瑕疵的，或者说是错误的，比如：

我要只单列显示目录名字，很多人给出的答案：

ls -la |grep -v "^-" |awk '{print $9}'

这实际上是一条错误的命令，awk其实是不适宜用在这个地方的，因为目录名称中有很多情况，如果只是用awk单纯的去打印第几列，显然会有问题，比如文件名称中含有空格，或者很有多个空格，你根本就无法区分列数，尽管awk可以添加条件语句，但这个条件并不好判断，而且情况很多，



看到了吧，根本就没有正确显示目录名称，因为它把目录名称中的空格符当成分隔符了。



而上图中的命令才是显示目录的完美方法，

如果你要显示目录的详细信息，甚至排序，加上相关参数就可以了，比如：

ls -ld \*/ //最普通的详细显示目录信息

ls -laFhSr |grep /$ //全部的可读懂的按大小倒序排列显示目录



从上面的用法中你可以看到，ls对于目录的操作是完美的，无论是多列还是单列，无论是排序，还是详细信息，都可以完美显示。但对于文件操作，稍有遗憾，就是我们没有找到对于文件名的多列显示方法。

#### 2.1.0.2、touch

如果你是使用的mac的系统的话，那么创建一个文件将是你遇到的第一个，让你头痛的问题，因为系统中没有提供让你创建文件的方法。

那么，大名鼎鼎的mac，连文件都不让用户创建吗，当然不是。

其实创建文件，一直是unix系统的一个习惯，就是unix系统中也没有这样的专门命令。

那么我们又如何完成这个任务呢，其实创建一个文件非常简单，就是用我们这节给大家讲的touch命令。

touch并不是一个标准的创建文件的命令，它的中文意思是触摸，其实他是一个对文件时间进行监控的命令，只是它有一个副产品，就是当这个文件不存在时，它就会创建它。

touch命令用来修改文件的访问时间、修改时间。如果没有指定时间，则将文件时间属性改为当前时间。当指定文件不存在，则touch命令变为创建该文件。

#### 2.1.0.3、mkdir rmdir

创建和删除目录的专用命令

mkdir -p 可以创建多层目录

rmdir其实是一个没有什么用的命令，通常删除目录我们不用这个命令，因为他只能删除空目录。

#### 2.0.1.4、rm

删除命令，既可以删除目录，也可以删除文件，慎用，这是一个非常危险的命令。

-f 不提示删除

-i 提示用户是否删除

-r 递归删除，包括目录文件，

我们通常用的就是rm -rf 目标

或者 rm -f 目标

用这个命令时一定要注意路径，千万别删除错了，因为它不会把它放到回收站，而是直接干掉了，你没有后悔药可吃，不能撤销。

#### 2.0.1.5、cp mv

#### 2.0.1.6、tar

作为linux下最重要的解压缩命令，这里有必要做一些说明。

tar [-cxtzjvfpPN] 文件与目录 ....

参数：

-c ：建立一个压缩文件的参数指令(create 的意思)；

-x ：解开一个压缩文件的参数指令！

-t ：查看 tarfile 里面的文件！

特别注意，在参数的下达中， c/x/t 仅能存在一个！不可同时存在！

因为不可能同时压缩与解压缩。

-z ：是否同时具有 gzip 的属性？亦即是否需要用 gzip 压缩？

-j ：是否同时具有 bzip2 的属性？亦即是否需要用 bzip2 压缩？

-v ：压缩的过程中显示文件！这个常用，但不建议用在背景执行过程！

-f ：使用档名，请留意，在 f 之后要立即接档名喔！不要再加参数！

　　　例如使用『 tar -zcvfP tfile sfile』就是错误的写法，要写成

　　　『 tar -zcvPf tfile sfile』才对喔！

-p ：使用原文件的原来属性（属性不会依据使用者而变）

-P ：可以使用绝对路径来压缩！

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

--exclude FILE：在压缩的过程中，不要将 FILE 打包！

非常有意思的是，tar是一个折中的压缩命令，无论是压缩效率和压缩时间都不是最高效的，但却是综合性能最好的，所以tar.gz是目前linux下最流行的的压缩格式。

另外还有一点，需要大家注意，tar的这些命令参数，有一些是有顺序要求的，有一些事需要配合使用的，下面就一些常用方法给大家介绍一下：

打包压缩显示指定名称，注意f，后边必须指定文档名称，

tar –zcvf /tmp/dbback/etc\_$date.tar.gz /etc /dbback/db.txt

查询压缩包内文件，可以指定具体查询目标，

知道了这个以后，我们就可以单独解压一个文件或者目录

tar –ztvf /tmp/dbback/etc\_$date.tar.gz db.txt

打包解压显示指定名称，指定压缩包内文件db.txt，解压到指定目录/tmp/下

tar –zxvf /tmp/dbback/etc\_$date.tar.gz db.txt -C /tmp/

这里还有两个参数是你必须了解的，一个t是查看压缩包内的文件内容，另一个p是保留文件的属性，这样我们在备份了系统的目录后，恢复后既可以正常，否则，这个目录就无法正常使用，

这个p参数非常重要，保留原有文件权限属性，任何用户解压后，任然保留，

当然前提是压缩时保留了这个参数，解压时才会起作用。

tar –zcvpf /tmp/dbback/etc\_$date.tar.gz /etc

范例：

范例一：将整个 /etc 目录下的文件全部打包成为 /tmp/etc.tar

[root@linux ~]# tar -cvf /tmp/etc.tar /etc<==仅打包，不压缩！

[root@linux ~]# tar -zcvf /tmp/etc.tar.gz /etc <==打包后，以 gzip 压缩

[root@linux ~]# tar -jcvf /tmp/etc.tar.bz2 /etc <==打包后，以 bzip2 压缩

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

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

# 如果加 j 参数，则以 .tar.bz2 来作为附档名啊～

# 上述指令在执行的时候，会显示一个警告讯息：

# 『tar: Removing leading `/" from member names』那是关於绝对路径的特殊设定。

范例二：查阅上述 /tmp/etc.tar.gz 文件内有哪些文件？

[root@linux ~]# tar -ztvf /tmp/etc.tar.gz

# 由於我们使用 gzip 压缩，所以要查阅该 tar file 内的文件时，

# 就得要加上 z 这个参数了！这很重要的！

范例三：将 /tmp/etc.tar.gz 文件解压缩在 /usr/local/src 底下

[root@linux ~]# cd /usr/local/src

[root@linux src]# tar -zxvf /tmp/etc.tar.gz

# 在预设的情况下，我们可以将压缩档在任何地方解开的！以这个范例来说，

# 我先将工作目录变换到 /usr/local/src 底下，并且解开 /tmp/etc.tar.gz ，

# 则解开的目录会在 /usr/local/src/etc 呢！另外，如果您进入 /usr/local/src/etc

# 则会发现，该目录下的文件属性与 /etc/ 可能会有所不同喔！

范例四：在 /tmp 底下，我只想要将 /tmp/etc.tar.gz 内的 etc/passwd 解开而已

[root@linux ~]# cd /tmp

[root@linux tmp]# tar -zxvf /tmp/etc.tar.gz etc/passwd

# 我可以透过 tar -ztvf 来查阅 tarfile 内的文件名称，如果单只要一个文件，

# 就可以透过这个方式来下达！注意到！ etc.tar.gz 内的根目录 / 是被拿掉了！

范例五：将 /etc/ 内的所有文件备份下来，并且保存其权限！

[root@linux ~]# tar -zxvpf /tmp/etc.tar.gz /etc

# 这个 -p 的属性是很重要的，尤其是当您要保留原本文件的属性时！

范例六：在 /home 当中，比 2005/06/01 新的文件才备份

[root@linux ~]# tar -N "2005/06/01" -zcvf home.tar.gz /home

范例七：我要备份 /home, /etc ，但不要 /home/dmtsai

[root@linux ~]# tar --exclude /home/dmtsai -zcvf myfile.tar.gz /home/\* /etc

范例八：将 /etc/ 打包后直接解开在 /tmp 底下，而不产生文件！

[root@linux ~]# cd /tmp

[root@linux tmp]# tar -cvf - /etc | tar -xvf -

# 这个动作有点像是 cp -r /etc /tmp 啦～依旧是有其有用途的！

# 要注意的地方在於输出档变成 - 而输入档也变成 - ，又有一个 | 存在～

# 这分别代表 standard output, standard input 与管线命令啦！

### 2.1.1 进程管理命令

#### 2.1.1.1、ps

Linux中的ps命令是Process Status的缩写。ps命令用来列出系统中当前运行的那些进程。ps命令列出的是当前那些进程的快照，就是执行ps命令的那个时刻的那些进程，如果想要动态的显示进程信息，就可以使用top命令。

要对进程进行监测和控制，首先必须要了解当前进程的情况，也就是需要查看当前进程，而 ps 命令就是最基本同时也是非常强大的进程查看命令。使用该命令可以确定有哪些进程正在运行和运行的状态、进程是否结束、进程有没有僵死、哪些进程占用了过多的资源等等。总之大部分信息都是可以通过执行该命令得到的。

ps 为我们提供了进程的一次性的查看，它所提供的查看结果并不动态连续的；如果想对进程时间监控，应该用 top 工具。

kill 命令用于杀死进程。

ps这个命令是一个unix的命令，因为从它的参数就可以看出，ps命令通常有四种最常用的用法，分别是：

ps aux

ps -aux

ps -le

ps -ef

这四种用法其实都是显示系统某一时刻进程信息的，注意他不是连续的，只是显示的某一时刻，ps的也有很多参数，只是其它用法不是很常用，这里说的四个参数是常用的，那么他们有什么区别吗，当然有。

##### 1. ps aux和ps -aux

ps aux和ps -aux让很多同学费解，这两个不是一样的吗，呵呵，非常不好意思的告诉你，他们的含义有很大区别。关于ps这个命令，我们man ps后发现：在ps帮助的首段有这样一段话：

请注意"ps -aux"不同于"ps aux"。POSIX和UNIX的标准要求"ps -aux"打印用户名为"x"的用户的所有进程，以及打印所有将由-a选项选择的过程。如果用户名为"x"不存在，ps的将会解释为"ps aux"，而且会打印一个警告。这种行为是为了帮助转换旧脚本和习惯。它是脆弱的，即将更改，因此不应依赖。

如果你运行ps -aux >/dev/null，那么你就会得到下面这行警告信息

Warning: bad ps syntax, perhaps a bogus '-'? See http://procps.sf.net/faq.html

**综上： 使用时两者之间直接选择ps aux**

##### 2. ps aux 和ps -ef

两者的输出结果差别不大，但展示风格不同。aux是BSD风格，-ef是System V风格。这是次要的区别，一个影响使用的区别是aux会截断command列，而-ef不会。当结合grep时这种区别会影响到结果。

**综上： 使用时两者之间直接选择ps -ef**

##### 3. ps -le和ps -ef

-le和-ef的主要区别还是关于command列的显示上，-le只是显示进程名称，没有路径，而-ef则不同，他会显示完整路径，并且不会像aux那样截取，而是完整显示，一行显示不下，他会自动换行显示。

[root@localhost /]# ps aux

USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND

root 1 0.0 0.1 2896 900 ? Ss 08:42 0:01 /sbin/init

root 2 0.0 0.0 0 0 ? S 08:42 0:00 [kthreadd]

下面就aux的显示结果解释一下各参数的意义：

USER 进程的拥有者

PID 进程的id

PPID 父进程id

%CPU 进程占用的CPU百分比

%MEM 进程占用内存的百分比

NI 进程的NICE值，数值大，表示较少占用CPU时间

VSZ 进程使用的虚拟內存量（KB）

RSS 进程占用的固定內存量（KB）（驻留中页的数量）

TTY 进程在哪个终端上运行（登陆者得终端位置）

若与终端无关，则显示（？）

若为pts/0等，则表示由网络连接主机进程

WCHAN 当前进程是否正在进行，若为-表示正在进行

START 该进程被触发启动时间

TIME 该进程实际使用cup运行的时间

COMMAND 命令的名称和参数

linux上进程有5种状态:

1. 运行(正在运行或在运行队列中等待)

2. 中断(休眠中, 受阻, 在等待某个条件的形成或接受到信号)

3. 不可中断(收到信号不唤醒和不可运行, 进程必须等待直到有中断发生)

4. 僵死(进程已终止, 但进程描述符存在, 直到父进程调用wait4()系统调用后释放)

5. 停止(进程收到SIGSTOP, SIGSTP, SIGTIN, SIGTOU信号后停止运行运行)

ps工具标识进程的5种状态码:

D 不可中断 uninterruptible sleep (usually IO)

R 运行 runnable (on run queue)

S 中断 sleeping

T 停止 traced or stopped

Z 僵死 a defunct (”zombie”) process

##### 4.ps自定义显示

上边的都是ps的标准用法，当然你也可以自定义使用：

ps -eo stat,euid,ruid,tty,tpgid,sess,pgrp,ppid,pid,pcpu:10,comm

注意一下这个命令的参数用的是-eo，其中e表示全部，o表示指定格式，

而后边的指定格式中，有一些需要说明一下，比如这个pcpu显示出来的是%CPU，就是说你不能用%做参数，而要用用p来代替%，那内存的显示方法显然应该是pmem了，:10,表示字段间距，comm表示只显示进程名称，command表示显示进程完整路径名称。

[root@localhost /]# ps -eo user,stat,ppid,pid,pcpu:10,command,pmem

USER STAT PPID PID %CPU COMMAND %MEM

root Ss 0 1 0.0 /sbin/init 0.1

root S 0 2 0.0 [kthreadd] 0.0

root S 2 3 0.0 [migration/0] 0.0

### 2.1.2 vmstat

vmstat是一个监控系统资源的命令，他比top命令更简洁，占用系统资源更少，他的格式如下：vmstat 刷新时间 刷新次数

vmstat 1 10表示的的意思就是监控结果每1秒刷新一次，刷新10次，

### 2.1.3 dmseg 查询开机内核检测信息

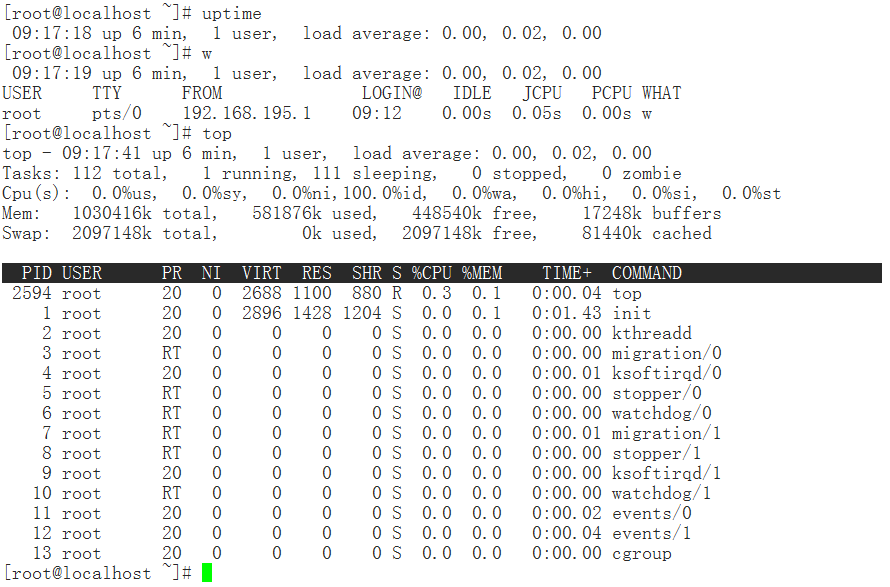
dmseg显示的每次开机时，内核检测到的硬件信息，他并不是实时的。

dmseg |grep CPU

这样就可以显示和CPU相关的硬件信息了，当然你还可以通过/proc/cpuinfo来查询cpu信息。也就是说/proc这个目录中存放着系统硬件信息文件，但这些文件都不是永久的，因为他是内存的挂载点，系统一断电，这些文件也就不存在了。

### 2.1.4 uptime w top

这三个命令的第一行都可以显示系统当前时间和平均负载。



当然w和top命令更重要的功能不是这个，w是显示当前登陆终端的，top是用来查询系统健康状态的。

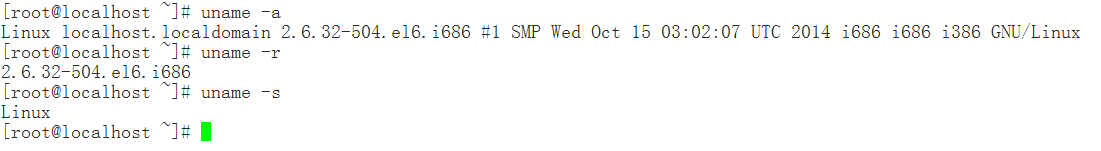
### 2.1.5 uname 查看内核信息

uname查看系统内核信息的：

-a 表示查看完整内核信息

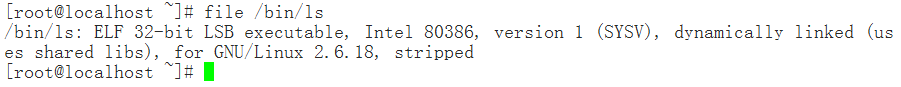
-r 表示查看内核版本

-s 表示查看内核名称



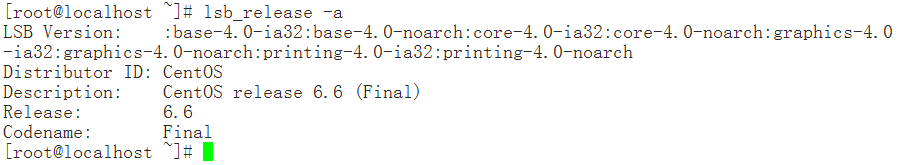
### 2.1.6 file 查看系统位数

linux并没有查看系统位数的命令，但这里我们弄了一个曲线方式，因为file命令可以查看文件的信息，间接的达到了这个目的。注意不是查看任意文件，都可以，而是要查看系统外部命令文件，才可以的，哦。



很神奇吧，这样我们就知道，当前系统是32位系统了。

### 2.1.7 lsb\_release 查看系统当前发行版本



### 2.1.8 lsof 列出进程打开或者使用的文件信息

-c 字符串：只列出以字符串开头的进程打开的文件

-u 用户名：只列出某个用户的进程打开的文件

-p pid ：只列出某个pid进程打开的文件

### 2.1.9 系统定时任务

做一个系统运维工程师，显然你必须学会编写自动化脚本，来让系统根据脚本自动执行某些任务，比如备份，重启等一些工作，因为这些工作，

crond是定时任务的守护进程

crontab 是具体设置定时任务的命令，通常有如下3个参数比较重要

-l：用来查询当前定时任务情况

-e：用来设置具体的定时任务

-r：用来删除定时任务，注意，这个参数是删除所有定时任务，

如果想单条删除，你只能用-e参数，进入编辑模式，然后删除

### 2.1.10日志

## 2.2、用户配置文件管理

嘿嘿！帐号管理最重要的两个档案就是/etc/passwd 与 /etc/shadow了！这两个档案可以说是 Linux 里头最重要的档案之一了！ 如果没有这两个档案的话，呵呵！您可是无法登入 Linux 的呦！所以，底下我们先针对这两个档案来进行说明。 当然啰，更详细的资料您可以自行 man 5 passwd 及 man 5 shadow 的啦～

### 2.2.1 用户信息文件/etc/passwd

Linux所有的用户信息都是记录在passwd文件下，包括root，系统伪用户，普通用户。每一行都代表一个帐号， 有几行就代表有几个帐号在你的系统中！不过需要特别留意的是，里头很多系统伪账号，例如 bin, daemon, adm, nobody 等等，这些帐号是系统正常运作时所需要的，请不要随意的杀掉他呢！每一行的格式如下：

name:password:uid:gid:comment:home:shell

我们先来看一下每个 Linux 系统都会有的第一行，就是 root 这个系统管理员那一行好了，你可以明显的看出来，每一行使用6个“:”分隔开，共有七个咚咚，分别是：



1. 帐号名称：就是帐号名称啦！对应 UID 用的！例如 root 就是预设的系统管理员的帐号名称；

2. 密码：早期的 Unix 系统的密码是放在这个文件中的，但是因为这个文件的特性是所有的程序都能够读取，所以，这样一来很容易造成资料的被窃取，因此后来就将这个字段的密码资料给他改放到 /etc/shadow 中了，关于 /etc/shadow 这一部份等一下再说。而这里你会看到一个 x ，呵呵！别担心，这表示密码已经被移动到 shadow 这个加密过后的档案啰，这个字段不能为空，linux不允许空密码登陆系统。

3. UID：这个就是使用者识别码 (ID) 啰！通常 Linux 对于 UID 有几个限制需要说给您瞭解一下：

0：当UID是0时，代表这个帐号是‘系统管理员’！所以当你要作另一个系统管理员帐号时，你可以将该帐号的UID改成0即可；这也就是说，一部系统上面的系统管理员不见得只有root喔！不过，不很建议有多个帐号的UID是 0 啦～

1~499：保留给系统使用的ID，其实1~65534之间的帐号并没有不同， 也就是除了0之外，其他的UID并没有不一样，预设 500 以下给系统作为保留帐号只是一个习惯。这样的好处是，以有名的DNS 伺服器的启动服务‘named’为例，这个程式的预设所有人named的帐号UID是25，当有其他的帐号同样是25时，很可能会造成系统的一些小问题！为了杜绝这样的问题，建议保留500以前的UID给系统吧！

不过，一般来说， 1~99 会保留给系统预设的帐号，另外 100~499 则保留给一些服务来使用。

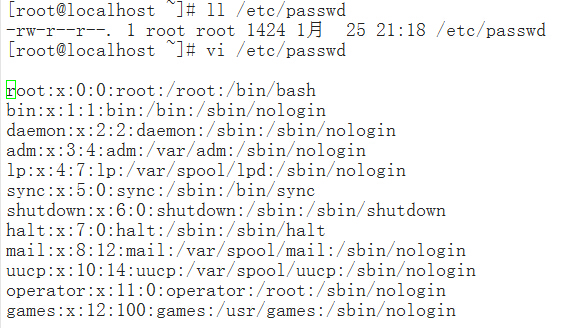
500~65535给一般使用者用的。事实上，目前的linux核心 (2.6.x 版)已经可以支援到4294967295 (2^32-1)这么大的 UID号码喔！

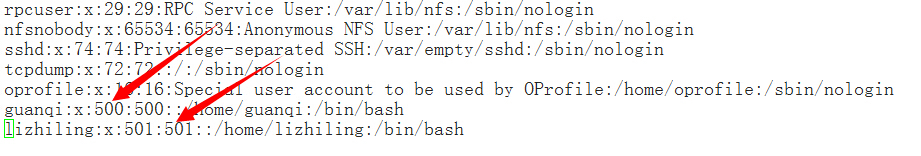
4. GID：这个与/etc/group有关！其实/etc/group的观念与/etc/passwd差不多，只是他是用来规范group的而已！

5. 使用者资讯说明栏：这个栏位基本上并没有什么重要用途， 只是用来解释这个帐号的意义而已！不过，如果您提供使用finger的功能时，这个栏位可以提供很多的讯息呢！底下的chfn可以解释一下啰！

6. 家目录：这是使用者的家目录，以上面为例， root 的家目录在 /root ，所以当 root 登入之后，就会立刻跑到 /root 里头啦！呵呵！ 如果你有个帐号的使用空间特别的大，你想要将该帐号的家目录移动到其他的硬碟去，没有错！可以在这里进行修改呦！预设的使用者家目录在/home/yourIDname

7. Shell：所谓的 shell 是用来沟通人类下达的指令与硬体之间真正动作的界面！我们通常使用/bin/bash这个shell来进行指令的下达！嘿嘿！发现了吧？我们在bash章节里面提到很多次，登入 Linux时为何预设是bash呢？就是这里设定的啦～ 这里比较需要注意的是，有一个shell可以用来替代成让帐号无法登入的指令！那就是/sbin/nologin 这个东西！这也可以用来制作纯pop邮件帐号者的资料呢！





总结一下：例如：abc:x:501:501::/home/abc:/bin/bash

（1）：用户名。abc

（2）：密码（已经加密迁移）x

（3）：UID（用户标识）,操作系统自己用的 501

（4）：GID组标识。501

（5）：用户全名或本地帐号（使用者资讯说明栏）通常为空

（6）：开始目录 /home/abc

（7）：登录使用的Shell，命令解释器。/bin/bash

### 2.2.2 影子文件/etc/shadow

上面约略提到，由于每个程序都需要取得 uid 与 gid 来判断权限的问题，所以，/etc/passwd的权限必须要设定成为 -rw-r--r-- 这样的权限，在这样的情况下，使用者的密码不就任何人都可以看到吗？即使这个文件内的密码栏是加密的，坏心肠的朋友也可能利用暴力破解法去try and error找出您的密码资料......

因为这样的关系，所以后来发展出将密码移动到/etc/shadow这个文件分隔开来的技术，而且还加入很多的密码限制参数在 /etc/shadow里头呢！我们先来瞭解一下这个文件的构造吧！

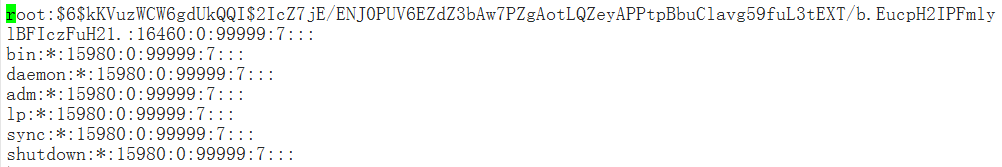
基本上， shadow和passwd的格式一样，如果数一数，会发现共有九个字段， 这九个栏位的用途是这样的：

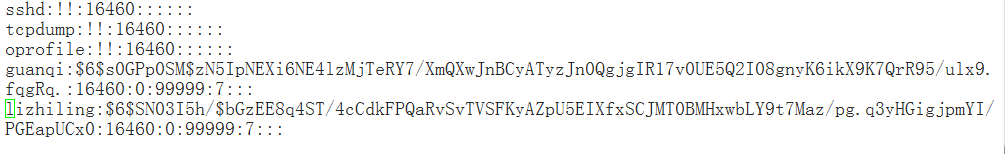
首先查询一下这个文件的权限



我们看到了这个文件的神奇之处就是权限全是零，就是说系统中除了root用户外，其他任何用户连读的权限都没有的。

打开这个文件显示如下





1. 帐号名称：由于密码也需要与帐号对应啊～因此，这个文件的第一字段就是帐号，必须要与/etc/passwd相同才行！

2. 密码：这个才是真正的密码，而且是经过编码过的密码啦！ 你只会看到有一些特殊符号的字母就是了！需要特别留意的是，虽然这些加密过的密码很难被解出来，但是‘很难’不等于‘不会’，所以， 这个文件的预设属性是‘----------’，亦即只有 root 才可以读写就是了！你得随时注意，不要不小心更动了这个文件的属性呢！

另外，如果是在密码栏的第一个字元为‘\*’或者是‘！！’，表示这个帐号并不会被用来登入的意思。所以万一哪一天你的某个使用者不乖时，可以先在这个档案中，将他的密码栏位的最前面多加一个 \*或者！！嘿嘿！他就无法使用该帐号啰！

　　3. 最近更动密码的日期：这个字段记录了‘更动密码的那一天’的日期， 不过，很奇怪呀！在我的例子中怎么会是16460呢？呵呵，这个是因为计算Linux日期的时间是以 1970 年 1 月 1 日作为 1 ，而 1971 年 1 月 1 日则为 366 啦！ 所以这个日期是累加的呢！得注意一下这个资料呦！那么最近的 2015年1月25日就是 16460 啦，瞭解了吗？



　　4. 密码不可被更动的天数： 第四个字段记录了这个帐号的密码需要经过几天才可以被变更！如果是 0 的话，表示密码随时可以更动的意思。这的限制是为了怕密码被某些人一改再改而设计的！如果设定为 20天的话，那么当你设定了密码之后， 20 天之内都无法改变这个密码呦！

　　5. 密码需要重新变更的天数（有效期）： 由于害怕密码被某些‘有心人士’窃取而危害到整个系统的安全，所以有了这个字段的设计。你必须要在这个时间之内重新设定你的密码，否则这个帐号将会暂时失效。 而如果像上面的99999的话，那就表示，呵呵，密码不需要重新输入啦！ 不过，如果是为了安全性，最好可以设定一段时间之后，严格要求使用者变更密码呢！

　　6. 密码需要变更期限前的警告期限：当帐号的密码失效期限快要到的时候，就是上面那个‘必须变更密码’的那个时间时， 系统会依据这个字段的设定，发出‘警告’言论给这个帐号，提醒他‘再过 n 天你的密码就要失效了，请尽快重新设定你的密码呦！’，如上面的例子，则是密码到期之前的 7 天之内，系统会警告该用户。

　　7. 密码过期的恕限时间：如果用户过了警告期限没有重新输入密码，使得密码失效了，也就是说，你在‘必须变更密码的期限前，并没有变更你的密码！’ 那么该组密码就称为‘失效的密码’啰～怎么办？没关系，还有这个栏位的天数设计啊～ 意思就是说，当密码失效后，你还可以用这个密码在 n 天内进行登入的意思。 而如果在这个天数后还是没有变更密码，呵呵！那么您的帐号就失效了！无法登入！

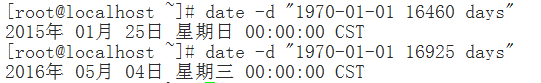
　　8. 帐号失效日期：这个日期跟第三个字段一样，都是使用 1970 年以来的总日数设定。这个字段表示：这个帐号在此字段规定的日期之后，将无法再使用。 这个字段会被使用通常应该是在‘收费服务’的系统中， 你可以规定一个日期让该帐号不能再使用啦！

　　9. 保留：最后一个字段是保留的，看以后有没有新功能加入。

下面举个例子来复习一下，假如我的liming这个使用者的密码栏如下所示：

liming:$1$8zdAKdfC$XDa8eSus2I7nQL7UjRsIy/:16460:3:60:7:2:16925:

那么首先就看一下这两个时间戳的具体时间是多少



　　1,2 那么前两个字段就不用解释了，用户名和加密的密码。

3\* 最近一次更动密码的日期是 2015/01/25 (16460)；

4\* 能够修改密码的时间是3天以后，也就是 2015/01/28 以前 liming不能修改自己的密码； 如果使用者还是尝试要更动自己的密码，系统就会出现这样的讯息：

　　You must wait longer to change your password

passwd: Authentication token manipulation error

　　5\* 密码有效期为60天，使用者必须要在 2015/01/25 到 2015/03/26 之间的 60 天限制内去修改自己的密码，若 2015/03/26 之后还是没有变更密码时，该帐号就会宣告失效；

6\* 如果使用者一直没有更改密码，那么在 2015/03/26 之前的 7天内，系统会警告liming应该修改密码的相关资讯；例如当liming 登入时，系统会主动提示如下的资讯：

　　Warning: your password will expire in 5 days

　　7\* 如果该帐号一直到 2015/03/26 都没有更改密码，由于还有两天的恕限时间，因此，liming还是可以在 2015/03/28 以前继续登入；

　　\* 如果使用者在 2015/03/28（包括了2天的宽恕期） 以前变更过密码，那么那个 16460 的日期就会跟著改变，因此，所有的限制日期也会跟著相对变动喔！^\_^

　　8\* 无论李明如何动作，到了 16925 ，大约是 2016/05/04 左右，该帐号就失效了～

　　常常听到：我的密码忘记或者被更动了？怎么办？

　　有的时候会发生这样的情况，就是说，你的root密码忘记了！要怎么办？重新安装吗？另外，有的时候是被入侵了， root的密码被更动过，该如何是好？

　　这个时候就必须要使用到/etc/shadow这个文件了！我们刚刚知道密码是存在这个文件中的，所以只要你能够以各种可行的方法开机进入Linux，例如单人维护模式，或者是以live CD (KNOPPIX) 来进入 Linux 系统。之后，将硬碟顺利挂载，然后进入/etc/shadow这个文件中，将root的密码这一栏全部清空！然后再登入Linux一次，这个时候root将不需要密码 (有的时候需要输入空格)就可以登入了！这个时候请赶快以 passwd 设定 root 密码即可。

### 2.2.3 组用户文件/etc/group和组密码文件/etc/gshadow

在/etc/group组文件中,记录了系统中所有的组，组文件到结构，比用户文件要简单一些，/etc/group的格式如下：

group\_name:passwd:GID:user\_list

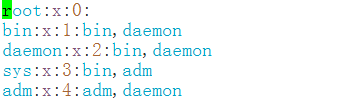
组 名:口 令:组ID:组内用户列表

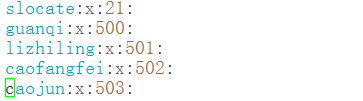
1）“组名”是用户组的名称，由字母或数字构成。与/etc/passwd中的登录名一样，组名不应重复。

2）“口令”字段存放的是用户组加密后的口令字。一般Linux系统的用户组都没有口令，即这个字段一般为空，或者是x。

3）“组ID” ：GID

4）“组内用户列表”：每个用户之间用,号分割；本字段可以为空；如果字段为空表示用户组为GID的用户名；





/etc/gshadow是/etc/group的加密资讯文件，比如用户组（Group）管理密码就是存放在这个文件。

/etc/gshadow和/etc/group是互补的两个文件；对于大型服务器，针对很多用户和组，定制一些关系结构比较复杂的权限模型，设置用户组密码是极有必要的。

比如我们不想让一些非用户组成员永久拥有用户组的权限和特性，这时我们可以通过密码验证的方式来让某些用户临时拥有一些用户组特性，这时就要用到用户组密码。

[root@localhost ~]# cat /etc/gshadow

root:::root

bin:::root,bin,daemon

组名：口令：组管理者：组内用户列表

1) 组名: 是用户组的名称，由字母或数字构成。

2) 口令：用户组密码，这个段可以是空的或!，如果是空的或有!，表示没有密码

3) 组管理者：这个字段也可为空，如果有多个用户组管理者，用,号分割

4) 组内用户列表：如果有多个成员，用,号分割 ;

### 2.2.4 家目录

用户初始登陆目录，

普通用户：/home/用户名

所有者和所属组都是该用户，权限都是700



root用户：/root/

所有者和所属组都是root用户，权限是550



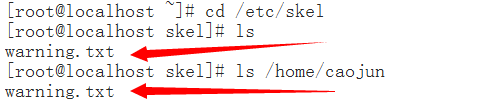
这里大家可能比较奇怪，普通用户的家目录权限是700，为什么root用户的权限只有550，显然这个目录对于root来说，给与什么样的权限都一样，可以读写和执行，也就是没有限制的，但这样设置还是比较奇怪的。

### 2.2.5 邮件/var/spool/mail/和模板目录/etc/skel

Liunx的邮件目录是自动生成的，当用useradd命令建立一个新的用户时，在/var/spool/mail目录下，会和家目录一样，生成一个邮件目录。



而模板文件，顾名思义，就是在/etc/skel这个目录下的东西，是所有新建用户的模板文件，所有新建用户的家目录都会自动按照这个模板生成。



## 2.2、时间戳

查询linux时间戳具体日期。

date –d “1970-01-01 16460 days”

把日期换算成时间戳

echo $(($(date --date="2015/01/25" +%s)/86400+1))

## 2.3、ls –l后边有一个点

执行了下面的命令，结果在权限后面还有个“.”，这个.是干什么的呢？

[root@localhost /]# ls -l /etc/shadow

----------. 1 root root 1142 9月 25 16:19 /etc/shadow

因为新版本ls把acl和selinux属性加进去了，与系统无关，新版本的ls代码使用12个字节长的char类型的字符数组modebuf来记录文件属性，前面10个和老版本一样，

第一个字节表示类型，-表示文件，D表示目录，l表示软连接

后9个表示普通权限，rwxrw-r—(764)

第11个表示selinux和acl的设置，

最后一个当然是c的结尾符\0了。

当文件或者文件夹只使用了selinux context的属性，那么这个在ls -l时后面会是一个点(.)。

只要设置了acl，就是那个setacl设置的，后面将是一个加(+)号

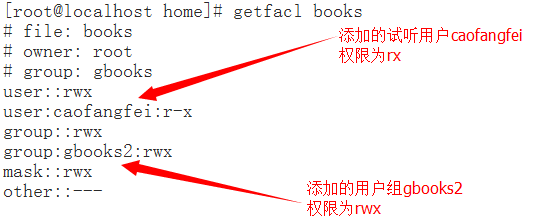
## 2.4、ACL权限

setfacl –m u:caofangfei:rx /home/books

setfacl –m g:gbooks2:rwx /home/books

cd /home

getfacl books





## 11、vim

Vim是一个类似于Vi的文本编辑器，不过在Vi的基础上增加了很多新的特性，Vim普遍被推崇为类Vi编辑器中最好的一个，事实上真正的劲敌来自Emacs的不同变体。1999年Emacs被选为Linuxworld文本编辑分类的优胜者，Vim屈居第二。但在2000年2月Vim赢得了Slashdot Beanie的最佳开放源代码文本编辑器大奖，又将Emacs推至二线， 总的来看， Vim和Emacs同样都是非常优秀的文本编辑器。

### 11.1 vim的工作模式：

1.命令模式

vi在初始启动后首先进入命令模式，这时用户可以利用一些预先定义的按键来移动光标、删除文字、复制或粘贴文字等。这些按键均是普通的字符，例如l是向右移动光标，相当于向右箭头键，k是向下移动光标，相当于向下箭头键。在编辑模式下，用户还可以利用一些特殊按键选定文字，然后再进行删除、或复制等操作。

2.插入模式

当用户在命令模式下键入 i, a, o 等命令之后，可进入插入模式。在插入模式下，用户随后输入的，除 Esc 之外的任何字符均将被看成是插入到编辑缓冲区中的字符。按Esc之后，从插入模式切换到命令模式。

3.编辑模式

在命令模式下，键入 : 可进入编辑模式。在该模式下，Vi 将把光标挪到屏幕的最下方，并在第一个字符的位置显示一个 :（冒号）。这时，用户就可以键入一些命令。这些命令可用来保存文件、读取文件内容、执行 Shell 命令、设置 Vi 参数、以正则表达式的方式查找字符串或替换字符串等。

### Vim全选模式

曾经也在找看看有没有快捷的方法全选，但是网上很多都是重复，并且错误的，比如：

1,$y,这是什么？要写也要写成:1,$y,但是我告诉你，写成:1,$y是全部复制，不是全选

dG，这个就更加误导人了，这是删除光标所在行到最后一行的内容（包括光标所在行的内容）

上面这里都扯到哪里去了。全选正确的答案是：

ggVG

稍微解释一下上面的命令

gg 让光标移到首行，在vim才有效，vi中无效

V 是进入Visual(可视）模式

G 光标移到最后一行

选中内容以后就可以其他的操作了，比如：

d 删除选中内容

y 复制选中内容到0号寄存器

"+y 复制选中内容到＋寄存器，也就是系统的剪贴板，供其他程序用

|  |  |  |
| --- | --- | --- |
| 命令模式 | 插入模式 | 编辑模式 |
| 打开vi首先进入的就是命令模式 | a,i,r,o,A,I,R,O进入编辑模式 | 命令模式下直接输入 ：  插入模式下按Esc键，再按： |
| x 删除光标所在字符  nx 删除光标所在处后n个字符  dd 删除光标当前行  ndd 删除n行  dG 删除光标所在行到文件尾字符  D 删除光标所在处到行尾字符  n1,n2d 删除指定行字符 例如1,5d | a 光标所在字符后插入  A 光标所在行尾插入 | :w 【filename】 |
| yy 复制当前行  nyy 复制n行 | i 光标所在字符前插入  I 光标所在行首插入 | :open 【filename】 |
| p,P 粘贴 | o 光标下一行插入  O 光标上一行插入 | :split 横排打开多个文件  :vsplit 纵排打开多个文件 |
| u 撤消 | r 表示替换单个字符  R 进入替换模式 | :set nu  :set nonu |
| G 跳转到最后一行  gg 跳转到第一行  nG 跳转到第n行 |  | :n 设置行号后跳转到第n行。 |
|  |  | /string 直接输入/in表示查找in字符串  按n查找下一个位置。  :set ic不区分大小写  :set noic区分大小写 |
| 0 移动到行首 |  |  |
| $ 移动到行尾 |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| [Ctrl]+z 暂停退出 |  |  |
| [Ctrl]+r 重做上一个动作 |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

vi一般用法

一般模式 编辑模式 指令模式

h 左 a,i,r,o,A,I,R,O :w 保存

j 下 进入编辑模式 :w! 强制保存

k 上 dd 删除光标当前行 :q! 不保存离开

l 右 ndd 删除n行 :wq! 保存后离开

0 移动到行首 yy 复制当前行 :e! 还原原始档

$ 移动到行尾 nyy 复制n行 :w filename 另存为

H 屏幕最上 p,P 粘贴 :set nu 设置行号

M 屏幕中央 u 撤消 :set nonu 取消行号

L 屏幕最下 [Ctrl]+r 重做上一个动作 ZZ 保存离开

G 档案最后一行 [ctrl]+z 暂停退出 :set nohlsearch 永久地关闭高亮显示

/work 向下搜索 :sp 同时打开两个文档

?work 向上搜索 [Ctrl]+w 两个文档设换

gg 移动到档案第一行 :nohlsearch 暂时关闭高亮显示

vim——打开多个文件、同时显示多个文件、在文件之间切换

打开多个文件：

1.vim还没有启动的时候：

在终端里输入

vim file1 file2 ... filen便可以打开所有想要打开的文件

2.vim已经启动

输入

:open file

可以再打开一个文件，并且此时vim里会显示出file文件的内容。

同时显示多个文件：

:split

:vsplit

在文件之间切换：

1.文件间切换

Ctrl+6—下一个文件

:bn—下一个文件

:bp—上一个文件

对于用(v)split在多个窗格中打开的文件，这种方法只会在当前窗格中切换不同的文件。

2.在窗格间切换的方法

Ctrl+w+方向键——切换到前／下／上／后一个窗格

Ctrl+w+h/j/k/l ——同上

Ctrl+ww——依次向后切换到下一个窗格中

11.1 vim的基本编辑命令

Vim是unix或者linux系统中最原始，也是最基本的文本编辑器，虽然原始，但其功能之强，效率之高，是每一位系统管理员和程序员必须掌握的工具。

### 11.2 vim的使用技巧

:r !date

直接导入命令查询结果。

:map ^P I#<ESC>

定义^P为行首添加注释#。

连续多行注释

1,4s/^/#/g 第一行到第四行 行首添加#，不询问

1,4s/#//g 第一行到第四行 行首删除#，不询问

1,4s/^/\/\//g 第一行到第四行 行首添加//，不询问

其中，^表示行首，添加/需要用转义字符\

ab 为替换字符，这个通常比有用。

ab mymail [adonis1978@163.com](mailto:adonis1978@163.com)

通常这些都是需要用用户自己来维护的，因为这些快捷方式都是根据自身习惯设置的，root和普通用户的配置文件都在其家目录下，配置文件为 .vimrc

set nu

map ^P I#<ESC>

ab mymail [adonis1978@163.com](mailto:adonis1978@163.com)

:wq

这样，用户每次登陆系统，vim就都可以执行这些快捷方式了。

## 12、单用户模式下修改Root用户的密码

http://www.cnblogs.com/xiaoluo501395377/archive/2013/05/19/3087664.html

我们知道Linux系统的启动级别一共有6种级别，通过 /etc/inittab 这个文件我们就能看到：

[root@xiaoluo ~]# cat /etc/inittab

# inittab is only used by upstart for the default runlevel.

#

# ADDING OTHER CONFIGURATION HERE WILL HAVE NO EFFECT ON YOUR SYSTEM.

#

# System initialization is started by /etc/init/rcS.conf

#

# Individual runlevels are started by /etc/init/rc.conf

#

# Ctrl-Alt-Delete is handled by /etc/init/control-alt-delete.conf

#

# Terminal gettys are handled by /etc/init/tty.conf and /etc/init/serial.conf,

# with configuration in /etc/sysconfig/init.

#

# For information on how to write upstart event handlers, or how

# 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 networking)

# 3 - Full multiuser mode

# 4 - unused

# 5 - X11

# 6 - reboot (Do NOT set initdefault to this)

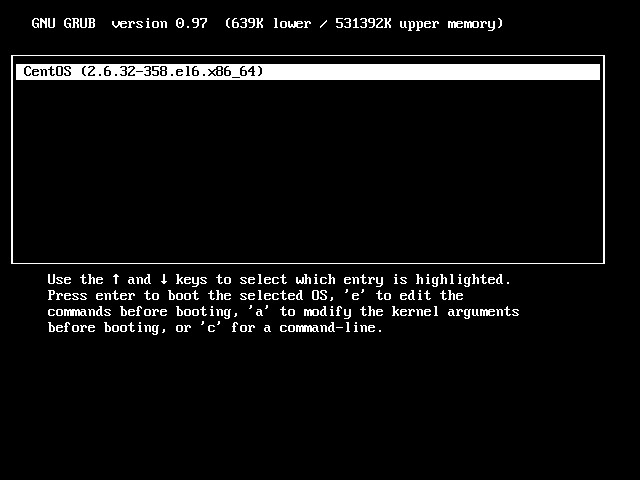
#

id:5:initdefault:

这里我们看到系统的默认启动级别是5，也就是有图形界面的那个。

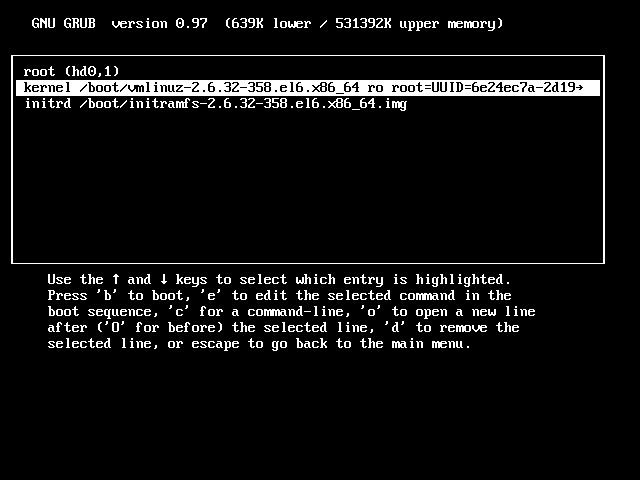
但是，在现实生活中可能会出现这种问题，我们可能忘记了一台主机的root密码，但是我们又需要通过root用户登录该系统去处理一些事情，这个时候我们怎么办呢？我们看到在Linux的启动级别中有一个单用户模式启动，也就是启动级别1，当我们如果忘记了root用户的秘密，但是又需要修改root密码的时候，这个时候我们就要通过在启动的时候给系统的内核传递一个参数1或者single来告诉内核，我需要以单用户模式登陆操作系统，这个时候我们就能能够通过passwd命令来重设root用户的密码。具体操作是怎样的呢？咱们有图有有真像!!

首先我们重启我们的系统(我这里是CentOS)，然后在界面启动时让它停留一下，随便按一下键盘上的一个键，此时就会进入到操作系统配置引导界面



 还记得上一篇随笔里面**/boot/grub/grub.conf**这个文件里的第一个 title 字段吗？ 没错，每一个title都是一个操作系统的配置选项，这里我们只有一个，也就是上面图片显示的那个，如果有多个title字段，这里就会列出来供我们选择不同的操作系统。

好了，通过下面的英文提示我们发现按下键盘上的 **e** 键就可以进入到编辑界面



 看到这个界面是不是很熟悉呢？没错，这三个选项就是我们 **/boot/grub/grub.conf** 文件里的配置信息，我们通过查看这个文件的内容来看一下：

[复制代码](javascript:void(0);)

[root@xiaoluo ~]# cat /boot/grub/grub.conf

# grub.conf generated by anaconda

#

# Note that you do not have to rerun grub after making changes to this file

# NOTICE: You do not have a /boot partition. This means that

# all kernel and initrd paths are relative to /, eg.

# root (hd0,1)

# kernel /boot/vmlinuz-version ro root=/dev/sda2

# initrd /boot/initrd-[generic-]version.img

#boot=/dev/sda

default=0

timeout=5

splashimage=(hd0,1)/boot/grub/splash.xpm.gz

hiddenmenu

title CentOS (2.6.32-358.el6.x86\_64)

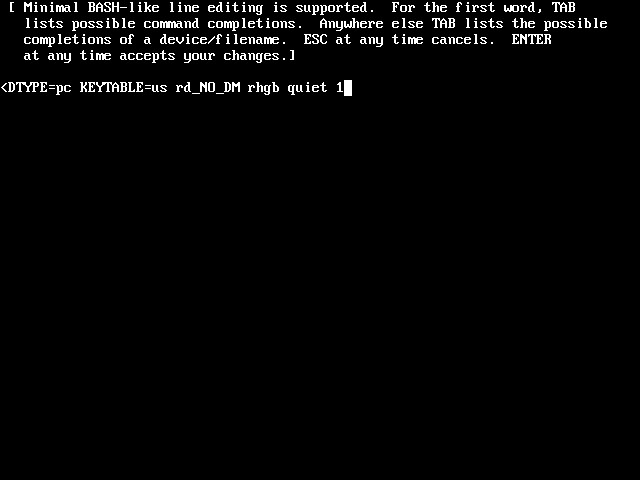
root (hd0,1)

kernel /boot/vmlinuz-2.6.32-358.el6.x86\_64 ro root=UUID=6e24ec7a-2d19-466e-bacc-92750b1f4bef rd\_NO\_LUKS rd\_NO\_LVM LANG=en\_US.UTF-8 rd\_NO\_MD SYSFONT=latarcyrheb-sun16 crashkernel=auto KEYBOARDTYPE=pc KEYTABLE=us rd\_NO\_DM rhgb quiet

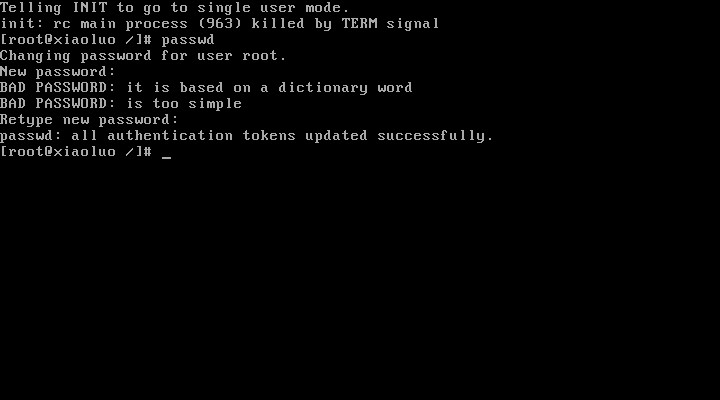
initrd /boot/initramfs-2.6.32-358.el6.x86\_64.img

[复制代码](javascript:void(0);)

这时我们需要将光标移动到第二个选项，也就是Linux内核那里，同样按下键盘上的 **e** 键进入到编辑界面，此时我们在后面追加参数 1 或者参数 single就可以告诉Linux系统的内核，我需要以单用户的模式登陆该系统。



然后我们回车，进入到上层的界面，此时按下键盘上的 **b** 键就是启动操作系统，此时我们的Linux操作系统就是以单用户的模式登陆了。我们发现系统非常快速的就进入到了命令行模式下的界面，因为单用户模式是不 会启动任何服务的，同时也不需要输入root密码，就能直接进入到root用户下，此时我们就可以通过passwd 命令来重置我们的root用户的密码



然后我们通过exit命令就可以退出单用户模式，此时操作系统内核就会根据我们之前看到的 /etc/inittab 配置文件中设置的默认启动级别来启动。

这样我们就可以通过在进入GRUB引导时在内核参数里面追加参数 1 或者 single 来进入单用户模式来修改我们的root用户密码。

但是，请注意，因为上面的操作只需要通过单用户模式就可以修改我们的root用户密码，所以说一旦别人有机会接触到我们的服务器主机，那么root用户的密码就很容易被别人修改了，这样当然是非常的不安全的，所以我们还是需要通过GRUB加密来对开启一层密码防护。

通过在 /boot/grub/grub.conf 启动配置中加入以下类似的代码来对grub进行加密：

password --md5 $1$6H92B1$PzoPV63kTMk4uEhZQTAZ//

其中后面那一串是MD5加密算法，这个我们可以通过 **grub-md5-crypt** 这个命令来生成加密后的算法

[root@xiaoluo ~]# grub-md5-crypt

Password:

Retype password:

$1$UGA2B1$DriIdrVTEgVg95fHHX4H./

[root@xiaoluo ~]#

在上面的Password、Retype password那里输入我们需要设置的密码，然后就能生成MD5算法加密后的密码，我们将这个加密后的密码添加到

/boot/grub/grub.conf 这个配置文件里即可，例如：

[复制代码](javascript:void(0);)

[root@xiaoluo ~]# vi /boot/grub/grub.conf

password --md5 $1$UGA2B1$DriIdrVTEgVg95fHHX4H./

# grub.conf generated by anaconda

#

# Note that you do not have to rerun grub after making changes to this file

# NOTICE: You do not have a /boot partition. This means that

# all kernel and initrd paths are relative to /, eg.

# root (hd0,1)

# kernel /boot/vmlinuz-version ro root=/dev/sda2

# initrd /boot/initrd-[generic-]version.img

#boot=/dev/sda

default=0

timeout=5

splashimage=(hd0,1)/boot/grub/splash.xpm.gz

hiddenmenu

title CentOS (2.6.32-358.el6.x86\_64)

root (hd0,1)

kernel /boot/vmlinuz-2.6.32-358.el6.x86\_64 ro root=UUID=6e24ec7a-2d19-466e-bacc-92750b1f4bef rd\_NO\_LUKS rd\_NO\_LVM LANG=en\_US.UTF-8 rd\_NO\_MD SYSFONT=latarcyrheb-sun16 crashkernel=auto KEYBOARDTYPE=pc KEYTABLE=us rd\_NO\_DM rhgb quiet

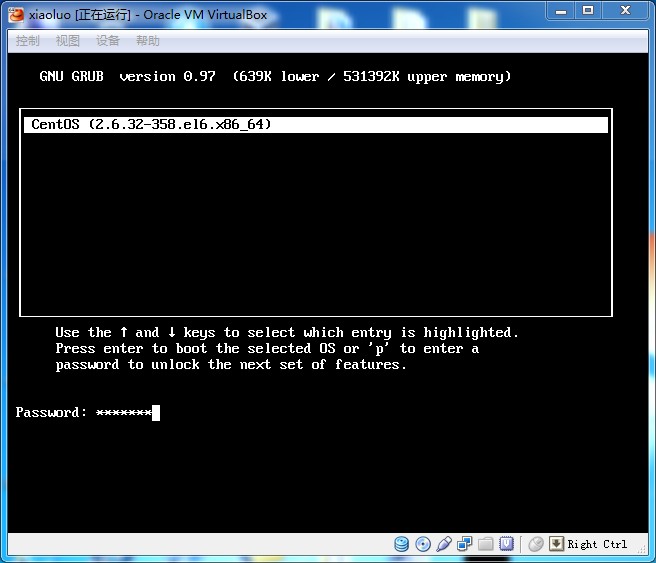
initrd /boot/initramfs-2.6.32-358.el6.x86\_64.img

[复制代码](javascript:void(0);)

这个时候我们再次重启一下系统，试着进入到GRUB里面来看看，

0

我们发现下面的英文提示已经变了，之前可以按键盘上的 **e** 键进入到GRUB里面，现在按 **e** 键已经没用了，此时提示我们按 **p** 键来输入GRUB的密码才能进入到GRUB里面



此时我们输入之前设置的GRUB密码即可，然后界面就会进入到我们熟悉的修改GRUB那里了。

所以说通过GRUB的加密算法我们可以对进入GRUB进行加密，这样就能防止别人恶意进入单用户模式，从而修改root密码了!!

当然，如果我们连这个GRUB的秘密都忘记的话，并且又忘记了root密码，那就真的不能登陆到root用户了。。。。

本篇随笔主要记录了如何通过单用户模式来修改root用户的密码，并通过设置grub的MD5加密算法来对进入grub进行加密，从而限制别人轻易进入单用户模式，在以后的学习Linux过程中，将继续记录学习Linux的心得!!!!

## 13、mount

功能：加载指定的文件系统。

语法：mount [-afFhnrvVw] [-L] [-o] [-t] [设备名] [加载点]

用法说明：mount可将指定设备中指定的文件系统加载到Linux目录下（也就是装载点）。可将经常使用的设备写入文件/etc/fastab,以使系统在每次启动时自动加载。mount加载设备的信息记录在/etc/mtab文件中。使用umount命令卸载设备时，记录将被清除。

常用参数和选项：

-a 加载文件/etc/fstab中设置的所有设备。

-f 不实际加载设备。可与-v等参数同时使用以查看mount的执行过程。

-F 需与-a参数同时使用。所有在/etc/fstab中设置的设备会被同时加载，可加快执行速度。

-h 显示在线帮助信息。

-L 加载文件系统标签为的设备。

-n 不将加载信息记录在/etc/mtab文件中。

-r 以只读方式加载设备。

-w 以可读写模式加载设备，默认设置。

-v 执行时显示详细的信息。

-V 显示版本信息。

-t 指定设备的文件系统类型。常用的选项说明有：

minix Linux最早使用的文件系统。

ext2 Linux目前的常用文件系统。

msdos MS-DOS 的 FAT。

vfat Win85/98 的 VFAT。

nfs 网络文件系统。

iso9660 CD-ROM光盘的标准文件系统。

ntfs Windows NT的文件系统。

hpfs OS/2文件系统。Windows NT 3.51之前版本的文件系统。

auto 自动检测文件系统。

-o 指定加载文件系统时的选项。有些选项也可在/etc/fstab中使用。这些选项包括：

|  |  |
| --- | --- |
| -o选项 | 功能含义 |
| async  sync | 以异步的方式执行文件系统的输入输出动作。  以同步方式执行文件系统的输入输出动作。 |
| atime  noatime | 每次存取都更新inode的存取时间，默认设置，  取消选项为。 |
| auto  noauto | 必须在/etc/fstab文件中指定此选项。执行-a参数时，会加载设置为auto的设备，取消选取为noauto。 |
| dev  nodev | 可读文件系统上的字符或块设备，  取消选项为nodev。 |
| exec  noexec | 可执行二进制文件，  取消选项为noexec。 |
| suid  nosuid | set-user-identifier(设置用户ID)设置位  set-group-identifer(设置组ID)设置位  关闭用户设置位  关闭组设置位 |
| user  nouser | 设置普通用户可以执行挂载操作。  普通用户无法执行挂载操作，默认设置。 |
| ro  rw | 以只读模式加载。  以读写模式加载。 |
| defaults | 默认选项为rw、suid、dev、exec、anto nouser与async。就是fstab中的 |
| remount | 重新加载设备。通常用于改变设备的设置状态。 |
| codepage=XXX | 代码页 |
| iocharset=XXX | 字符集 |
| loop | 用来把一个文件当成硬盘分区挂接上系统 |

用来把一个文件当成硬盘分区挂接上系统

defaults 使用默认的选项。默认选项为rw、suid、dev、exec、anto nouser与async。

noatime 每次存取时不更新inode的存取时间。

noauto 无法使用-a参数来加载。

nodev 不读文件系统上的字符或块设备。

noexec 无法执行二进制文件。

nosuid 关闭set-user-identifier(设置用户ID)与set-group-identifer(设置组ID)设置位。

nouser 使一位用户无法执行加载操作，默认设置。

codepage=XXX 代码页

iocharset=XXX 字符集

remount 重新加载设备。通常用于改变设备的设置状态。

ro 以只读模式加载。

rw 以可读写模式加载。

suid 启动set-user-identifier(设置用户ID)与set-group-identifer(设置组ID)设置位，取消选项为nosuid。

sync 以同步方式执行文件系统的输入输出动作。

user 可以让一般用户加载设备。

简单用法:

fat32的分区

mount -o codepage=936,iocharset=cp936 /dev/hda7 /mnt/cdrom (mount -t vfat -o iocharset=cp936 /dev/hda7 /mnt/cdrom)

ntfs的分区

mount -o iocharset=cp936 /dev/hda7 /mnt/cdrom

iso文件

mount -o loop /abc.iso /mnt/cdrom

软盘

mount /dev/fd0 /mnt/floppy

USB闪存

mount /dev/sda1 /mnt/cdrom

所有/etc/fstab内容 mount -a

可以指定文件格式"-t 格式", 格式可以为vfat, ext2, ext3等.

镜像文件：

mount fileName mountPoint -o loop，fileName是镜像文件名(\*.iso,\*.img)，其它的不用说了，跟上面一样。用例：如我有一个a.iso光盘镜像文件，mount a.iso a -o loop，这样进入目录a你就能浏览a.iso的内容了，\*.img文件的用法一样。

显示中文方面,如果是fedora core 1 的话，用utf8更好

mount -o iocharset=uft8 /dev/hda1 /mnt/c

mount -o iocharset=uft8 /dev/sda1 /mnt/usb

mount -t smbfs //ip\_addr/share/dir /mnt/smb/share -o iocharset=uft8,username=name, password=passwd

开机就mount上windows下的分区

自动将windows的d盘挂到/mnt/d上，用vi打开/etc/fstab，加入以下一行

/dev/hda5 /mnt/d vfat defaults,codepage=936,iocharset=cp936 0 0

注意，先得手工建立一个/mnt/d目录

mount局域网上其他windows机器共享出的目录(bjchenxu)

mount -t smbfs -o username=guest,password=guest //machine/path /mnt/cdrom

挂接命令(mount)

首先，介绍一下挂接(mount)命令的使用方法，mount命令参数非常多，这里主要讲一下今天我们要用到的。

命令格式：

mount [-t vfstype] [-o options] device dir

其中：

1.-t vfstype 指定文件系统的类型，通常不必指定。mount 会自动选择正确的类型。常用类型有：

光盘或光盘镜像：iso9660

DOS fat16文件系统：msdos

Windows 9x fat32文件系统：vfat

Windows NT ntfs文件系统：ntfs

Mount Windows文件网络共享：smbfs

UNIX(LINUX) 文件网络共享：nfs

2.-o options 主要用来描述设备或档案的挂接方式。常用的参数有：

loop：用来把一个文件当成硬盘分区挂接上系统

ro：采用只读方式挂接设备

rw：采用读写方式挂接设备

iocharset：指定访问文件系统所用字符集

3.device 要挂接(mount)的设备。

4.dir设备在系统上的挂接点(mount point)。

挂接光盘镜像文件

由于近年来磁盘技术的巨大进步，新的电脑系统都配备了大容量的磁盘系统，在Windows下许多人都习惯把软件和资料做成光盘镜像文件通过虚拟光驱来使用。这样做有许多好处：一、减轻了光驱的磨损;二、现在硬盘容量巨大存放几十个光盘镜像文件不成问题，随用随调十分方便;三、硬盘的读取速度要远远高于光盘的读取速度，CPU占用率大大降低。其实linux系统下制作和使用光盘镜像比Windows系统更方便，不必借用任何第三方软件包。

1、从光盘制作光盘镜像文件。将光盘放入光驱，执行下面的命令。

#cp /dev/cdrom /home/sunky/mydisk.iso 或

#dd if=/dev/cdrom of=/home/sunky/mydisk.iso

注：执行上面的任何一条命令都可将当前光驱里的光盘制作成光盘镜像文件/home/sunky/mydisk.iso

2、将文件和目录制作成光盘镜像文件，执行下面的命令。

#mkisofs -r -J -V mydisk -o /home/sunky/mydisk.iso /home/sunky/ mydir

注：这条命令将/home/sunky/mydir目录下所有的目录和文件制作成光盘镜像文件/home/sunky/mydisk.iso，光盘卷标为：mydisk

3、光盘镜像文件的挂接(mount)

#mkdir /mnt/vcdrom

注：建立一个目录用来作挂接点(mount point)

#mount -o loop -t iso9660 /home/sunky/mydisk.iso /mnt/vcdrom

注：使用/mnt/vcdrom就可以访问盘镜像文件mydisk.iso里的所有文件了

挂接移动硬盘

对linux系统而言，USB接口的移动硬盘是当作SCSI设备对待的。插入移动硬盘之前，应先用fdisk –l 或 more /proc/partitions查看系统的硬盘和硬盘分区情况。

[root at pldyrouter /]# fdisk -l

Disk /dev/sda: 73 dot 4 GB, 73407820800 bytes

255 heads, 63 sectors/track, 8924 cylinders

Units = cylinders of 16065 \* 512 = 8225280 bytes

Device Boot Start End Blocks Id System

/dev/sda1 1 4 32098+ de Dell Utility

/dev/sda2 \* 5 2554 20482875 7 HPFS/NTFS

/dev/sda3 2555 7904 42973875 83 Linux

/dev/sda4 7905 8924 Win95 Ext'd (LBA)

/dev/sda5 7905 8924 8193118+ 82 Linux swap

在这里可以清楚地看到系统有一块SCSI硬盘/dev/sda和它的四个磁盘分区/dev/sda1 -- /dev/sda4, /dev/sda5是分区/dev/sda4的逻辑分区。接好移动硬盘后，再用fdisk –l 或 more /proc/partitions查看系统的硬盘和硬盘分区情况

[root at pldyrouter /]# fdisk -l

Disk /dev/sda: 73 dot 4 GB, 73407820800 bytes

255 heads, 63 sectors/track, 8924 cylinders

Units = cylinders of 16065 \* 512 = 8225280 bytes

Device Boot Start End Blocks Id System

/dev/sda1 1 4 32098+ de Dell Utility

/dev/sda2 \* 5 2554 20482875 7 HPFS/NTFS

/dev/sda3 2555 7904 42973875 83 Linux

/dev/sda4 7905 8924 Win95 Ext'd (LBA)

/dev/sda5 7905 8924 8193118+ 82 Linux swap

Disk /dev/sdc: 40.0 GB, 40007761920 bytes

255 heads, 63 sectors/track, 4864 cylinders

Units = cylinders of 16065 \* 512 = 8225280 bytes

Device Boot Start End Blocks Id System

/dev/sdc1 1 510 4096543+ 7 HPFS/NTFS

/dev/sdc2 511 4864 Win95 Ext'd (LBA)

/dev/sdc5 511 4864 34973473+ b Win95 FAT32

大家应该可以发现多了一个SCSI硬盘/dev/sdc和它的两个磁盘分区/dev/sdc1?、/dev/sdc2,其中/dev/sdc5是/dev/sdc2分区的逻辑分区。我们可以使用下面的命令挂接/dev/sdc1和/dev/sdc5。

#mkdir -p /mnt/usbhd1

#mkdir -p /mnt/usbhd2

注：建立目录用来作挂接点(mount point)

#mount -t ntfs /dev/sdc1 /mnt/usbhd1

#mount -t vfat /dev/sdc5 /mnt/usbhd2

注：对ntfs格式的磁盘分区应使用-t ntfs 参数，对fat32格式的磁盘分区应使用-t vfat参数。若汉字文件名显示为乱码或不显示，可以使用下面的命令格式。

#mount -t ntfs -o iocharset=cp936 /dev/sdc1 /mnt/usbhd1

#mount -t vfat -o iocharset=cp936 /dev/sdc5 /mnt/usbhd2

linux系统下使用fdisk分区命令和mkfs文件系统创建命令可以将移动硬盘的分区制作成linux系统所特有的ext2、ext3格式。这样，在linux下使用就更方便了。使用下面的命令直接挂接即可。

#mount /dev/sdc1 /mnt/usbhd1

挂接U盘

和USB接口的移动硬盘一样对linux系统而言U盘也是当作SCSI设备对待的。使用方法和移动硬盘完全一样。插入U盘之前，应先用fdisk –l 或 more /proc/partitions查看系统的硬盘和硬盘分区情况。

[root at pldyrouter root]# fdisk -l

Disk /dev/sda: 73 dot 4 GB, 73407820800 bytes

255 heads, 63 sectors/track, 8924 cylinders

Units = cylinders of 16065 \* 512 = 8225280 bytes

Device Boot Start End Blocks Id System

/dev/sda1 1 4 32098+ de Dell Utility

/dev/sda2 \* 5 2554 20482875 7 HPFS/NTFS

/dev/sda3 2555 7904 42973875 83 Linux

/dev/sda4 7905 8924 Win95 Ext'd (LBA)

/dev/sda5 7905 8924 8193118+ 82 Linux swap

插入U盘后，再用fdisk –l 或 more /proc/partitions查看系统的硬盘和硬盘分区情况。

[root at pldyrouter root]# fdisk -l

Disk /dev/sda: 73 dot 4 GB, 73407820800 bytes

255 heads, 63 sectors/track, 8924 cylinders

Units = cylinders of 16065 \* 512 = 8225280 bytes

Device Boot Start End Blocks Id System

/dev/sda1 1 4 32098+ de Dell Utility

/dev/sda2 \* 5 2554 20482875 7 HPFS/NTFS

/dev/sda3 2555 7904 42973875 83 Linux

/dev/sda4 7905 8924 Win95 Ext'd (LBA)

/dev/sda5 7905 8924 8193118+ 82 Linux swap

Disk /dev/sdd: 131 MB, 131072000 bytes

9 heads, 32 sectors/track, 888 cylinders

Units = cylinders of 288 \* 512 = 147456 bytes

Device Boot Start End Blocks Id System

/dev/sdd1 \* 1 889 127983+ b Win95 FAT32

Partition 1 has different physical/logical endings:

phys=(1000, 8, 32) logical=(888, 7, 31)

系统多了一个SCSI硬盘/dev/sdd和一个磁盘分区/dev/sdd1,/dev/sdd1就是我们要挂接的U盘。

#mkdir -p /mnt/usb

注：建立一个目录用来作挂接点(mount point)

#mount -t vfat /dev/sdd1 /mnt/usb

注：现在可以通过/mnt/usb来访问U盘了, 若汉字文件名显示为乱码或不显示，可以使用下面的命令。

#mount -t vfat -o iocharset=cp936 /dev/sdd1 /mnt/usb

类似于windows的网络共享，UNIX(Linux)系统也有自己的网络共享，那就是NFS(网络文件系统)，下面我们就以SUN Solaris2.8和REDHAT as server 3 为例简单介绍一下在linux下如何mount nfs网络共享。在linux客户端挂接(mount)NFS磁盘共享之前，必须先配置好NFS服务端。

1、Solaris系统NFS服务端配置方法如下：

(1)修改 /etc/dfs/dfstab, 增加共享目录

share -F nfs -o rw /export/home/sunky

(2)启动nfs服务

# /etc/init.d/nfs.server start

(3)NFS服务启动以后，也可以使用下面的命令增加新的共享

# share /export/home/sunky1

# share /export/home/sunky2

注：/export/home/sunky和/export/home/sunky1是准备共享的目录

如果想在运行的Linux下访问其它文件系统中的资源的话，就要用mount命令来实现。

mount的基本格式：

mount [-t vfstype] [-o options] device dir

mount [-参数] 设备名称 挂载点

其中常用的参数有：

-a 挂载在/etc/fstab文件中类出的所有文件系统。

-f 伪装mount，作出检查设备和目录的样子，但并不真正挂载文件系统。

-n 不把挂载记录在/etc/mtab 文件中。

-r 将文件系统安装为只读。

-v 详细显示安装信息。

-w 将文件系统安装为可写，为命令默认情况。

-t vfstype 指定文件系统的类型，通常不必指定。mount 会自动选择正确的类型。常用类型有：

ext2 linux目前常用的文件系统还有ext3，ext4等

msdos MS-DOS的fat，就是fat16

vfat windows98常用的fat32

nfs 网络文件系统

iso9660 CD-ROM光盘标准文件系统

ntfs windows NT/2000/XP的文件系统

auto 自动检测文件系统

## -o 指定挂载文件系统时的选项，有些也可写到在/etc/fstab中。常用的有：

defaults 使用所有选项的默认值（auto、nouser、rw、suid）

auto/noauto 允许/不允许以–a选项进行安装

dev/nodev 对/不对文件系统上的特殊设备进行解释

exec/noexec 允许/不允许执行二进制代码

suid/nosuid 确认/不确认suid和sgid位

user /nouser 允许/不允许一般用户挂载

codepage=XXX 代码页

iocharset=XXX 字符集

ro 以只读方式挂载

rw 以读写方式挂载

remount 重新挂载已经安装了的文件系统

需要注意的是，挂载点必须是一个已经存在的目录，这个目录可以不为空，但挂载后这个目录下以前的内容将不可用，umount以后会恢复正常。使用多个-o参数的时候，-o 只用一次，参数之间用半角逗号隔开：

[Copy to clipboard]

CODE:

# mount –o remount,rw /

例如要挂载windows下文件系统为FAT32的D盘，一般而言在Linux下这个分区对应/dev/hda5，根据具体的分区情况会有不同，这里就以hda5来举例说明：

[Copy to clipboard]

CODE:

# mkdir /mnt/hda5 //创建hda5的目录作为挂载点，位置和目录名可自定义//

# mount -t vfat /dev/hda5 /mnt/hda5

一般而言，Linux会自动探测分区的文件系统，除非让你指定时，否则-t vfat 可以省掉。

[Copy to clipboard]

CODE:

# mount /dev/hda5 /mnt/hda5

这样就可以进入/mnt/hda5目录去访问分区中的资源了。

3. 为什么mount上分区后显示不了中文文件为问号/乱码？

显示问号表明你的系统中没有可识别使用的中文字体，请先安装中文字体。确保你的系统已经可以很好的显示中文。显示为乱码一般是mount默认使用的文件系统编码和文件系统中文件的实际编码不一致造成的。要想正常显示中文文件，mount时需要用到 -o 参数里的codepage和iocharset选项。codepage指定文件系统的代码页，简体中文中文代码是936；iocharset指定字符集，简体中文一般用cp936或gb2312。

[Copy to clipboard]

CODE:

# mount –o iocharset=gb2312 codepage=936 /dev/hda5 /mnt/hda5

一般来说 mount –o iocharset=cp936 /dev/hda5 /mnt/hda5 就可以解决问题了。

如果这样做了以后还有问题，请尝试UTF-8编码：

[Copy to clipboard]

CODE:

# mount –o iocharset=utf8 /dev/hda5 /mnt/hda5

4. 为什么mount上去以后分区普通用户不可写？

mount时加上 –o umask=000 即可：

[Copy to clipboard]

CODE:

# mount –o umask=000, iocharset=cp936 /dev/hda5 /mnt/hda5

5. 为什么mount上去后的分区中的文件都变成短文件名了？

这是文件系统挂错的原因，将FAT32挂载成FAT16时就会出现这种情况，先umount，然后用 –t vfat 重新挂载即可解决问题。

[Copy to clipboard]

CODE:

# mount –t vat /dev/hda5 /mnt/hda5

6. 为什么不能mount ntfs分区？

这是内核不支持NTFS文件系统的原因，请重新编译内核或者安装内核的NTFS文件系统支持包，以使得内核有NTFS文件系统的支持。

7. 如何挂载U盘和mp3？

如果计算机没有其它SCSI设备和usb外设的情况下，插入的U盘的设备路径是 /dev/sda1，用命令：

[Copy to clipboard]

CODE:

# mkdir /mnt/u

# mount /dev/sda1 /mnt/u

挂载即可。

8. 可以直接使用iso文件吗？

可以，就是mount的这一选项使得Linux下有免费虚拟光驱的说法，具体用法是：

[Copy to clipboard]

CODE:

# mkdir /mnt/iso

# mount –o loop linux.iso /mnt/iso

当然，挂载以后挂载点/mnt/iso也是只读的。

9. 我怎么不可以mount iso文件？

一般而言，大多数的发行版使用的内核均已将loop设备的支持编译进去了，但是也有没有的情况，所以请确保系统所使用的内核支持loop设备。

第二种情况是iso文件被放置到了NTFS或其它只读文件系统中了。挂载loop 设备必须要求挂载到一个可写的分区中，目前Linux内核对NTFS文件系统的写支持非常有限，请将iso文件复制到其它可写文件系统中后再挂载。

10. 如何挂载光驱和软驱

一般来说CDROM的设备文件是/dev/hdc，软驱的设备名是/dev/fd0

[Copy to clipboard]

CODE:

# mkdir /mnt/cdrom

# mount /dev/hdc /mnt/cdrom //挂载光驱 //

# mkdir /mnt/floppy

# mount /dev/fd0 /mnt/floppy //挂载软驱 //

11. 为何挂载的CD-ROM不能显示中文文件？

使用 –o iocharset=cp936 选项一般能解决问题，否则使用utf-8编码。

[Copy to clipboard]

CODE:

# mount –o iocharset=cp936 /dev/hdc /mnt/cdrom

12. 如何开机自动挂载分区？

每次挂载都要输入那么长的命令的确是繁琐了些，只要将分区信息写到/etc/fstab文件中即可实现系统启动的自动挂载，例如对于/dev/hda5的自动挂载添加如下的行即可：

[Copy to clipboard]

CODE:

/dev/hda5 /mnt/hda5 vfat defaults,iocharset=cp936, rw 0 0

13. 如何挂载samba 分区？

[Copy to clipboard]

CODE:

# mkdir /mnt/share

# mount -t smbfs -o username=root,password=abc,codepage=936,iocharset=gb2312//192.168.1.100/share /mnt/share

如果中文显示不正常请尝试UTF-8编码。当然可以写到fstab中实现自动挂载。

14. mount --bind是什么意思？

mount --bind 是将一个目录中的内容挂载到另一个目录上，用法是

[Copy to clipboard]

CODE:

# mount --bind olddir newdir

这个命令使得自己搭建的FTP要共享某个目录的时候变得特别方便。如果要取消mount用命令：

[Copy to clipboard]

CODE:

# mount --move olddir newdir 即可。

如果mount --bind 也想写入fstab中的话格式如下：

[Copy to clipboard]

CODE:

olddir newdir none bind 0 0

15. umount基本用法是？

譬如 /dev/hda5 已经挂载在/mnt/hda5上,用一下三条命令均可卸载挂载的文件系统

[Copy to clipboard]

CODE:

# umount /dev/hda5

# umount /mnt/hda5

# umount /dev/hda5 /mnt/hda5

16. 为什么umount的时候老显示 device busy？

这是因为有程序正在访问这个设备，最简单的办法就是让访问该设备的程序退出以后再umount。可能有时候用户搞不清除究竟是什么程序在访问设备，如果用户不急着umount，则可以用:

[Copy to clipboard]

CODE:

# umount -l /mnt/hda5

来卸载设备。选项 –l 并不是马上umount，而是在该目录空闲后再umount。还可以先用命令 ps aux 来查看占用设备的程序PID，然后用命令kill来杀死占用设备的进程，这样就umount的非常放心了。

## 14、字符编码codepage iocharset

为什么会把这两个选项单独拿出来说一下，主要是很多同学对这个字符设置完全无法理解。装完linux+winXP双系统后,在linux中挂载winXP分区,有乱码现象,应该是字符编码的问题。查查资料,解决如下:

linux下挂载FAT32分区或者挂在移动硬盘时中文会出现乱码的现象（表现为中文变成一连串的问号），这是因为编码格式的问题，linux内核中默认的编码格式是codepage=417、iocharset=ascii，经过实验可以通过下面的方式来加以解决。

1、挂载时增加两个参数codepage=936、iocharset=utf8。这样挂载挂文件系统时会用上述的两个指定参数替换掉默认的编码参数从而解决乱码的问题。

2、 重新编译内核，在File System->DOS/FAT/NT Filesystems中可以看到Default codepage for FAT和Default iocharset for FAT两个选项。分别修改成936和utf8。重新编译内核后用新内核启动系统，可以基本上永久解决乱码的现象。

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

既然有问题,那就继续深入了解一下codepage与iocharset吧,毕竟不懂就得学嘛

作者 FireMeteor

参考资料：内核文档，mount的manpage。windows部分基本上凭自己的记忆－－不可靠

由于资料来源与本人理解表述的关系，可能存在错漏之处

codepage是m$搞出来的东西。早期的操作系统都是直接使用本地语言字符集(nativelanguage character set,NLS)，屏幕显示如此，内部表示也如此。dos就是一个典型的例子。m$把这种本地字符集称作codepage。常见的codepage有CP437（美、加），CP850（欧），CP932（日），CP936（简体中文,gbk），CP950(繁体中文BIG5)等。在dos时代，fat文件系统里面存储的文件名也使用codepage。这些东西似乎是unicode出来以前，m$根据一些国家的国家标准搞的，所以相互之间编码冲突比较严重，一旦代码页设置错误，显示结果就乱七八糟。

windows9x似乎主要还是采用代码页机制，但是对unicode也有了部分支持。其新增的长文件名支持使用unicode存放文件名。在linux中叫做vfat。也就是说，短文件名用codepage，长文件名用unicode。记得以前在windows9x里面打日文游戏的时候还需要日文环境的支持，否则全是乱码。但是不管怎么支持，屏幕中总有一部分乱码，不是中文乱，就是日文乱。这大概就是codepage只支持一种本地语言的缘故。

到2000以后，系统内核处理字符串的时候总是先转换成unicode，在显示输出的时候，再视情况转回本地字符集，所以情况要好很多。理论上说，所有unicode程序，在2000下都能同时正常显示，也就是说中文、日文、韩文等都可以同屏显示。但是使用本地字符集

的程序还是同时只有一种语言能够正常显示。

下面摘一段win XP区域和语言选项“对话框里面关于本地语言的说明文字：这个设置启动某些非Unicode程序以便用母语显示菜单和对话。这不会影响到Unicode的程序“以上都是背景资料，也就是说都是废话-\_-

在linux下mount东西的时候，只有在mount和m$有关的东西的时候才需要设置codepage。最典型的就是fat。joliet格式的CDROM可能也要，但是我不用cd好多年，哪位帮忙确认一下……还有就是smbfs，在samba server端配置好dos charset参数，然后mount的时候codepage写成跟server一样即可。关于samba，跑题一句，其实doscharset参数在用smbclient访问的时候根本不用设，因为新的客户端（win2000,xp）都用unicode通信，这个参数只对使用codepage的dos，9x有效，但是因为smbmount也使用codepage，所以……codepage设置以后起的作用是用来在读写文件系统时进行编码与解码。因为在linux内部表示字符串的时候也是用unicode的，这里存在一次unicode到本地字符集之间的转换。

以上是关于codepage的部分，下面是iocharset部分

iocharset比较简单，它只与显示输出有关。这里不得不提起locale，因为locale

控制了软件关于输入输出的很多细节。可以分别配置locale的每个选项，但是这跟我现在要说的没多大关系，所以我假设只配置LANG或者LC\_ALL。说了那么多废话，其实iocharset控制的就是系统内部unicode表示到显示输出的表示间的转换。也就是说，iocharset要与你使用的locale匹配，否则显示乱码那是理所当然的了。

总结一下：

文件系统编码<--codepage--->内部unicode表示<---iocharset/NLS--->显示输出编码其实上面说那么多都是废话。最王道的办法应该是把locale设成utf8(zh\_CN.UTF-8或者en\_US.UTF-8随便) ，然后在mount fat或者nfts的时候用一个-o utf8参数，整个世界都清静了。codepage？iocharset？那是啥？不过话又说回来locale设成utf8有一点不好的地方，如果文件内容是用本地语言编码的，而且软件不支持编码转换的话，可能会无法正常显示。但是这种情况使用gb2312或者gbk也不能完全避免，只是发生的概率要小一些罢了。实际上即使在utf8下我也只碰到过一个实例：xmms不能处理某些mp3的标签，因为ID3V1是用bgk编码的,所以我单独用bgk编码启动xmms～～

文章引用自： http://blog.sina.com.cn/s/blog\_4b807d53010006wj.html

---------------------------------------------------------------------------------------

这篇也可以参考:http://blog.csdn.net/lesterjames/archive/2005/09/28/491619.aspx

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

再接着拓展一下知识面吧,看看各种字符的编码吧!发现有了网络真的很方便.不过找找资料,发现字符编码资料是很多,也有些乱.很多还好象有点矛盾.发现这篇比较合我的胃口,既然花了时间了,就记下吧;

字符编码笔记：ASCII，Unicode和UTF-8

文章出处: http://www.ruanyifeng.com/blog/2007/10/ascii\_unicode\_and\_utf-8.html

1. ASCII码

我们知道，在计算机内部，所有的信息最终都表示为一个二进制的字符串。每一个二进制位（bit）有0和1两种状态，因此八个二进制位就可以组合出256种状态，这被称为一个字节（byte）。也就是说，一个字节一共可以用来表示256种不同的状态，每一个状态对应一个符号，就是256个符号，从0000000到11111111。

上个世纪60年代，美国制定了一套字符编码，对英语字符与二进制位之间的关系，做了统一规定。这被称为ASCII码，一直沿用至今。

ASCII码一共规定了128个字符的编码，比如空格“SPACE”是32（二进制00100000），大写的字母A是65（二进制01000001）。这128个符号（包括32个不能打印出来的控制符号），只占用了一个字节的后面7位，最前面的1位统一规定为0。

2、非ASCII编码

英语用128个符号编码就够了，但是用来表示其他语言，128个符号是不够的。比如，在法语中，字母上方有注音符号，它就无法用ASCII码表示。于是，一些欧洲国家就决定，利用字节中闲置的最高位编入新的符号。比如，法语中的é的编码为130（二进制10000010）。这样一来，这些欧洲国家使用的编码体系，可以表示最多256个符号。

但是，这里又出现了新的问题。不同的国家有不同的字母，因此，哪怕它们都使用256个符号的编码方式，代表的字母却不一样。比如，130在法语编码中代表了é，在希伯来语编码中却代表了字母Gimel (?)，在俄语编码中又会代表另一个符号。但是不管怎样，所有这些编码方式中，0—127表示的符号是一样的，不一样的只是128—255的这一段。

至于亚洲国家的文字，使用的符号就更多了，汉字就多达10万左右。一个字节只能表示256种符号，肯定是不够的，就必须使用多个字节表达一个符号。比如，简体中文常见的编码方式是GB2312，使用两个字节表示一个汉字，所以理论上最多可以表示256x256=65536个符号。

中文编码的问题需要专文讨论，这篇笔记不涉及。这里只指出，虽然都是用多个字节表示一个符号，但是GB类的汉字编码与后文的Unicode和UTF-8是毫无关系的。

3.Unicode

正如上一节所说，世界上存在着多种编码方式，同一个二进制数字可以被解释成不同的符号。因此，要想打开一个文本文件，就必须知道它的编码方式，否则用错误的编码方式解读，就会出现乱码。为什么电子邮件常常出现乱码？就是因为发信人和收信人使用的编码方式不一样。

可以想象，如果有一种编码，将世界上所有的符号都纳入其中。每一个符号都给予一个独一无二的编码，那么乱码问题就会消失。这就是Unicode，就像它的名字都表示的，这是一种所有符号的编码。

Unicode当然是一个很大的集合，现在的规模可以容纳100多万个符号。每个符号的编码都不一样，比如，U+0639表示阿拉伯字母Ain，U+0041表示英语的大写字母A，U+4E25表示汉字“严”。具体的符号对应表，可以查询unicode.org，或者专门的汉字对应表。

4. Unicode的问题

需要注意的是，Unicode只是一个符号集，它只规定了符号的二进制代码，却没有规定这个二进制代码应该如何存储。

比如，汉字“严”的unicode是十六进制数4E25，转换成二进制数足足有15位（100111000100101），也就是说这个符号的表示至少需要2个字节。表示其他更大的符号，可能需要3个字节或者4个字节，甚至更多。

这里就有两个严重的问题，第一个问题是，如何才能区别unicode和ascii？计算机怎么知道三个字节表示一个符号，而不是分别表示三个符号呢？第二个问题是，我们已经知道，英文字母只用一个字节表示就够了，如果unicode统一规定，每个符号用三个或四个字节表示，那么每个英文字母前都必然有二到三个字节是0，这对于存储来说是极大的浪费，文本文件的大小会因此大出二三倍，这是无法接受的。

它们造成的结果是：1）出现了unicode的多种存储方式，也就是说有许多种不同的二进制格式，可以用来表示unicode。2）unicode在很长一段时间内无法推广，直到互联网的出现。

5.UTF-8

互联网的普及，强烈要求出现一种统一的编码方式。UTF-8就是在互联网上使用最广的一种unicode的实现方式。其他实现方式还包括UTF-16和UTF-32，不过在互联网上基本不用。重复一遍，这里的关系是，UTF-8是Unicode的实现方式之一。

UTF-8最大的一个特点，就是它是一种变长的编码方式。它可以使用1~4个字节表示一个符号，根据不同的符号而变化字节长度。

UTF-8的编码规则很简单，只有二条：

1）对于单字节的符号，字节的第一位设为0，后面7位为这个符号的unicode码。因此对于英语字母，UTF-8编码和ASCII码是相同的。

2）对于n字节的符号（n>1），第一个字节的前n位都设为1，第n+1位设为0，后面字节的前两位一律设为10。剩下的没有提及的二进制位，全部为这个符号的unicode码。

下表总结了编码规则，字母x表示可用编码的位。

Unicode符号范围 | UTF-8编码方式

(十六进制) | （二进制）

--------------------+---------------------------------------------

0000 0000-0000 007F | 0xxxxxxx

0000 0080-0000 07FF | 110xxxxx 10xxxxxx

0000 0800-0000 FFFF | 1110xxxx 10xxxxxx 10xxxxxx

0001 0000-0010 FFFF | 11110xxx 10xxxxxx 10xxxxxx 10xxxxxx

下面，还是以汉字“严”为例，演示如何实现UTF-8编码。

已知“严”的unicode是4E25（100111000100101），根据上表，可以发现4E25处在第三行的范围内（0000 0800-0000 FFFF），因此“严”的UTF-8编码需要三个字节，即格式是“1110xxxx 10xxxxxx 10xxxxxx”。然后，从“严”的最后一个二进制位开始，依次从后向前填入格式中的x，多出的位补0。这样就得到了，“严”的UTF-8编码是“11100100 10111000 10100101”，转换成十六进制就是E4B8A5。

6. Unicode与UTF-8之间的转换

通过上一节的例子，可以看到“严”的Unicode码是4E25，UTF-8编码是E4B8A5，两者是不一样的。它们之间的转换可以通过程序实现。

在Windows平台下，有一个最简单的转化方法，就是使用内置的记事本小程序Notepad.exe。打开文件后，点击“文件”菜单中的“另存为”命令，会跳出一个对话框，在最底部有一个“编码”的下拉条。

里面有四个选项：ANSI，Unicode，Unicode big endian 和 UTF-8。

1）ANSI是默认的编码方式。对于英文文件是ASCII编码，对于简体中文文件是GB2312编码（只针对Windows简体中文版，如果是繁体中文版会采用Big5码）。

2）Unicode编码指的是UCS-2编码方式，即直接用两个字节存入字符的Unicode码。这个选项用的little endian格式。

3）Unicode big endian编码与上一个选项相对应。我在下一节会解释little endian和big endian的涵义。

4）UTF-8编码，也就是上一节谈到的编码方法。

选择完”编码方式“后，点击”保存“按钮，文件的编码方式就立刻转换好了。

7. Little endian和Big endian

上一节已经提到，Unicode码可以采用UCS-2格式直接存储。以汉字”严“为例，Unicode码是4E25，需要用两个字节存储，一个字节是4E，另一个字节是25。存储的时候，4E在前，25在后，就是Big endian方式；25在前，4E在后，就是Little endian方式。

这两个古怪的名称来自英国作家斯威夫特的《格列佛游记》。在该书中，小人国里爆发了内战，战争起因是人们争论，吃鸡蛋时究竟是从大头(Big-Endian)敲开还是从小头(Little-Endian)敲开。为了这件事情，前后爆发了六次战争，一个皇帝送了命，另一个皇帝丢了王位。

因此，第一个字节在前，就是”大头方式“（Big endian），第二个字节在前就是”小头方式“（Little endian）。

那么很自然的，就会出现一个问题：计算机怎么知道某一个文件到底采用哪一种方式编码？

Unicode规范中定义，每一个文件的最前面分别加入一个表示编码顺序的字符，这个字符的名字叫做”零宽度非换行空格“（ZERO WIDTH NO-BREAK SPACE），用FEFF表示。这正好是两个字节，而且FF比FE大1。

如果一个文本文件的头两个字节是FE FF，就表示该文件采用大头方式；如果头两个字节是FF FE，就表示该文件采用小头方式。

8. 实例

下面，举一个实例。

打开”记事本“程序Notepad.exe，新建一个文本文件，内容就是一个”严“字，依次采用ANSI，Unicode，Unicode big endian 和 UTF-8编码方式保存。

然后，用文本编辑软件UltraEdit中的”十六进制功能“，观察该文件的内部编码方式。

1）ANSI：文件的编码就是两个字节“D1 CF”，这正是“严”的GB2312编码，这也暗示GB2312是采用大头方式存储的。

2）Unicode：编码是四个字节“FF FE 25 4E”，其中“FF FE”表明是小头方式存储，真正的编码是4E25。

3）Unicode big endian：编码是四个字节“FE FF 4E 25”，其中“FE FF”表明是大头方式存储。

4）UTF-8：编码是六个字节“EF BB BF E4 B8 A5”，前三个字节“EF BB BF”表示这是UTF-8编码，后三个“E4B8A5”就是“严”的具体编码，它的存储顺序与编码顺序是一致的。

## 15、关于lixunx下光驱的那点事

注意eject这个命令不是任何机器都好用的。通常关闭光驱也可以用这个命令，但通常也不是所有的光驱都支持的。

## 16、fdisk mkfs

这是系统挂载的第二块硬盘的第一个分区，

linux挂载第二块硬盘，我们手动进行了分区和格式化，并且建立了自动挂载

下面把这个过程给大家演示一下

首先用fdisk命令来建立分区，并写入分区表

fdisk /dev/sdb

用m命令查看，命令列表，p查看分区表

然后用n命令，来建立一个新的分区

有e、p两个选项，e是建立扩展分区，p是建立主分区，系统最多支持4个主分区

首先建立一个主分区

然后选择起始柱面

分配大小，

接着再用n命令，建立一个新的主分区

我们已经建立了2个主分区，然后我们试着建立一个扩展分区，

扩展分区也要占一个主分区号，我们可以选择3、或者4，然后选择剩余空间，并且

只能建立一个扩展分区。

这时再输入n命令，发现可以建立逻辑分区了，选择l

然后在扩展分区中，建立两个逻辑分区，

最后，执行w命令，把建立的分区，写入分区表，退出fdisk。

给硬盘分区后，我们还不能使用，还必须格式化，才能使用。

格式化我们用mkfs命令，

mkfs -t ext4 /dev/sdb1

mkfs -t ext4 /dev/sdb2

mkfs -t ext4 /dev/sdb5

mkfs -t ext4 /dev/sdb6

这样这块硬盘，我们就分完区，并且格式化了，就可以使用了，大家发现没有，扩

展分区是从sdb5开始的。没错，逻辑分区就是从5开始的，因为1-4是为主分区分配

的，扩展分区中的逻辑分区，只能从5开始了。只一点要特别注意。

最后，我们来自动挂载，因为硬盘是固定的，所以可以把它写入fstab中。

fstab是linux系统中，自动挂载的配置文件，这是一个非常重要的系统文件，系统

启动时自动安配置挂载，一旦出现错误，系统将无法启动，所以必须要仔细填写。

这个文件的格式起始非常简单

设备名或者UUID 挂载点 文件类型 挂载参数 优先级 优先级

/dev/sdb1 /mnt/disk21 ext4 default 0 1

/dev/sdb2 /mnt/disk22 ext4 default 0 1

/dev/sdb5 /mnt/disk25 ext4 default 1 1

/dev/sdb6 /mnt/disk26 ext4 default 1 6

这样就编辑完成了，看来linux下就是挂块硬盘也是比较繁琐的，但其实并不复杂

。

## 17、Linux下的自动启动设置

Linux系统自动启动通常分为两种方式，一种手动添加方式，一种是自动启动添加方式。

手动模式，就是在/etc/rc.d/rc.local文件中添加启动命令

自动模式，是把软件启动服务，添加到系统服务列表中，并设置自动启动。

添加方式有两种一种是把启动脚本添加到/etc/rc.d/init.d目录下，这时可通过service命令启动。

另外一种就是通过修改启动脚本，让后chkconfig –add 脚本名称

然后 chkconfig –level 2345 脚本名称 on

来设置自动启动。

## 18、linux下查看和添加PATH环境变量

了解在Linux不同地方的设置，对合理设置环境变量有指导性的作用。方法/步骤

1. /etc/profile此文件为系统的每个用户设置环境信息，当用户第一次登录时，该文件被执行。并从“/etc/profile.d”目录的配置文件中搜集shell的设置。所以如果你有对/etc/profile有修改的话必须得重启你的修改才会生效，此修改对每个用户都生效。
2. /etc/bashrc为每一个运行bash shell的用户执行此文件。当bash shell被打开时，该文件被读取。如果你想对所有的使用bash的用户修改某个配置并在以后打开的bash都生效的话可以修改这个文件，修改这个文件不用重启，重新打开一个bash即可生效。
3. ~/.bash\_profile”每个用户都可使用该文件输入专用于自己使用的shell信息，当用户登录时，该文件仅仅执行一次！默认情况下，它设置一些环境变量，执行用户的.bashrc文件。此文件类似于/etc/profile，也是需要重启才会生效，/etc/profile对所有用户生效，~/.bash\_profile只对当前用户生效。
4. ~/.bashrc“该文件包含专用于你的bash shell的bash信息，当登录时以及每次打开新的shell时，该文件被读取。（每个用户都有一个.bashrc文件，在用户目录下）。此文件类似于/etc/bashrc，不需要重启生效，重新打开一个bash即可生效， /etc/bashrc对所有用户新打开的bash都生效，但~/.bashrc只对当前用户新打开的bash生效。
5. ~/.bash\_logout”当每次退出系统（退出bash shell）时，执行该文件。

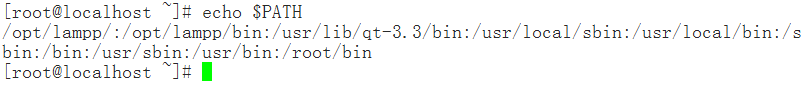
END

注意事项

“/etc/profile”，“/etc/bashrc”是系统全局环境变量设定。

“~/.profile”，“~/.bashrc”用户家目录下的私有环境变量设定。

1、单独查看PATH环境变量，可用：echo $PATH



2、临时添加PATH环境变量，可用：export

[root@localhost ~ ]#export PATH=/opt/lampp/bin:/opt/lamp/:$PATH

再次查看：如果有刚才添加路径，则说明添加PATH成功。

3、上述方法的PATH 在终端关闭 后就会消失。所以还是建议通过编辑/etc/profile来改PATH，也可以改家目录下的.bashrc(即：~/.bashrc)。

# vim /etc/profile

在文档最后，添加:

export PATH="/opt/lampp/bin:/opt/lamp/:$PATH"

保存，退出，然后运行：

#source /etc/profile

不报错则成功。

## 19、chmod和chown

这两个命令其实就是用户权限管理中比较常用的两个了，chmod就是修改用户权限的，而chown则是修改文件的宿主关系。基本用法如下：

chmod -R 777 /home/test/

chmod -R a+777 /home/test/

chmod -R ugo+rwx /home/test/

这是三条执行目标一样的命令，都是给test目录增加rwx权限，其中-R是递归，当然像v这样的参数是显示详细信息的。

chown -Rv caojun:root /home/test/

更改test目录的所有者和所属组，并且包括其子目录,要特别注意参数的大小写。

## 20、which whereis locate find

linux中如何查找某个命令所处的目录位置，如何查找在根目录下查找某个文件的位置

which－命令的位置，在变量环境PATH中查找

which -a command\_name， -a列出所有指令，不加-a只列第一个找到的命令

whereis－搜索特定文件

-b，只找binary文件 -m，只找说明文件 -s只找source来源文件 -u没有说明档的文件

whereis -b passwd -> passwd: /usr/bin/passwd /etc/passwd /etc/passwd.OLD

locate－输入文件部分名称即可搜索

find不常用，速度慢，很耗硬盘:

find [PATH] [option] [action]

find /etc -name '\*httpd\*'

## 21、su sudo

su:( Switch user切换用户)，可让一个普通用户切换为超级用户或其他用户，并可临时拥有所切换用户的权限，切换时需输入欲切换用户的密码；也可以让超级用户切换为普通用户，临时以低权限身份处理事务，切换时无需输入欲切换用户的密码。

su[-flmp][--help][--version][-][-c<指令>][-s<shell>][欲切换账户]

-c<指令>或--command=<指令>:执行一个指令后就恢复原来的身份；

-f或--fast:使shell不用读取启动文件，适用于csh和tsch；

-，-l,--login:切换用户时，使环境变量(home，shell，user，logname，path等）和欲切换的用户相同、不使用则取得用户的临时权限；

-m，-p，:变更身份时，保留环境变量不变；

-s<shell>,--shell=<shell>:指定执行命令的shell；

--help:显示帮助；

--version:显示版本信息；

用户账户:指定欲切换用户，默认切换为root。

注:root权限极高，几乎可以做任何事，谨慎使用，平时使用普通用户登录，必要时临时切换为root，用完即切换回普通用户。

实例

假设一个普通用户trunk，首先以trunk登陆：

su root

临时切换到root用户，需要输入密码，切换后环境变量不变，取得root的部分权限，且只能使用trunk用户path路径中的命令，不能使用root用户path路径中的独有命令。

su - root 切换为root用户，需要输入密码，切换后环境变量改变，几乎可以不受限制的做任何事。

su - trunk 从root切换回普通用户，无需输入密码，切换后只拥有trunk权限。

### 语法

sudo [ -Vhl LvkKsHPSb ] │ [ -p prompt ] [ -c class│- ] [ -a auth\_type ] [-u username│#uid ] command

### 参数

|  |  |
| --- | --- |
| -V | 显示版本编号 |
| -h | 会显示版本编号及指令的使用方式说明 |
| -l | 显示出自己（执行 sudo 的使用者）的权限 |
| -v | 因为 sudo 在第一次执行时或是在 N 分钟内没有执行（N 预设为五）会问密码，这个参数是重新做一次确认，如果超过 N 分钟，也会问密码 |
| -k | 将会强迫使用者在下一次执行 sudo 时问密码（不论有没有超过 N 分钟） |
| -b | 将要执行的指令放在背景执行 |
| -p | prompt 可以更改问密码的提示语，其中 %u 会代换为使用者的帐号名称， %h 会显示主机名称 |
| -u | username/#uid 不加此参数，代表要以 root 的身份执行指令，而加了此参数，可以以 username 的身份执行指令（#uid 为该 username 的使用者号码） |
| -s | 执行环境变数中的 SHELL 所指定的 shell ，或是 /etc/passwd 里所指定的 shell |
| -H | 将环境变数中的 HOME （家目录）指定为要变更身份的使用者家目录（如不加 -u 参数就是[系统管理](http://baike.baidu.com/view/635537.htm)者 root ） |
| command | 要以[系统管理](http://baike.baidu.com/view/635537.htm)者身份（或以 -u 更改为其他人）执行的指令 |

### 范例

1. sudo -l 列出目前的权限

若用户不在sudoers中会提示不能运行sudo命令，存在则会显示详细的权限。

2. sudo -V 列出 sudo 的版本资讯

3. 指令名称:sudoers（在fc5下显示不能找到此命令，但用man可以查到其用法。）用来显示可以使用sudo的用户

## 6其他配置[编辑](http://baike.baidu.com/view/1138183.htm)

# User privilege specification

root ALL=(ALL) ALL

# Members of the admin group may gain root privileges

%admin ALL=(ALL) ALL

下面对以上配置做简要说明：

第一项配置的作用，是允许root用户使用sudo命令变成系统中任何其它类型的用户。第二个配置规定，管理组中的 所有成员都能以root的身份执行所有命令。因此，在默认安装的Ubuntu系统中，要想作为root身份来执行命令的话，只要在sudo后面跟上欲执行 的命令即可。

我们用一个实例来详细解释/etc/sudoers文件的配置语法，请看下面的例子：

jorge ALL=(root) /usr/bin/find, /bin/rm

上面的第一栏规定它的适用对象：用户或组，就本例来说，它是用户jorge。此外，因为系统中的组和用户可以重名，要想指定该规则的适用对象是组而非用户的话，组对象的名称一定要用[百分号](http://baike.baidu.com/view/1332267.htm)%开头。

第二栏指定该规则的适用[主机](http://baike.baidu.com/view/23880.htm)。当我们在多个系统之间部署sudo环境时，这一栏格外有用，这里的ALL代表所有[主机](http://baike.baidu.com/view/23880.htm)。但是，对于桌面系统或不想将sudo部署到多个系统的情况，这一栏就换成相应的[主机名](http://baike.baidu.com/view/1617349.htm)。

第三栏的值放在括号内，指出第一栏规定的用户能够以何种身份来执行命令。本例中该值设为root，这意味着用户jorge能够以root用户的身份来运行后面列出的命令。该值也可以设成[通配符](http://baike.baidu.com/view/451.htm)ALL，jorge便能作为系统中的任何用户来执行列出的命令了。

最后一栏（即/usr/bin/find,/bin/rm）是使用逗号分开的命令表，这些命令能被第一栏规定的用户以第三栏指出的身份来运行它们。本例中，该配置允许jorge作为[超级用户](http://baike.baidu.com/view/230823.htm)运行/usr/bin/find和 /bin/rm这两个命令。需要指出的是，这里列出的命令一定要使用绝对路径。

进一步：

我们可以利用这些规则为系统创建具体的角色。例如，要让一个组负责帐户管理，你一方面不想让这些用户具备完全的 root访问权限，另一方面还得让他们具有增加和删除用户的权利，那么我们可以在系统上创建一个名为accounts的组，然后把那些用户添加到这个组 里。之后，再使用visudo为/etc/sudoers添加下列内容： %accounts ALL=(root) /usr/sbin/useradd,/usr/sbin/userdel, /usr/sbin/[usermod](http://baike.baidu.com/view/1705625.htm)

现在好了，accounts组中的任何成员都能运行useradd、userdel和[usermod](http://baike.baidu.com/view/1705625.htm)命令了。如果过一段时间后，您发现该角色还需要其他工具，只要在该表的尾部将其添上就行了。这样真是方便极了！

需要注意的是，当我们为用户定义可以运行的命令时，必须使用完整的命令路径。这样做是完全出于安全的考虑，如果我们给出的命令只是简单的userad而非/usr/sbin/useradd，那么用户有可能创建一个他自己的[脚本](http://baike.baidu.com/view/54.htm)，也叫做useradd，然后放在它的本地路径中，如此一来他就能够通过这个名为useradd的本地[脚本](http://baike.baidu.com/view/54.htm)，作为root来执行任何他想要的命令了。这是相当危险的！

sudo命令的另一个便捷的功能，是它能够指出哪些命令在执行时不需要输入密码。这很有用，尤其是在非交互式脚本中以[超级用户](http://baike.baidu.com/view/230823.htm)的身份来运行某些命令的时候。例如，想要让用户作为[超级用户](http://baike.baidu.com/view/230823.htm)不 必输入密码就能执行kill命令，以便用户能立刻杀死一个失控的进程。为此，在命令行前边加上NOPASSWD:属性即可。例如，可以在/etc /sudoers文件中加上下面一行，从而让jorge获得这种权力： jorge ALL=(root)NOPASSWD: /bin/kill, /usr/bin/killall

这样一来，jorge就能运行以下命令，作为root用户来杀死失控的rm进程了。 jorge@ubuntu:~$ sudo killall rm

linux su和sudo命令的区别

来源：互联网 作者：佚名 时间：04-13 11:02:50 【大 中 小】

一. 使用 su 命令临时切换用户身份 1、su 的适用条件和威力 　　su命令就是切换用户的工具，怎么理解呢？比如我们以普通用户beinan登录的，但要添加用户任务，执行useradd ，beinan用户没有这个权限，而这个权限恰恰由root所拥有。解决办法无法有两个，一是退出beinan用户

一. 使用 su 命令临时切换用户身份

1、su 的适用条件和威力

　　su命令就是切换用户的工具，怎么理解呢？比如我们以普通用户beinan登录的，但要添加用户任务，执行useradd ，beinan用户没有这个权限，而这个权限恰恰由root所拥有。解决办法无法有两个，一是退出beinan用户，重新以root用户登录，但这种办法并不是最好的；二是我们没有必要退出beinan用户，可以用su来切换到root下进行添加用户的工作，等任务完成后再退出root。我们可以看到当然通过su 切换是一种比较好的办法；

　　通过su可以在用户之间切换，如果超级权限用户root向普通或虚拟用户切换不需要密码，什么是权力？这就是！而普通用户切换到其它任何用户都需要密码验证；

　　2、su 的用法：

　　su [OPTION选项参数] [用户]

　　-, -l, ——login 登录并改变到所切换的用户环境；

　　-c, ——commmand=COMMAND 执行一个命令，然后退出所切换到的用户环境；

　　至于更详细的，请参看man su ；

　　3、su 的范例：

　　su 在不加任何参数，默认为切换到root用户，但没有转到root用户家目录下，也就是说这时虽然是切换为root用户了，但并没有改变root登录环境；用户默认的登录环境，可以在/etc/passwd 中查得到，包括家目录，SHELL定义等；

　　[beinan@localhost ~]?$ suPassword:[root@localhost beinan]# pwd/home/beinan

　　su 加参数 - ，表示默认切换到root用户，并且改变到root用户的环境；

　　[beinan@localhost ~]?$ pwd/home/beinan[beinan@localhost ~]?$ su -Password:[root@localhost ~]# pwd/root

　　su 参数 - 用户名

　　[beinan@localhost ~]?$ su - root 注：这个和su - 是一样的功能；

　　Password:

　　[root@localhost ~]# pwd

　　/root

　　[beinan@localhost ~]?$ su - linuxsir 注：这是切换到 linuxsir用户

　　Password: 注：在这里输入密码；

　　[linuxsir@localhost ~]?$ pwd 注：查看用户当前所处的位置；

　　/home/linuxsir

　　[linuxsir@localhost ~]?$ id 注：查看用户的UID和GID信息，主要是看是否切换过来了；

　　uid=505(linuxsir) gid=502(linuxsir) groups=0(root),500(beinan),502(linuxsir)

　　[linuxsir@localhost ~]?$

　　[beinan@localhost ~]?$ su - -c ls 注：这是su的参数组合，表示切换到root用户，并且改变到root环境，然后列出root家目录的文件，然后退出root用户；

　　Password: 注：在这里输入root的密码；

　　anaconda-ks.cfg Desktop install.log install.log.syslog testgroup testgroupbeinan testgrouproot

　　[beinan@localhost ~]?$ pwd 注：查看当前用户所处的位置；

　　/home/beinan

　　[beinan@localhost ~]?$ id 注：查看当前用户信息；

　　uid=500(beinan) gid=500(beinan) groups=500(beinan)

　　4、su的优缺点；

　　su 的确为管理带来方便，通过切换到root下，能完成所有系统管理工具，只要把root的密码交给任何一个普通用户，他都能切换到root来完成所有的系统管理工作；但通过su切换到root后，也有不安全因素；比如系统有10个用户，而且都参与管理。如果这10个用户都涉及到超级权限的运用，做为管理员如果想让其它用户通过su来切换到超级权限的root，必须把root权限密码都告诉这10个用户；如果这10个用户都有root权限，通过root权限可以做任何事，这在一定程度上就对系统的安全造成了威协；想想Windows吧，简直就是恶梦；“没有不安全的系统，只有不安全的人”，我们绝对不能保证这 10个用户都能按正常操作流程来管理系统，其中任何一人对系统操作的重大失误，都可能导致系统崩溃或数据损失；所以su 工具在多人参与的系统管理中，并不是最好的选择，su只适用于一两个人参与管理的系统，毕竟su并不能让普通用户受限的使用；超级用户root密码应该掌握在少数用户手中，这绝对是真理！所以集权而治的存在还是有一定道理的；

二. sudo 授权许可使用的su，也是受限制的su

　　1. sudo 的适用条件

　　由于su 对切换到超级权限用户root后，权限的无限制性，所以su并不能担任多个管理员所管理的系统。如果用su 来切换到超级用户来管理系统，也不能明确哪些工作是由哪个管理员进行的操作。特别是对于服务器的管理有多人参与管理时，最好是针对每个管理员的技术特长和管理范围，并且有针对性的下放给权限，并且约定其使用哪些工具来完成与其相关的工作，这时我们就有必要用到 sudo。

　　通过sudo，我们能把某些超级权限有针对性的下放，并且不需要普通用户知道root密码，所以sudo 相对于权限无限制性的su来说，还是比较安全的，所以sudo 也能被称为受限制的su ；另外sudo 是需要授权许可的，所以也被称为授权许可的su；

　　sudo 执行命令的流程是当前用户切换到root（或其它指定切换到的用户），然后以root（或其它指定的切换到的用户）身份执行命令，执行完成后，直接退回到当前用户；而这些的前提是要通过sudo的配置文件/etc/sudoers来进行授权；

比如我们想用beinan普通用户通过more /etc/shadow文件的内容时，可能会出现下面的情况；

　　[beinan@localhost ~]?$ more /etc/shadow/etc/shadow: 权限不够

　　这时我们可以用sudo more /etc/shadow 来读取文件的内容；就就需要在/etc/soduers中给beinan授权

　　于是我们就可以先su 到root用户下通过visudo 来改/etc/sudoers ；（比如我们是以beinan用户登录系统的）

　　[beinan@localhost ~]?$ su

　　Password: 注：在这里输入root密码

　　下面运行visodu；

　　[root@localhost beinan]# visudo 注：运行visudo 来改 /etc/sudoers

　　加入如下一行，退出保存；退出保存，在这里要会用vi，visudo也是用的vi编辑器；至于vi的用法不多说了；beinan ALL=/bin/more 表示beinan可以切换到root下执行more 来查看文件；

　　退回到beinan用户下，用exit命令；

　　[root@localhost beinan]# exit

　　exit

　　[beinan@localhost ~]?$

　　查看beinan的通过sudo能执行哪些命令？

　　[beinan@localhost ~]?$ sudo -l

　　Password: 注：在这里输入beinan用户的密码

　　User beinan may run the following commands on this host: 注：在这里清晰的说明在本台主机上，beinan用户可以以root权限运行more ；在root权限下的more ，可以查看任何文本文件的内容的；

　　(root) /bin/more

　　最后，我们看看是不是beinan用户有能力看到/etc/shadow文件的内容；

　　[beinan@localhost ~]?$ sudo more /etc/shadow

　　beinan 不但能看到 /etc/shadow文件的内容，还能看到只有root权限下才能看到的其它文件的内容，比如；

　　[beinan@localhost ~]?$ sudo more /etc/gshadow

　　对于beinan用户查看和读取所有系统文件中，我只想把/etc/shadow 的内容可以让他查看；可以加入下面的一行；

　　beinan ALL=/bin/more /etc/shadow

　　题外话：有的弟兄会说，我通过su 切换到root用户就能看到所有想看的内容了，哈哈，对啊。但咱们现在不是在讲述sudo的用法吗？如果主机上有多个用户并且不知道root用户的密码，但又想查看某些他们看不到的文件，这时就需要管理员授权了；这就是sudo的好处；

　　实例五：练习用户组在/etc/sudoers中写法；

　　如果用户组出现在/etc/sudoers 中，前面要加%号，比如%beinan ，中间不能有空格；%beinan ALL=/usr/sbin/\*,/sbin/\*

　　如果我们在 /etc/sudoers 中加上如上一行，表示beinan用户组下的所有成员，在所有可能的出现的主机名下，都能切换到root用户下运行 /usr/sbin和/sbin目录下的所有命令；

　　实例六：练习取消某类程序的执行：

　　取消程序某类程序的执行，要在命令动作前面加上!号； 在本例中也出现了通配符的\*的用法；

　　beinan ALL=/usr/sbin/\*,/sbin/\*,!/usr/sbin/fdisk 注：把这行规则加入到/etc/sudoers中；但您得有beinan这个用户组，并且beinan也是这个组中的才行；

　　本规则表示beinan用户在所有可能存在的主机名的主机上运行/usr/sbin和/sbin下所有的程序，但fdisk 程序除外；

　　[beinan@localhost ~]?$ sudo -l

　　Password: 注：在这里输入beinan用户的密码；

　　User beinan may run the following commands on this host:(root) /usr/sbin/\*(root) /sbin/\*(root) !/sbin/fdisk[beinan@localhost ~]?$ sudo /sbin/fdisk -lSorry, user beinan is not allowed to execute '/sbin/fdisk -l' as root on localhost.

　　注：不能切换到root用户下运行fdisk 程序；

如果有sudo 的权限而没有su的权限: sudo su;

7、traceroute netstat route

8、fdisk mkfs fstab

9、find locate

10、mkdir rmdir touch cp mv rm

11、more less tail head

12、useradd userdel groupadd

13 mount

14

# 三、ssh公钥登陆

## 3.1、简介

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。利用 SSH 协议可以有效防止远程管理过程中的信息泄露问题。SSH最初是UNIX系统上的一个程序，后来又迅速扩展到其他操作平台。SSH在正确使用时可弥补网络中的漏洞。SSH客户端适用于多种平台。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。

传统的网络服务程序，如：ftp、pop和telnet在本质上都是不安全的，因为它们在网络上用明文传送口令和数据，别有用心的人非常容易就可以截获这些口令和数据。而且，这些服务程序的安全验证方式也是有其弱点的， 就是很容易受到“中间人”（man-in-the-middle）这种方式的攻击。所谓“中间人”的攻击方式， 就是“中间人”冒充真正的服务器接收你传给服务器的数据，然后再冒充你把数据传给真正的服务器。服务器和你之间的数据传送被“中间人”一转手做了手脚之后，就会出现很严重的问题。通过使用SSH，你可以把所有传输的数据进行加密，这样"中间人"这种攻击方式就不可能实现了，而且也能够防止DNS欺骗和IP欺骗。使用SSH，还有一个额外的好处就是传输的数据是经过压缩的，所以可以加快传输的速度。SSH有很多功能，它既可以代替Telnet，又可以为FTP、PoP、甚至为PPP提供一个安全的"通道"。

## 3.2、验证

从客户端来看，SSH提供两种级别的安全验证。

第一种级别（基于口令的安全验证）

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

第二种级别（基于密匙的安全验证）

需要依靠密匙，也就是你必须为自己创建一对密匙，并把公用密匙放在需要访问的服务器上。如果你要连接到SSH服务器上，客户端软件就会向服务器发出请求，请求用你的密匙进行安全验证。服务器收到请求之后，先在该服务器上你的主目录下寻找你的公用密匙，然后把它和你发送过来的公用密匙进行比较。如果两个密匙一致，服务器就用公用密匙加密“质询”（challenge）并把它发送给客户端软件。客户端软件收到“质询”之后就可以用你的私人密匙解密再把它发送给服务器。

用这种方式，你必须知道自己密匙的口令。但是，与第一种级别相比，第二种级别不需要在网络上传送口令。

第二种级别不仅加密所有传送的数据，而且“中间人”这种攻击方式也是不可能的（因为他没有你的私人密匙）。但是整个登录的过程可能需要10秒。

## 3.3、使用ssh公钥实现免密码登录

ssh 无密码登录要使用公钥与私钥。linux下可以用用ssh-keygen生成公钥/私钥对，下面我以CentOS为例。

有机器A(192.168.1.155)，B(192.168.1.181)。

现想A通过ssh免密码登录到B。

首先以root账户登陆为例。

1.在A机下生成公钥/私钥对。

[root@A ~]# ssh-keygen -t rsa -P ''

-P表示密码，-P '' 就表示空密码，也可以不用-P参数，这样就要三车回车，用-P就一次回车。

该命令将在/root/.ssh目录下面产生一对密钥id\_rsa和id\_rsa.pub。

一般采用的ssh的rsa密钥，也就是rsa2类型的加密方式。

id\_rsa 私钥

id\_rsa.pub 公钥

下述命令产生不同类型的密钥

ssh-keygen -t dsa

ssh-keygen -t rsa

ssh-keygen -t rsa1

2.复制公钥到目标机B

把A机下的/root/.ssh/id\_rsa.pub

复制到B机的 /root/.ssh/authorized\_keys文件里，

先要在B机上创建好 /root/.ssh 这个目录，用scp复制。

[root@A ~]# scp /root/.ssh/id\_rsa.pub root@192.168.1.181:/root/.ssh/authorized\_keys

root@192.168.1.181's password:

id\_rsa.pub 100% 223 0.2KB/s 00:00

由于还没有免密码登录的，所以要输入一次B机的root密码。

3.authorized\_keys的权限要是600!!!

[root@B ~]# chmod 600 /root/.ssh/authorized\_keys

4.A机登录B机。

[root@A ~]# ssh -l root 192.168.1.181

The authenticity of host '192.168.1.181 (192.168.1.181)' can't be established.

RSA key fingerprint is 00:a6:a8:87:eb:c7:40:10:39:cc:a0:eb:50:d9:6a:5b.

Are you sure you want to continue connecting (yes/no)? yes

Warning: Permanently added '192.168.1.181' (RSA) to the list of known hosts.

Last login: Thu Jul 3 09:53:18 2008 from root

[root@B ~]#

第一次登录是时要你输入yes。

现在A机可以无密码登录B机了。

小结：登录的机子可有私钥，被登录的机子要有登录机子的公钥。这个公钥/私钥对一般在私钥宿主机产生。上面是用rsa算法的公钥/私钥对，当然也可以用dsa(对应的文件是id\_dsa，id\_dsa.pub)

想让A，B机无密码互登录，那B机以上面同样的方式配置即可。

另外，这里用的文件名，是可以改动的，包括ssh的配置，因为，光靠一对秘钥显然是不安全的，强大的ssh还有更加牛逼的配置，比如ssh协议版本，这些配置，都在/etc/ssh/sshd\_config,包括协议版本，秘钥路径，验证方式等等，这个后边会详细介绍。

SSH-KeyGen 的用法

假设 A 为客户机器，B为目标机；

要达到的目的：

A机器ssh登录B机器无需输入密码；

加密方式选 rsa|dsa均可以，默认dsa

做法：

1、登录A机器

2、ssh-keygen -t [rsa|dsa]，将会生成密钥文件和私钥文件 id\_rsa,id\_rsa.pub或id\_dsa,id\_dsa.pub

3、将 .pub 文件复制到B机器的 .ssh 目录， 并 cat id\_dsa.pub >> ~/.ssh/authorized\_keys

4、大功告成，从A机器登录B机器的目标账户，不再需要密码了；

ssh-keygen做密码验证可以使在向对方机器上ssh ,scp不用使用密码.

具体方法如下:

ssh-keygen -t rsa

然后全部回车,采用默认值.

这样生成了一对密钥，存放在用户目录的~/.ssh下。

将公钥考到对方机器的用户目录下，并拷到~/.ssh/authorized\_keys中。

要保证.ssh和authorized\_keys都只有用户自己有写权限。否则验证无效。（今天就是遇到这个问题，找了好久问题所在），其实仔细想想，这样做是为了不会出现系统漏洞。

## 3.4、sshd\_config详细配置

vi /etc/ssh/sshd\_config

# 1. 关于 SSH Server 的整体设定，包含使用的 port 啦，以及使用的密码演算方式

Port 22　　　　　　　　　　# SSH 预设使用 22 这个 port，您也可以使用多的 port ！

　　　　　　　　　　　　　 # 亦即重复使用 port 这个设定项目即可！

Protocol 2,1　　　　　　　 # 选择的 SSH 协议版本，可以是 1 也可以是 2 ，

　　　　　　　　　　　　　 # 如果要同时支持两者，就必须要使用 2,1 这个分隔了！

#ListenAddress 0.0.0.0　　 # 监听的主机适配卡！举个例子来说，如果您有两个 IP，

　　　　　　　　　　　　　 # 分别是 192.168.0.100 及 192.168.2.20 ，那么只想要

　　　　　　　　　　　　　 # 开放 192.168.0.100 时，就可以写如同下面的样式：

ListenAddress 192.168.0.100 # 只监听来自 192.168.0.100 这个 IP 的SSH联机。

　　　　　　　　　　　　　　　　　　 # 如果不使用设定的话，则预设所有接口均接受 SSH

PidFile /var/run/sshd.pid　　　　　　# 可以放置 SSHD 这个 PID 的档案！左列为默认值

LoginGraceTime 600　　　　 # 当使用者连上 SSH server 之后，会出现输入密码的画面，

　　　　　　　　　　　　　 # 在该画面中，在多久时间内没有成功连上 SSH server ，

　　　　　　　　　　　　　 # 就断线！时间为秒！

Compression yes　　　　　　# 是否可以使用压缩指令？当然可以啰！

# 2. 说明主机的 Private Key 放置的档案，预设使用下面的档案即可！

HostKey /etc/ssh/ssh\_host\_key　　　　# SSH version 1 使用的私钥

HostKey /etc/ssh/ssh\_host\_rsa\_key　　# SSH version 2 使用的 RSA 私钥

HostKey /etc/ssh/ssh\_host\_dsa\_key　　# SSH version 2 使用的 DSA 私钥

# 2.1 关于 version 1 的一些设定！

KeyRegenerationInterval 3600　 　　　# 由前面联机的说明可以知道， version 1 会使用

　　　　　　　　　　　　　　　　　　 # server 的 Public Key ，那么如果这个 Public

　　　　　　　　　　　　　　　　　　 # Key 被偷的话，岂不完蛋？所以需要每隔一段时间

　　　　　　　　　　　　　　　　　　 # 来重新建立一次！这里的时间为秒！

ServerKeyBits 768 　　　　　　　　　 # 没错！这个就是 Server key 的长度！

# 3. 关于登录文件的讯息数据放置与 daemon 的名称！

SyslogFacility AUTH　　　　　　　　　# 当有人使用 SSH 登入系统的时候，SSH会记录资

　　　　　　　　　　　　　　　　　　 # 讯，这个信息要记录在什么 daemon name 底下？

　　　　　　　　　　　　　　　　　　 # 预设是以 AUTH 来设定的，即是 /var/log/secure

　　　　　　　　　　　　　　　　　　 # 里面！什么？忘记了！回到 Linux 基础去翻一下

　　　　　　　　　　　　　　　　　　 # 其它可用的 daemon name 为：DAEMON,USER,AUTH,

　　　　　　　　　　　　　　　　　　 # LOCAL0,LOCAL1,LOCAL2,LOCAL3,LOCAL4,LOCAL5,

LogLevel INFO　　　　　　　　　　　　# 登录记录的等级！嘿嘿！任何讯息！

　　　　　　　　　　　　　　　　　　 # 同样的，忘记了就回去参考！

# 4. 安全设定项目！极重要！

# 4.1 登入设定部分

PermitRootLogin no　　 　　# 是否允许 root 登入！预设是允许的，但是建议设定成 no！

UserLogin no　　　　　　　 # 在 SSH 底下本来就不接受 login 这个程序的登入！

StrictModes yes　　　　　　# 当使用者的 host key 改变之后，Server 就不接受联机，

　　　　　　　　　　　　　 # 可以抵挡部分的木马程序！

#RSAAuthentication yes　　 # 是否使用纯的 RSA 认证！？仅针对 version 1 ！

PubkeyAuthentication yes　 # 是否允许 Public Key ？当然允许啦！只有 version 2

AuthorizedKeysFile .ssh/authorized\_keys

　　　　　　　　　　　　　 # 上面这个在设定若要使用不需要密码登入的账号时，那么那个

　　　　　　　　　　　　　 # 账号的存放档案所在档名！

# 4.2 认证部分

RhostsAuthentication no　　# 本机系统不止使用 .rhosts ，因为仅使用 .rhosts 太

　　　　　　　　　　　　　 # 不安全了，所以这里一定要设定为 no ！

IgnoreRhosts yes　　　　　 # 是否取消使用 ~/.ssh/.rhosts 来做为认证！当然是！

RhostsRSAAuthentication no # 这个选项是专门给 version 1 用的，使用 rhosts 档案在

　　　　　　　　　　　　　 # /etc/hosts.equiv配合 RSA 演算方式来进行认证！不要使用

HostbasedAuthentication no # 这个项目与上面的项目类似，不过是给 version 2 使用的！

IgnoreUserKnownHosts no　　# 是否忽略家目录内的 ~/.ssh/known\_hosts 这个档案所记录

　　　　　　　　　　　　　 # 的主机内容？当然不要忽略，所以这里就是 no 啦！

PasswordAuthentication yes # 密码验证当然是需要的！所以这里写 yes 啰！

PermitEmptyPasswords no　　# 若上面那一项如果设定为 yes 的话，这一项就最好设定

　　　　　　　　　　　　　 # 为 no ，这个项目在是否允许以空的密码登入！当然不许！

ChallengeResponseAuthentication yes # 挑战任何的密码认证！所以，任何 login.conf

　　　　　　　　　　　　　　　　　　 # 规定的认证方式，均可适用！

#PAMAuthenticationViaKbdInt yes # 是否启用其它的 PAM 模块！启用这个模块将会

　　　　　　　　　　　　　　　　　　 # 导致 PasswordAuthentication 设定失效！

# 4.3 与 Kerberos 有关的参数设定！因为我们没有 Kerberos 主机，所以底下不用设定！

#KerberosAuthentication no

#KerberosOrLocalPasswd yes

#KerberosTicketCleanup yes

#KerberosTgtPassing no

# 4.4 底下是有关在 X-Window 底下使用的相关设定！

X11Forwarding yes

#X11DisplayOffset 10

#X11UseLocalhost yes

# 4.5 登入后的项目：

PrintMotd no # 登入后是否显示出一些信息呢？例如上次登入的时间、地点等

　　　　　　　　　　　　　# 等，预设是 yes ，但是，如果为了安全，可以考虑改为 no ！

PrintLastLog yes　　　　　# 显示上次登入的信息！可以啊！预设也是 yes ！

KeepAlive yes　　　　　　 # 一般而言，如果设定这项目的话，那么 SSH Server 会传送

　　　　　　　　　　　　　# KeepAlive 的讯息给 Client 端，以确保两者的联机正常！

　　　　　　　　　　　　　# 在这个情况下，任何一端死掉后， SSH 可以立刻知道！而不会

　　　　　　　　　　　　　# 有僵尸程序的发生！

UsePrivilegeSeparation yes # 使用者的权限设定项目！就设定为 yes 吧！

MaxStartups 10　　　　　　# 同时允许几个尚未登入的联机画面？当我们连上 SSH ，

　　　　　　　　　　　　　# 但是尚未输入密码时，这个时候就是我们所谓的联机画面啦！

　　　　　　　　　　　　　# 在这个联机画面中，为了保护主机，所以需要设定最大值，

　　　　　　　　　　　　　# 预设最多十个联机画面，而已经建立联机的不计算在这十个当中

# 4.6 关于使用者抵挡的设定项目：

DenyUsers \*　　　　　　　 # 设定受抵挡的使用者名称，如果是全部的使用者，那就是全部

　　　　　　　　　　　　　# 挡吧！若是部分使用者，可以将该账号填入！例如下列！

DenyUsers test

DenyGroups test　　　　　 # 与 DenyUsers 相同！仅抵挡几个群组而已！

# 5. 关于 SFTP 服务的设定项目！

Subsystem sftp /usr/lib/ssh/sftp-server

## 3.5、更加详细的配置，并且翻译较好

sshd\_config配置详解

名称

sshd\_config - OpenSSH SSH 服务器守护进程配置文件

大纲

/etc/ssh/sshd\_config

描述

sshd(8) 默认从 /etc/ssh/sshd\_config 文件(或通过 -f 命令行选项指定的文件)读取配置信息。

配置文件是由"指令 值"对组成的，每行一个。空行和以'#'开头的行都将被忽略。

如果值中含有空白符或者其他特殊符号，那么可以通过在两边加上双引号(")进行界定。

[注意]值是大小写敏感的，但指令是大小写无关的。

当前所有可以使用的配置指令如下：

AcceptEnv

指定客户端发送的哪些环境变量将会被传递到会话环境中。[注意]只有SSH-2协议支持环境变量的传递。

细节可以参考 ssh\_config(5) 中的 SendEnv 配置指令。

指令的值是空格分隔的变量名列表(其中可以使用'\*'和'?'作为通配符)。也可以使用多个 AcceptEnv 达到同样的目的。

需要注意的是，有些环境变量可能会被用于绕过禁止用户使用的环境变量。由于这个原因，该指令应当小心使用。

默认是不传递任何环境变量。

AddressFamily

指定 sshd(8) 应当使用哪种地址族。取值范围是："any"(默认)、"inet"(仅IPv4)、"inet6"(仅IPv6)。

AllowGroups

这个指令后面跟着一串用空格分隔的组名列表(其中可以使用"\*"和"?"通配符)。默认允许所有组登录。

如果使用了这个指令，那么将仅允许这些组中的成员登录，而拒绝其它所有组。

这里的"组"是指"主组"(primary group)，也就是/etc/passwd文件中指定的组。

这里只允许使用组的名字而不允许使用GID。相关的 allow/deny 指令按照下列顺序处理：

DenyUsers, AllowUsers, DenyGroups, AllowGroups

AllowTcpForwarding

是否允许TCP转发，默认值为"yes"。

禁止TCP转发并不能增强安全性，除非禁止了用户对shell的访问，因为用户可以安装他们自己的转发器。

AllowUsers

这个指令后面跟着一串用空格分隔的用户名列表(其中可以使用"\*"和"?"通配符)。默认允许所有用户登录。

如果使用了这个指令，那么将仅允许这些用户登录，而拒绝其它所有用户。

如果指定了 USER@HOST 模式的用户，那么 USER 和 HOST 将同时被检查。

这里只允许使用用户的名字而不允许使用UID。相关的 allow/deny 指令按照下列顺序处理：

DenyUsers, AllowUsers, DenyGroups, AllowGroups

AuthorizedKeysFile

存放该用户可以用来登录的 RSA/DSA 公钥。

该指令中可以使用下列根据连接时的实际情况进行展开的符号：

%% 表示'%'、%h 表示用户的主目录、%u 表示该用户的用户名。

经过扩展之后的值必须要么是绝对路径，要么是相对于用户主目录的相对路径。

默认值是".ssh/authorized\_keys"。

Banner

将这个指令指定的文件中的内容在用户进行认证前显示给远程用户。

这个特性仅能用于SSH-2，默认什么内容也不显示。"none"表示禁用这个特性。

ChallengeResponseAuthentication

是否允许质疑-应答(challenge-response)认证。默认值是"yes"。

所有 login.conf(5) 中允许的认证方式都被支持。

Ciphers

指定SSH-2允许使用的加密算法。多个算法之间使用逗号分隔。可以使用的算法如下：

"aes128-cbc", "aes192-cbc", "aes256-cbc", "aes128-ctr", "aes192-ctr", "aes256-ctr",

"3des-cbc", "arcfour128", "arcfour256", "arcfour", "blowfish-cbc", "cast128-cbc"

默认值是可以使用上述所有算法。

ClientAliveCountMax

sshd(8) 在未收到任何客户端回应前最多允许发送多少个"alive"消息。默认值是 3 。

到达这个上限后，sshd(8) 将强制断开连接、关闭会话。

需要注意的是，"alive"消息与 TCPKeepAlive 有很大差异。

"alive"消息是通过加密连接发送的，因此不会被欺骗；而 TCPKeepAlive 却是可以被欺骗的。

如果 ClientAliveInterval 被设为 15 并且将 ClientAliveCountMax 保持为默认值，

那么无应答的客户端大约会在45秒后被强制断开。这个指令仅可以用于SSH-2协议。

ClientAliveInterval

设置一个以秒记的时长，如果超过这么长时间没有收到客户端的任何数据，

sshd(8) 将通过安全通道向客户端发送一个"alive"消息，并等候应答。

默认值 0 表示不发送"alive"消息。这个选项仅对SSH-2有效。

Compression

是否对通信数据进行加密，还是延迟到认证成功之后再对通信数据加密。

可用值："yes", "delayed"(默认), "no"。

DenyGroups

这个指令后面跟着一串用空格分隔的组名列表(其中可以使用"\*"和"?"通配符)。默认允许所有组登录。

如果使用了这个指令，那么这些组中的成员将被拒绝登录。

这里的"组"是指"主组"(primary group)，也就是/etc/passwd文件中指定的组。

这里只允许使用组的名字而不允许使用GID。相关的 allow/deny 指令按照下列顺序处理：

DenyUsers, AllowUsers, DenyGroups, AllowGroups

DenyUsers

这个指令后面跟着一串用空格分隔的用户名列表(其中可以使用"\*"和"?"通配符)。默认允许所有用户登录。

如果使用了这个指令，那么这些用户将被拒绝登录。

如果指定了 USER@HOST 模式的用户，那么 USER 和 HOST 将同时被检查。

这里只允许使用用户的名字而不允许使用UID。相关的 allow/deny 指令按照下列顺序处理：

DenyUsers, AllowUsers, DenyGroups, AllowGroups

ForceCommand

强制执行这里指定的命令而忽略客户端提供的任何命令。这个命令将使用用户的登录shell执行(shell -c)。

这可以应用于 shell 、命令、子系统的完成，通常用于 Match 块中。

这个命令最初是在客户端通过 SSH\_ORIGINAL\_COMMAND 环境变量来支持的。

GatewayPorts

是否允许远程主机连接本地的转发端口。默认值是"no"。

sshd(8) 默认将远程端口转发绑定到loopback地址。这样将阻止其它远程主机连接到转发端口。

GatewayPorts 指令可以让 sshd 将远程端口转发绑定到非loopback地址，这样就可以允许远程主机连接了。

"no"表示仅允许本地连接，"yes"表示强制将远程端口转发绑定到统配地址(wildcard address)，

"clientspecified"表示允许客户端选择将远程端口转发绑定到哪个地址。

GSSAPIAuthentication

是否允许使用基于 GSSAPI 的用户认证。默认值为"no"。仅用于SSH-2。

GSSAPICleanupCredentials

是否在用户退出登录后自动销毁用户凭证缓存。默认值是"yes"。仅用于SSH-2。

HostbasedAuthentication

这个指令与 RhostsRSAAuthentication 类似，但是仅可以用于SSH-2。推荐使用默认值"no"。

推荐使用默认值"no"禁止这种不安全的认证方式。

HostbasedUsesNameFromPacketOnly

在开启 HostbasedAuthentication 的情况下，

指定服务器在使用 ~/.shosts ~/.rhosts /etc/hosts.equiv 进行远程主机名匹配时，是否进行反向域名查询。

"yes"表示 sshd(8) 信任客户端提供的主机名而不进行反向查询。默认值是"no"。

HostKey

主机私钥文件的位置。如果权限不对，sshd(8) 可能会拒绝启动。

SSH-1默认是 /etc/ssh/ssh\_host\_key 。

SSH-2默认是 /etc/ssh/ssh\_host\_rsa\_key 和 /etc/ssh/ssh\_host\_dsa\_key 。

一台主机可以拥有多个不同的私钥。"rsa1"仅用于SSH-1，"dsa"和"rsa"仅用于SSH-2。

IgnoreRhosts

是否在 RhostsRSAAuthentication 或 HostbasedAuthentication 过程中忽略 .rhosts 和 .shosts 文件。

不过 /etc/hosts.equiv 和 /etc/shosts.equiv 仍将被使用。推荐设为默认值"yes"。

IgnoreUserKnownHosts

是否在 RhostsRSAAuthentication 或 HostbasedAuthentication 过程中忽略用户的 ~/.ssh/known\_hosts 文件。

默认值是"no"。为了提高安全性，可以设为"yes"。

KerberosAuthentication

是否要求用户为 PasswordAuthentication 提供的密码必须通过 Kerberos KDC 认证，也就是是否使用Kerberos认证。

要使用Kerberos认证，服务器需要一个可以校验 KDC identity 的 Kerberos servtab 。默认值是"no"。

KerberosGetAFSToken

如果使用了 AFS 并且该用户有一个 Kerberos 5 TGT，那么开启该指令后，

将会在访问用户的家目录前尝试获取一个 AFS token 。默认为"no"。

KerberosOrLocalPasswd

如果 Kerberos 密码认证失败，那么该密码还将要通过其它的认证机制(比如 /etc/passwd)。

默认值为"yes"。

KerberosTicketCleanup

是否在用户退出登录后自动销毁用户的 ticket 。默认值是"yes"。

KeyRegenerationInterval

在SSH-1协议下，短命的服务器密钥将以此指令设置的时间为周期(秒)，不断重新生成。

这个机制可以尽量减小密钥丢失或者黑客攻击造成的损失。

设为 0 表示永不重新生成，默认为 3600(秒)。

ListenAddress

指定 sshd(8) 监听的网络地址，默认监听所有地址。可以使用下面的格式：

ListenAddress host|IPv4\_addr|IPv6\_addr

ListenAddress host|IPv4\_addr:port

ListenAddress [host|IPv6\_addr]:port

如果未指定 port ，那么将使用 Port 指令的值。

可以使用多个 ListenAddress 指令监听多个地址。

LoginGraceTime

限制用户必须在指定的时限内认证成功，0 表示无限制。默认值是 120 秒。

LogLevel

指定 sshd(8) 的日志等级(详细程度)。可用值如下：

QUIET, FATAL, ERROR, INFO(默认), VERBOSE, DEBUG, DEBUG1, DEBUG2, DEBUG3

DEBUG 与 DEBUG1 等价；DEBUG2 和 DEBUG3 则分别指定了更详细、更罗嗦的日志输出。

比 DEBUG 更详细的日志可能会泄漏用户的敏感信息，因此反对使用。

MACs

指定允许在SSH-2中使用哪些消息摘要算法来进行数据校验。

可以使用逗号分隔的列表来指定允许使用多个算法。默认值(包含所有可以使用的算法)是：

hmac-md5,hmac-sha1,umac-64@openssh.com,hmac-ripemd160,hmac-sha1-96,hmac-md5-96

Match

引入一个条件块。块的结尾标志是另一个 Match 指令或者文件结尾。

如果 Match 行上指定的条件都满足，那么随后的指令将覆盖全局配置中的指令。

Match 的值是一个或多个"条件-模式"对。可用的"条件"是：User, Group, Host, Address 。

只有下列指令可以在 Match 块中使用：AllowTcpForwarding, Banner,

ForceCommand, GatewayPorts, GSSApiAuthentication,

KbdInteractiveAuthentication, KerberosAuthentication,

PasswordAuthentication, PermitOpen, PermitRootLogin,

RhostsRSAAuthentication, RSAAuthentication, X11DisplayOffset,

X11Forwarding, X11UseLocalHost

MaxAuthTries

指定每个连接最大允许的认证次数。默认值是 6 。

如果失败认证的次数超过这个数值的一半，连接将被强制断开，且会生成额外的失败日志消息。

MaxStartups

最大允许保持多少个未认证的连接。默认值是 10 。

到达限制后，将不再接受新连接，除非先前的连接认证成功或超出 LoginGraceTime 的限制。

PasswordAuthentication

是否允许使用基于密码的认证。默认为"yes"。

PermitEmptyPasswords

是否允许密码为空的用户远程登录。默认为"no"。

PermitOpen

指定TCP端口转发允许的目的地，可以使用空格分隔多个转发目标。默认允许所有转发请求。

合法的指令格式如下：

PermitOpen host:port

PermitOpen IPv4\_addr:port

PermitOpen [IPv6\_addr]:port

"any"可以用于移除所有限制并允许一切转发请求。

PermitRootLogin

是否允许 root 登录。可用值如下：

"yes"(默认) 表示允许。"no"表示禁止。

"without-password"表示禁止使用密码认证登录。

"forced-commands-only"表示只有在指定了 command 选项的情况下才允许使用公钥认证登录。

同时其它认证方法全部被禁止。这个值常用于做远程备份之类的事情。

PermitTunnel

是否允许 tun(4) 设备转发。可用值如下：

"yes", "point-to-point"(layer 3), "ethernet"(layer 2), "no"(默认)。

"yes"同时蕴含着"point-to-point"和"ethernet"。

PermitUserEnvironment

指定是否允许 sshd(8) 处理 ~/.ssh/environment 以及 ~/.ssh/authorized\_keys 中的 environment= 选项。

默认值是"no"。如果设为"yes"可能会导致用户有机会使用某些机制(比如 LD\_PRELOAD)绕过访问控制，造成安全漏洞。

PidFile

指定在哪个文件中存放SSH守护进程的进程号，默认为 /var/run/sshd.pid 文件。

Port

指定 sshd(8) 守护进程监听的端口号，默认为 22 。可以使用多条指令监听多个端口。

默认将在本机的所有网络接口上监听，但是可以通过 ListenAddress 指定只在某个特定的接口上监听。

PrintLastLog

指定 sshd(8) 是否在每一次交互式登录时打印最后一位用户的登录时间。默认值是"yes"。

PrintMotd

指定 sshd(8) 是否在每一次交互式登录时打印 /etc/motd 文件的内容。默认值是"yes"。

Protocol

指定 sshd(8) 支持的SSH协议的版本号。

'1'和'2'表示仅仅支持SSH-1和SSH-2协议。"2,1"表示同时支持SSH-1和SSH-2协议。

PubkeyAuthentication

是否允许公钥认证。仅可以用于SSH-2。默认值为"yes"。

RhostsRSAAuthentication

是否使用强可信主机认证(通过检查远程主机名和关联的用户名进行认证)。仅用于SSH-1。

这是通过在RSA认证成功后再检查 ~/.rhosts 或 /etc/hosts.equiv 进行认证的。

出于安全考虑，建议使用默认值"no"。

RSAAuthentication

是否允许使用纯 RSA 公钥认证。仅用于SSH-1。默认值是"yes"。

ServerKeyBits

指定临时服务器密钥的长度。仅用于SSH-1。默认值是 768(位)。最小值是 512 。

StrictModes

指定是否要求 sshd(8) 在接受连接请求前对用户主目录和相关的配置文件进行宿主和权限检查。

强烈建议使用默认值"yes"来预防可能出现的低级错误。

Subsystem

配置一个外部子系统(例如，一个文件传输守护进程)。仅用于SSH-2协议。

值是一个子系统的名字和对应的命令行(含选项和参数)。比如"sft /bin/sftp-server"。

SyslogFacility

指定 sshd(8) 将日志消息通过哪个日志子系统(facility)发送。有效值是：

DAEMON, USER, AUTH(默认), LOCAL0, LOCAL1, LOCAL2, LOCAL3, LOCAL4, LOCAL5, LOCAL6, LOCAL7

TCPKeepAlive

指定系统是否向客户端发送 TCP keepalive 消息。默认值是"yes"。

这种消息可以检测到死连接、连接不当关闭、客户端崩溃等异常。

可以设为"no"关闭这个特性。

UseDNS

指定 sshd(8) 是否应该对远程主机名进行反向解析，以检查此主机名是否与其IP地址真实对应。默认值为"yes"。

UseLogin

是否在交互式会话的登录过程中使用 login(1) 。默认值是"no"。

如果开启此指令，那么 X11Forwarding 将会被禁止，因为 login(1) 不知道如何处理 xauth(1) cookies 。

需要注意的是，login(1) 是禁止用于远程执行命令的。

如果指定了 UsePrivilegeSeparation ，那么它将在认证完成后被禁用。

UsePrivilegeSeparation

是否让 sshd(8) 通过创建非特权子进程处理接入请求的方法来进行权限分离。默认值是"yes"。

认证成功后，将以该认证用户的身份创建另一个子进程。

这样做的目的是为了防止通过有缺陷的子进程提升权限，从而使系统更加安全。

X11DisplayOffset

指定 sshd(8) X11 转发的第一个可用的显示区(display)数字。默认值是 10 。

这个可以用于防止 sshd 占用了真实的 X11 服务器显示区，从而发生混淆。

X11Forwarding

是否允许进行 X11 转发。默认值是"no"，设为"yes"表示允许。

如果允许X11转发并且sshd(8)代理的显示区被配置为在含有通配符的地址(X11UseLocalhost)上监听。

那么将可能有额外的信息被泄漏。由于使用X11转发的可能带来的风险，此指令默认值为"no"。

需要注意的是，禁止X11转发并不能禁止用户转发X11通信，因为用户可以安装他们自己的转发器。

如果启用了 UseLogin ，那么X11转发将被自动禁止。

X11UseLocalhost

sshd(8) 是否应当将X11转发服务器绑定到本地loopback地址。默认值是"yes"。

sshd 默认将转发服务器绑定到本地loopback地址并将 DISPLAY 环境变量的主机名部分设为"localhost"。

这可以防止远程主机连接到 proxy display 。不过某些老旧的X11客户端不能在此配置下正常工作。

为了兼容这些老旧的X11客户端，你可以设为"no"。

XAuthLocation

指定 xauth(1) 程序的绝对路径。默认值是 /usr/X11R6/bin/xauth

时间格式

在 sshd(8) 命令行参数和配置文件中使用的时间值可以通过下面的格式指定：time[qualifier] 。

其中的 time 是一个正整数，而 qualifier 可以是下列单位之一：

<无> 秒

s | S 秒

m | M 分钟

h | H 小时

d | D 天

w | W 星期

可以通过指定多个数值来累加时间，比如：

1h30m 1 小时 30 分钟 (90 分钟)

文件

/etc/ssh/sshd\_config

sshd(8) 的主配置文件。这个文件的宿主应当是root，权限最大可以是"644"。

## 3.6、简明配置 ssh\_config和sshd\_config的比较

这两个文件都是在/etc/ssh/下的，我们看看他们有什么区别：

配置“/etc/ssh/ssh\_config”文件

“/etc/ssh/ssh\_config” 文件是OpenSSH系统范围的配置文件，允许你通过设置不同的选项来改变客户端程序的运行方式。这个文件的每一行包含“关键词－值”的匹配，其中“关键词”是忽略大小写的。下面列出来的是最重要的关键词，用man命令查看帮助页（ssh (1)）可以得到详细的列表。

编辑“ssh\_config”文件（vi /etc/ssh/ssh\_config），添加或改变下面的参数：

# Site-wide defaults for various options

Host \*

ForwardAgent no

ForwardX11 no

RhostsAuthentication no

RhostsRSAAuthentication no

RSAAuthentication yes

PasswordAuthentication yes

FallBackToRsh no

UseRsh no

BatchMode no

CheckHostIP yes

StrictHostKeyChecking no

IdentityFile ~/.ssh/identity

Port 22

Cipher blowfish

EscapeChar ~

下面逐行说明上面的选项设置：

Host \*

选项“Host”只对能够匹配后面字串的计算机有效。“\*”表示所有的计算机。

ForwardAgent no

“ForwardAgent”设置连接是否经过验证代理（如果存在）转发给远程计算机。

ForwardX11 no

“ForwardX11”设置X11连接是否被自动重定向到安全的通道和显示集（DISPLAY set）。

RhostsAuthentication no

“RhostsAuthentication”设置是否使用基于rhosts的安全验证。

RhostsRSAAuthentication no

“RhostsRSAAuthentication”设置是否使用用RSA算法的基于rhosts的安全验证。

RSAAuthentication yes

“RSAAuthentication”设置是否使用RSA算法进行安全验证。

PasswordAuthentication yes

“PasswordAuthentication”设置是否使用口令验证。

FallBackToRsh no

“FallBackToRsh”设置如果用ssh连接出现错误是否自动使用rsh。

UseRsh no

“UseRsh”设置是否在这台计算机上使用“rlogin/rsh”。

BatchMode no

“BatchMode”如果设为“yes”，passphrase/password（交互式输入口令）的提示将被禁止。当不能交互式输入口令的时候，这个选项对脚本文件和批处理任务十分有用。

CheckHostIP yes

“CheckHostIP”设置ssh是否查看连接到服务器的主机的IP地址以防止DNS欺骗。建议设置为“yes”。

StrictHostKeyChecking no

“StrictHostKeyChecking”如果设置成“yes”，ssh就不会自动把计算机的密匙加入“$HOME/.ssh/known\_hosts”文件，并且一旦计算机的密匙发生了变化，就拒绝连接。

IdentityFile ~/.ssh/identity

“IdentityFile”设置从哪个文件读取用户的RSA安全验证标识。

Port 22

“Port”设置连接到远程主机的端口。

Cipher blowfish

“Cipher”设置加密用的密码。

EscapeChar ~

“EscapeChar”设置escape字符。

——————————————————————————–

配置“/etc/ssh/sshd\_config”文件

“/etc/ssh/sshd\_config”是OpenSSH的配置文件，允许设置选项改变这个daemon的运行。这个文件的每一行包含“关键词－值”的匹配，其中“关键词”是忽略大小写的。下面列出来的是最重要的关键词，用man命令查看帮助页（sshd (8)）可以得到详细的列表。

编辑“sshd\_config”文件（vi /etc/ssh/sshd\_config），加入或改变下面的参数：

# This is ssh server systemwide configuration file.

Port 22

ListenAddress 192.168.1.1

HostKey /etc/ssh/ssh\_host\_key

ServerKeyBits 1024

LoginGraceTime 600

KeyRegenerationInterval 3600

PermitRootLogin no

IgnoreRhosts yes

IgnoreUserKnownHosts yes

StrictModes yes

X11Forwarding no

PrintMotd yes

SyslogFacility AUTH

LogLevel INFO

RhostsAuthentication no

RhostsRSAAuthentication no

RSAAuthentication yes

PasswordAuthentication yes

PermitEmptyPasswords no

AllowUsers admin

下面逐行说明上面的选项设置：

Port 22

“Port”设置sshd监听的端口号。

ListenAddress 192.168.1.1

“ListenAddress”设置sshd服务器绑定的IP地址。

HostKey /etc/ssh/ssh\_host\_key

“HostKey”设置包含计算机私人密匙的文件。

ServerKeyBits 1024

“ServerKeyBits”定义服务器密匙的位数。

LoginGraceTime 600

“LoginGraceTime”设置如果用户不能成功登录，在切断连接之前服务器需要等待的时间（以秒为单位）。

KeyRegenerationInterval 3600

“KeyRegenerationInterval”设置在多少秒之后自动重新生成服务器的密匙（如果使用密匙）。重新生成密匙是为了防止用盗用的密匙解密被截获的信息。

PermitRootLogin no

“PermitRootLogin”设置root能不能用ssh登录。这个选项一定不要设成“yes”。

IgnoreRhosts yes

“IgnoreRhosts”设置验证的时候是否使用“rhosts”和“shosts”文件。

IgnoreUserKnownHosts yes

“IgnoreUserKnownHosts”设置ssh daemon是否在进行RhostsRSAAuthentication安全验证的时候忽略用户的“$HOME/.ssh/known\_hosts”

StrictModes yes

“StrictModes”设置ssh在接收登录请求之前是否检查用户家目录和rhosts文件的权限和所有权。这通常是必要的，因为新手经常会把自己的目录和文件设成任何人都有写权限。

X11Forwarding no

“X11Forwarding”设置是否允许X11转发。

PrintMotd yes

“PrintMotd”设置sshd是否在用户登录的时候显示“/etc/motd”中的信息。

SyslogFacility AUTH

“SyslogFacility”设置在记录来自sshd的消息的时候，是否给出“facility code”。

LogLevel INFO

“LogLevel”设置记录sshd日志消息的层次。INFO是一个好的选择。查看sshd的man帮助页，已获取更多的信息。

RhostsAuthentication no

“RhostsAuthentication”设置只用rhosts或“/etc/hosts.equiv”进行安全验证是否已经足够了。

RhostsRSAAuthentication no

“RhostsRSA”设置是否允许用rhosts或“/etc/hosts.equiv”加上RSA进行安全验证。

RSAAuthentication yes

“RSAAuthentication”设置是否允许只有RSA安全验证。

PasswordAuthentication yes

“PasswordAuthentication”设置是否允许口令验证。

PermitEmptyPasswords no

“PermitEmptyPasswords”设置是否允许用口令为空的帐号登录。

AllowUsers admin

“AllowUsers”的后面可以跟着任意的数量的用户名的匹配串（patterns）或

[email=user@host]user@host[/email]

这样的匹配串，这些字符串用空格隔开。主机名可以是DNS名或IP地址。

使用SFTP代替FTP传输文件

FTP(文件传输协议)是一种使用非常广泛的在网络中传输文件的方式，但是，它也同样存在被网络窃听的危险，因为它也是以明文传送用户认证信息。其实在SSH软件包中，已经包含了一个叫作SFTP(Secure FTP)的安全文件传输子系统，SFTP本身没有单独的守护进程，它必须使用sshd守护进程（端口号默认是22）来完成相应的连接操作，所以从某种意义上来说，SFTP并不像一个服务器程序，而更像是一个客户端程序。SFTP同样是使用加密传输认证信息和传输的数据，所以，使用SFTP是非常安全的。但是，由于这种传输方式使用了加密/解密技术，所以传输效率比普通的FTP要低得多，如果您对网络安全性要求更高时，可以使用SFTP代替FTP。若要开启 SFTP功能可以修改sshd2\_config文件的下列内容：

# subsystem-sftp sftp-server

去掉行首的“#”，然后重新启动SSH服务器，这样在进行SSH连接时就可以同时使用SFTP传输文件。

关于客户端设置

以上是对服务器的设置，其实在SSH服务器中已经包含了一些客户端工具（如SSH,SFTP工具）。但是，更多的客户端用户使用Windows系统，下面就对Windows上的客户端系统设置加以说明。

首先从上文给出的网址下载“SSHSecureShellClient-3.2.3.exe”文件并安装。安装完成后，在桌面上会产成两个快捷方式，一个是“SSH Secure Shell Client”，用于远程管理，另一个是“SSH Secure File Transfer Client”，用于和服务器进行文件传输。在工具栏中点击“quick connnect”，输入正确的主机名和用户名，然后在弹出的对话框中输入密码完成登录，即可开始执行命令或者传输文件。在使用SFTP时，默认只能显示用户的宿主目录的内容和非隐藏文件。但是，有时候您可能还要查看其它目录或者隐藏文件，这时只需要在菜单“eidt->setting-> file transfer”的选项中选中“show root directory”和“show hidden file”两个选项即可。

使普通用户仅使用SFTP而没有使用Shell的权限

默认情况下管理员给系统添加的账号将同时具有SFTP和SSH的权限。让普通用户使用shell执行命令也是有很大的安全隐患的，如果能够禁止用户使用shell执行命令而仅使用SFTP传输文件，就能消除这种安全隐患，完全实现FTP的功能，

正如上文所述，SFTP没有单独的守护进程，只能借助于sshd守护进程，所以我们仍然需要使用SSH服务器，要保证sshd守护进程处于运行状态。具体实现方法如下：

首先，在编译安装时，编译中一定要有“–enable-static” 选项。安装成功后，在安装目录下的bin目录中执行下面的命令：

[root@localhost bin]# ls -l ssh-dummy-shell\* sftp-server2\*

将看到下列输出内容：

-rwxr-xr-x 1 root root 1350417 Apr 28 16:30 sftp-server2

-rwxr-xr-x 1 root root 3566890 Apr 28 16:30 sftp-server2.static

-rwxr-xr-x 1 root root 72388 Apr 28 16:30 ssh-dummy-shell

-rwxr-xr-x 1 root root 1813412 Apr 28 16:30 ssh-dummy-shell.static

其中带“static”后缀名，且比较大的两个文件就是加上“–enable-static”选项后生成的，后面我们将用到这里两个文件。

下面以添加普通账号test为例讲述具体操作步骤。

1．在“/home”目录（或者将要存放普通用户宿主目录的目录）下创建“bin”子目录，并将两个static文件复制到此目录下（复制后改名去掉static后缀），执行如下命令：

[root@localhost bin]# cd /usr/local/ssh3.2/bin

[root@localhost bin]#cp ssh-dummy-shell.static /home/bin/ssh-dummy-shell

[root@localhost bin]# cp sftp-server2.static /home/bin/sftp-server

[root@localhost bin]#chown -R root.root /home/bin

[root@localhost bin]#chmod -R 755 /home/bin

2．添加一个组，使以后所有禁止使用shell的用户都属于这个组，这样便于管理更多的用户：

[root@localhost bin]#groupadd template

3．在添加系统账号时使用如下命令：

[root@localhost root]#useradd -s /bin/ssh-dummy-shell -g template test

[root@localhost root]#passwd test

[root@localhost root]#mkdir /home/test/bin

[root@localhost root]#cd /home/test/bin

[root@localhost bin]#ln /home/bin/ssh-dummy-shell ssh-dummy-shell

[root@localhost bin]#ln /home/bin/sftp-server sftp-server

[root@localhost bin]#chown -R root.root /home/test/bin

[root@localhost bin]#chmod -R 755 /home/test/bin

3．用户添加成功后，还需要修改/etc/ssh2/sshd2\_config文件，将下列内容：

#ChRootGroups sftp,guest

改为：

ChRootGroups sftp,guest,template

修改上面这行内容，主要是为了禁止普通用户查看系统的其它目录，把其权限限制在自己的主目录下。重新启动SSH服务器程序，在客户端使用SSH Secure File Transfer Client登录，即使选择显示根目录，普通用户也看不到其它的任何目录，而是把自己的主目录当作根目录。注意，这里使用的是按用户所属组限制，这样可以使包含在template组内的所有用户都可以实现此功能。若您只要限制个别用户的话，可以修改下面的内容：

#ChRootUsers anonymous,ftp,guest

事实证明SSH是一种非常好的网络安全解决方案，但是，目前仍有很多管理员使用Telnet或FTP这种非常不安全的工具，希望尽快转移到SSH上来，以减少网络安全隐患。

# 四、vsFTPd

## 2.1、什么是FTP

FTP是File Transfer Protocol 文件传输协议的缩写,

FTP会话时包含了两个通道，一个叫控制通道，一个叫数据通道。

控制通道：控制通道是和FTP服务器进行沟通的通道，连接FTP，发送FTP指令都是通过控制通道来完成的。

数据通道：数据通道是和FTP服务器进行文件传输或者列表的通道。

FTP协议中，控制连接均由客户端发起，而数据连接有两种工作方式：PORT方式和PASV方式。

PORT模式(主动方式)

FTP 客户端首先和FTP Server的TCP 21端口建立连接，通过这个通道发送命令，客户端需要接收数据的时候在这个通道上发送PORT命令。PORT命令包含了客户端用什么端口(一个大于1024的端口)接收数据。在传送数据的时候，服务器端通过自己的TCP 20端口发送数据。FTP server必须和客户端建立一个新的连接用来传送数据。

PASV模式(被动方式)

在建立控制通道的时候和PORT模式类似，当客户端通过这个通道发送PASV 命令的时候，FTP server打开一个位于1024和5000之间的随机端口并且通知客户端在这个端口上传送数据的请求，然后FTP server 将通过这个端口进行数据的传送，这个时候FTP server不再需要建立一个新的和客户端之间的连接传送数据。

如果从C/S模型这个角度来说，PORT对于服务器来说是OUTBOUND，而PASV模式对于服务器是INBOUND，这一点请特别注意，尤其是在使用防火墙的企业里，这一点非常关键，如果设置错了，那么客户将无法连接。

### 2.1.1、Linux下ftp服务器

一般在各种linux的发行版中，默认带有的ftp软件是vsftp，从各个linux发行版对vsftp的认可可以看出，vsftp应该是一款不错的ftp软件。

使用如下命令可以检测出是否安装了vsftpd软件，

rpm -qa |grep vsftpd

例如在我的centos 6.5中显示的结果如下：



如果没有安装的话，可以下载安装，也可以使用软件源进行安装。这里不再赘述。

使用vsftpd软件，主要包括如下几个命令：

启动ftp：service vsftpd start

停止ftp：service vsftpd stop

重启ftp：service vsftpd restart

ftp的配置文件主要有三个，在centos6.5中位于/etc/vsftpd/目录下，分别是：

ftpusers  该文件用来指定那些用户不能访问ftp服务器。

user\_list 该文件用来指示的默认账户在默认情况下也不能访问ftp

vsftpd.conf   vsftpd的主配置文件

## 2.2、安装vsFTPd并配置虚拟用户

给实验室网站的服务器安装ftp服务器，安装完以后开始没有使用虚拟用户的方式，客户端连接的时候输入CentOS系统的一个用户名和密码，能是能正常上传下载，但是它访问到CentOS系统的所有目录，这也太不安全了，当然是不可行的。哎，没办法，最后还是设置成虚拟用户的方式吧，虽然麻烦点。虚拟用户的方式是所有客户端ftp用户都借助一个centos的系统上的用户进行上传下载。安装及配置步骤如下：(以下所有都是在root用户下操作)

### 2.2.1、准备工作

首先查看是否已经安装vsftpd

rpm -qa | grep vsftpd

如果没有，就安装，并设置开机启动

yum -y install vsftpd #主软件

yum -y install pam\* #用于用户认证)

yum -y install db4\* #生成虚拟用户的用户名密码的db文件

安装以后会发现vsftpd的配置路径在/etc/vsfpd/目录下。

同时会在系统内添加一个名称为ftp的用户（当然也有了叫ftp的一个组）。vim /etc/passwd 查看该用户的参数：

http://dl.iteye.com/upload/attachment/0082/4013/0ee118f7-daa2-34a6-8bdb-f970d29580dc.png

该用户不能通过登录系统，它的home主目录在/var/ftp。（如果要更改该用户的主目录等参数可以：usermod -d /home/ftpuser ftp 。但是这不是必要的。）

设置vsftpd开机启动：

chkconfig vsftpd on

这条命令也等同于：

chkconfig --level 2345 vsftpd on

通过chkconfig –list

http://dl.iteye.com/upload/attachment/0082/4019/c7efce30-a865-31bb-a1ab-58cdedd6b448.png

为了避免不能连接，修改SELinux（也可能没有必要）：

setsebool -P ftpd\_disable\_trans=1 //呵呵 直接关了。

setsebool -P ftp\_home\_dir=1 //设置ftp可以使用home目录

sersebool -P allow\_ftpd\_full\_access=1 //设置ftp用户可以有所有权限

mkdir /home/ftpuser

在home下新建一个ftpuser目录，这个就作为所有ftp文件上传下载的根目录（现在它的所有者是root，后面有更改它的所有者的操作）。

**vsftpd的配置文件**

|  |  |
| --- | --- |
| /etc/vsftpd/vsftpd.conf | 主配置文件 |
| /usr/sbin/vsftpd | Vsftpd的主程序 |
| /etc/rc.d/init.d/vsftpd | 启动脚本 |
| /etc/pam.d/vsftpd | PAM认证文件（此文件中file=/etc/vsftpd/ftpusers字段，指明阻止访问的用户来自/etc/vsftpd/ftpusers文件中的用户） |
| /etc/vsftpd/ftpusers | 禁止使用vsftpd的用户列表文件。记录不允许访问FTP服务器的用户名单，管理员可以把一些对系统安全有威胁的用户账号记录在此文件中，以免用户从FTP登录后获得大于上传下载操作的权利，而对系统造成损坏。（注意：linux-4中此文件在/etc/目录下） |
| /etc/vsftpd/user\_list | 禁止或允许使用vsftpd的用户列表文件。这个文件中指定的用户缺省情况（即在/etc/vsftpd /vsftpd.conf中设置userlist\_deny=YES）下也不能访问FTP服务器，在设置了userlist\_deny=NO时,仅允许 user\_list中指定的用户访问FTP服务器。（注意：linux-4中此文件在/etc/目录下） |
| /var/ftp | 匿名用户主目录；本地用户主目录为：/home/用户主目录，即登录后进入自己家目录 |
| /var/ftp/pub | 匿名用户的下载目录，此目录需赋权根chmod 1777 pub（1为特殊权限，使上载后无法删除） |
| /etc/logrotate.d/vsftpd.log | Vsftpd的日志文件 |

### 2.2.2、配置vsftpd.conf

所谓虚拟用户就是没有使用真实的帐户，只是通过映射到真实帐户和设置权限的目的。虚拟用户不能登录CentOS系统。修改配置文件/etc/vsftpd/vsftpd.conf,作如下介绍：

#设定不允许匿名访问, 12行

anonymous\_enable=NO

#设定本地用户可以访问。15行，/etc/passwd中的用户

local\_enable=YES

注：设定本地用户可以访问。注意：主要是为虚拟宿主用户，如果该项目设定为NO那么所有虚拟用户将无法访问。

#允许写入权限，包括修改，删除 18行

write\_enable=YES

#本地用户文件上传后的权限是-rw-r--r-- 22行

#本地用户上传最高权限为666，默认权限为666-022=644，即local\_umask=022

local\_umask=022

#允许匿名用户浏览，下载文件,默认没有这一项，只有在虚拟用户的配置文件里才有用

anon\_world\_readable\_only=YES

#默认注释掉了,允许匿名用户上传 27行

anon\_upload\_enable=YES

#默认注释掉了,允许匿名用户上传/建立目录 31行

anon\_mkdir\_write\_enable=YES

#默认没有这一项,允许匿名用户具有建立目录，上传之外的权限，如重命名，删除

anon\_other\_write\_enable=YES

#是否显示目录说明文件, 默认是YES 但需要手工创建.message文件,这个.message，

#只有用命令登陆或者用工具，才可以看见,他不是一个弹出对话框，而是一段字符,

#如在pub下建立一个.message,那么在客户端进入pub目录时就会显示.message文档中的内容 35行

dirmessage\_enable=YES

#记录使用者所有上传下载信息—日志 39行

xferlog\_enable=YES

#确保ftp-datad 数据传送使用port 20（20号端口）42行

connect\_from\_port\_20=YES

#默认注释了，开启匿名用户上传用户映射，和下面一行一起使用 47行

chown\_uploads=YES

#默认注释了，将匿名用户上传的文件的用户映射为whoever用户 48行

chown\_username=whoever

#将上传下载信息记录到/var/log/vsftpd.log中 52行，

xferlog\_file=/var/log/vsftpd.log

#日志标准格式 57行

xferlog\_std\_format=YES

#默认注释掉了这一项，如果使用者600秒没有动作，

#则强制离线，也就是指令超时时间， 60行

idle\_session\_timeout=600

#如果 client与 Server 间的数据传送在 120 秒内都无法传送成功，那 Client的联机就会被我们的 vsftpd 强制剔除！ 63行

data\_connection\_timeout=120

#设定支撑Vsftpd服务的宿主用户为手动建立的Vsftpd用户。

#注意，一旦做出更改宿主用户后，必须注意一起与该服务相关的读写文件的读#写赋权问题。比如日志文件就必须给与该用户写入权限等。 67行

nopriv\_user=ftpsecure

#设定是否支持异步传输功能。 72行

#async\_abor\_enable=YES

#默认被注释了，设定支持二进制模式的上传和下载功能，82 83行

ascii\_upload\_enable=YES

ascii\_download\_enable=YES

#ftp的欢迎信息 86行

ftpd\_banner=Welcome to blahFTP service.

#默认没有这一项，ftp的欢迎信息，欢迎信息存放在banner\_file.txt文件中，

#如果添加中文欢迎信息，就需要在xp中将此文件编辑好后上传到/etc/vsftpd 目录中。

#如果ftpd\_banner和banner\_file同时设置，那么banner\_file将覆盖ftpd\_banner的设置。

banner\_file=/etc/vsftpd/banner\_file.txt

#在默认配置下，本地用户登入FTP后可以使用cd命令切换到其他目录，这样会#对系统带来安全隐患。可以通过以下三条配置文件来控制用户切换目录。

#

#将所有用户锁定在主目录,YES为启用 NO禁用.(包括注释掉也为禁用)97行

#chroot\_local\_user=YES

#是否启动锁定用户的名单 YES为启用 NO禁用(包括注释掉也为禁用)98行

chroot\_list\_enable=YES

#用于指定用户列表文件，该文件用于控制哪些用户可以切换到用户家目录的上级目录。

chroot\_list\_file=/etc/vsftpd.chroot\_list

#上述3条规则，可以说是这个配置中最难理解的东西了，而且

#当3个选项同时设置时，更是有些困难，下面就具体设置给出实例

#①当chroot\_list\_enable=YES，chroot\_local\_user=YES时，

#在/etc/vsftpd.chroot\_list文件中列出的用户，可以切换到其他目录；

#未在文件中列出的用户，不能切换到其他目录。

#②当chroot\_list\_enable=YES，chroot\_local\_user=NO时，

#在/etc/vsftpd.chroot\_list文件中列出的用户，不能切换到其他目录；

#未在文件中列出的用户，可以切换到其他目录。

#③当chroot\_list\_enable=NO，chroot\_local\_user=YES时，

#所有的用户均不能切换到其他目录。

#④当chroot\_list\_enable=NO，chroot\_local\_user=NO时，

#所有的用户均可以切换到其他目录。

#是否允许递归查询 ，大型站点的 FTP 服务器启用此项可以方便远程用户查询

#ls\_recurse\_enable=YES

#如果设置为 YES ， 则 vsftpd 将以独立模式运行，由vsftpd自己监听和处理连接请求

listen=YES

#设定是否支持IPV6

#listen\_ipv6=YES

#设置PAM外挂模块提供的认证服务所使用的配置文件名，即/etc/pam.d/vsftpd文件

#此文件中file=/etc/vsftpd/ftpusers字段，说明了PAM模块能抵挡的帐号内容来自文件

#/etc/vsftpd/ftpusers中

pam\_service\_name=vsftpd

# 是否允许ftpusers文件中的用户登录FTP服务器，默认为NO

# 若此项设为YES，则user\_list文件中的用户允许登录FTP服务器

# 而如果同时设置了userlist\_deny=YES，

# 则user\_list文件中的用户将不允许登录FTP服务器，甚至连输入密码提示信息都没有

userlist\_enable=YES

# 设置是否阻扯user\_list文件中的用户登录FTP服务器，默认为YES

userlist\_deny=YES

# 是否使用tcp\_wrappers作为主机访问控制方式。

# tcp\_wrappers可以实现linux系统中网络服务的基于主机地址的访问控制

# 在/etc目录中的hosts.allow和hosts.deny两个文件用于设置tcp\_wrappers的访问控制

# 前者设置允许访问记录，后者设置拒绝访问记录。

# 如想限制某些主机对FTP服务器192.168.57.2的匿名访问，

# 编缉/etc/hosts.allow文件，如在下面增加两行命令：

# vsftpd:192.168.57.1:DENY 和vsftpd:192.168.57.9:DENY

# 表明限制IP为192.168.57.1/192.168.57.9主机访问IP为192.168.57.2的FTP服务器

# 此时FTP服务器虽可以PING通，但无法连接

tcp\_wrappers=YES

以上内容为vsftpd.conf的基本配置，以下这些是关于vsftpd虚拟用户支持的，默认vsftpd.conf中不包含这些设定项目，需要自己手动添加

#设定启用虚拟用户功能

guest\_enable=YES

#指定虚拟用户的宿主用户，CentOS中已经有内置的ftp用户了

guest\_username=ftp

#设定虚拟用户个人vsftp的CentOS FTP服务文件存放路径。

#存放虚拟用户个性的CentOS FTP服务文件(配置文件名=虚拟用户名

user\_config\_dir=/etc/vsftpd/vuser\_conf

#默认没有这一项，禁止10.1.6.0段访问此FTP

Listen\_address=10.1.6.0

### 2.2.3、配置FTP服务器的虚拟用户

在vsftpd服务器中支持匿名用户，本地用户，和虚拟用户3类用户账号，用途及区别如下：

匿名用户：是名为anonymous或ftp的FTP用户，匿名FTP用户登录后将FTP服务器中的/var/ftp作为FTP根目录。匿名用户通常用于提供公共文件的下载，如架设公共软件下载的FTP服务器，所有人都可以使用匿名用户进行软件下载。

本地用户：账号是FTP服务器中的系统用户账号，使用FTP本地用户账号登录FTP服务器后，登录目录为本地用户的宿主目录。本地FTP用户账号通常和Web服务器一起提供虚拟主机服务，作为网页虚拟主机更新网页的途径。

虚拟用户：账号是为了保证FTP服务器的安全性，由vsftpd服务器提供的非系统用户账号。虚拟用户FTP登录后将把指定的目录作为FTP根目录。虚拟用户与本地用户具有类似的功能，由于虚拟用户相对安全，因此正逐步替代本地用户账号。

由于虚拟用户账号具有较高的安全性，可以替代本地用户账号使用，下面是vsftpd虚拟用户账号设置的几个步骤：

以设置adonis1978和test两个虚拟用户帐号为例来配置vsftpd服务器

#### (1) 建立虚拟用户口令库文件

建立虚拟用户的口令文件，文件中奇数行设置虚拟用户的用户名，偶数行设置用户的口令。例如，使用vi编辑器建立名为vuser\_passwd.txt的用户口令库文件，在文件中设置用户adonis1978的口令为Ln18603093,用户test的口令是123456:

#vi vuser\_passwd.txt

adonis1978 ( 奇数行设置虚拟用户名)

Ln18603093 （偶数行设置用户口令）

test

123456

:wq(保存退出)

#### (2) 生成vsftpd 的认证文件

使用db\_load 命令生成认证文件。“-f”命令选项设置的值是虚拟用户的口令库文件，即上面创建的vuser\_passwd.txt。命令的参数设置为需要生成的认证文件名如 vuser\_passwd.db，该文件放置在目录/etc/vsftpd/下：

#db\_load -T -t hash -f vuser\_passwd.txt /etc/vsftpd/vuser\_passwd.db

#file /etc/vsftpd/vuser\_passwd.db #查看文件类型

生成的认证文件的权限应设置为只对root用户可读可写，即600

#chmod 600 /etc/vsftpd/vuser\_passwd.db

#### (3) 建立虚拟用户所需的PAM配置文件

在/etc/pam.d目录下建立vsftpd虚拟用户身份认证所需的PAM配置文件，名称是vsftpd.vu,内容为：

auth sufficient pam\_userdb.so db=/etc/vsftpd/vsftpd\_login

account sufficient pam\_userdb.so db=/etc/vsftpd/vsftpd\_login

auth required pam\_listfile.so item=user sense=deny file=/etc/vsftpd.ftpusers onerr=succeed

auth required pam\_stack.so service=system-auth

auth required pam\_shells.so

account required pam\_stack.so service=system-auth

session required pam\_stack.so service=system-auth

可以看出前面两行是对虚拟用户的验证，后面是对系统用户的验证。对虚拟用户的验证使用了sufficient这个控制标志。这个标志的含义是如果这个模块验证通过，就不必使用后面的层叠模块进行验证了；但如果失败了，就继续后面的认证，也就是使用系统真实用户的验证。

这里有一个非常不易被大家发现的问题就是64位的问题，注意，路径是lib64，而不是lib。还要写全，这个没有测试。

#cat /etc/pam.d/vsftpd.vu

auth required /lib64/security/pam\_userdb.so db=/etc/vsftpd/vuser\_passwd

account required /lib64/security/pam\_userdb.so db=/etc/vsftpd/vuser\_passwd

这里需要做一个重要的说明，/etc/pam.d/vsftpd和我们自己建立的vsftpd.vu的主要区别是，前者是系统pam文件，后者是专为虚拟用户提供的pam文件，同时，如果想本地用户和虚拟用户同时能登录ftp服务器，则需要读取vsftpd这个文件，如果不想本地用户登录则需要读取vsftpd.vu这个pam文件。这一点务必要清楚，不是任何设置问题，是文件名问题，并且这里还有一个关于pam文件的控制模式问题，如果都写在vsftpd中，必须把虚拟用户的验证写在前边，并且用sufficient模式，而不能用required模式，这一点也是也是至关重要的。

这就是已经建立好的关于本地和虚拟用户同时可登陆的vsftpd文件内容

auth sufficient /lib64/security/pam\_userdb.so db=/etc/vsftpd/vuser\_passwd

account sufficient /lib64/security/pam\_userdb.so db=/etc/vsftpd/vuser\_passwd

#%PAM-1.0

session optional pam\_keyinit.so force revoke

auth required pam\_listfile.so item=user sense=deny file=/etc/vsftpd/ftpusers onerr=succeed

auth required pam\_shells.so

auth include password-auth

account include password-auth

session required pam\_loginuid.so

session include password-auth

#### (4) 建立虚拟用户及要访问的目录并设置相应的权限

建立vsftpd虚拟用户所需的系统用户账号，账号名为virtualuser，指定用户的宿主目录是/home/ftpsite, 设置宿主目录的权限为700：

#useradd -d /home/ftpsite -s /sbin/nologin virtualuser

#chmod 700 /home/ftpsite/

#chown -R virtualuser:virtualuser /home/ftpsite

记得千万不要忘记改变用户所属，我们的虚拟用户都是通过这个宿主用户登录的，如果不更改这个目录的所属用户和组为virtualuser，则我们通过这个宿主用户virtualuser，创建的虚拟用户不能更改这个目录的属性，也就是没有操作这个目录权限，当然也就无法登陆了。

这样vsftpd服务器中的所有虚拟用户账号登录后都将在/home/ftpsite目录中

#### (5) 设置vsftpd.conf主配置文件

在对vsftpd.conf配置文件进行修改之前，应先将原有的文件进行备份，以便出现配置错误时可进行恢复：

#cd /etc/vsftpd

#cp vsftpd.conf vsftpd.conf.bak

在vsftpd.conf配置文件中添加虚拟用户的配置项，内容为：

guest\_enable=YES

guest\_username=virtualuser

pam\_service\_name=vsftpd.vu

这些东西当然你写在虚拟用户配置文件中也可以，那样比较直观。

#### (6) 重新启动vsftpd服务程序

在对vsftpd.conf配置文件进行了任何修改后都需要重新启动vsftpd服务，以便配置生效：

#service vsftpd restart

#### (7) 对不同的虚拟用户设置不同的权限

vsftpd服务器中的虚拟用户可以灵活的针对不同的用户账号设置不同的用户权限，配置的步骤如下：

1．设置主配置文件

在vsftpd.conf配置文件中添加user\_config\_dir配置项，并设置用户配置文件的保存目录

user\_config\_dir=/etc/vsftpd/vuser\_conf

这一行通常写在vsftpd.conf中。

在上面的配置实例中，设置在/etc/vsftpd/vuser\_conf目录中保存虚拟用户的配置文件，这个目录需要自己建立。

#mkdir /etc/vsftpd/vuser\_conf

2.为虚拟用户建立单独的配置文件

在/etc/vsftpd/vuser\_conf目录中可以为每个虚拟用户建立独立的配置文件，配置文件名称和用户名相同。例如，为用户adonis1978建立配置文件adonis978，为用户test建立配置文件test，并设置该用户具有浏览，下载，上传，改名，删除文件，建立和删除的权限。

#开启虚拟用户

guest\_enable=YES

#虚拟用户的系统宿主用户

guest\_username=virtualuser

#这两行我把它写在虚拟用户自己的配置文件中了，那是因为我调用

#的是vsftpd这个全局pam文件，如果是调用vsftpd.vu,

#就可以把这两行写入到vsftpd.conf中，

#因为，这两个pam文件影响的用户不同，

#vsftpd可实现本地用户和虚拟用户同时登陆ftp服务器。

#vsftpd.vu只能实现虚拟用户登陆ftp服务器。

#指定虚拟用户登陆目录，通过分别写在各自自己的配置文件中，来实现用户只能登陆到指定目录。

local\_root=/home/ftpsite/adonis1978

#用户可以浏览和下载文件，不能设为yes，否则无法看到文件

anon\_world\_readable\_only=NO

#用户可以创建文件

write\_enable=yes

#用户可以上传文件

anon\_upload\_enable=YES

#用户有创建和删除目录的权限

anon\_mkdir\_write\_enable=YES

#用户具有文件改名和删除文件的权限

anon\_other\_write\_enable=YES

通过对以上配置项的组合设置，vsftpd可以为每个虚拟用户配置不同的FTP权限，用户配置文件中没有的配置项将按照vsftpd.conf配置文件中的内容设置。

#### (8)安全选项

#限制使用者不能离开家目录,例如blue登陆后位于/home/blue下,设置该选项后,他不可以转到/home/blue的上层目录,如/bin, /usr,/opt...etc。

chroot\_list\_file=/etc/vsftpd/chroot\_list /默认这一项注释掉了

#与上条同时使用，设置要禁固宿主目录的用户的配置文件为/etc/vsftpd/chroot\_list 建立文本文件/etc/vsftpd/chroot\_list，写入要禁固宿主目录的用户，一行一个。当chroot\_list\_enable=YES时，则禁固chroot\_list文件里面的用户的宿主目录，而为NO时chroot\_list文件里面的用户不禁固宿主目录。

tcp\_wrappers=YES /默认没有这一项

#支持tcp\_wrappers,限制访问(/etc/hosts.allow,/etc/hosts.deny)

xferlog\_std\_format=YES

listen=YES /109行

#使用standalone启动vsftpd，而不是super daemon(xinetd)控制它 (vsftpd推荐使用standalone方式) 108行

listen\_port=21 /默认没有这一行

#ftp监听端口

pam\_service\_name=vsftpd /116行

#PAM所使用的名称.同userlist\_\*一样限制用户登陆，不同的是userlist\_\*在进行密码验证之前拒绝用户登陆，pam是在密码验证之后拒绝登陆.(提示密码错误) 用户列表默认存放在/etc/ftpusers中，一行一个. (可通过/etc/pam.d/vsftpd重定向用户列表存放文件)

#获得 /etc/pam.d/vsftpd：

#[root@LFSvsftpd-2.0.1]#cp RedHat/vsftpd.pam/etc/pam.d/vsftpd

userlist\_enable=YES / 为yes时， /etc/vsftpd/user\_\_list文件中的用户将不能访问vsftpd服务器 117行

userlist\_deny=YES /默认没有这一项，与下面这条参数：userlist\_file=/etc/vsftpd/localuser\_list 结合使用。

userlist\_file=/etc/vsftpd/localuser\_list /默认没有这一项, 与上面参数：userlist\_deny=YES结合使用此参数用，来改变存放可以登录或禁止登录的账号，suerlist\_deny为YES时,这里面的用户则不能登录，当为NO时，这里面的用户可以登录，和/etc/vsftpd/user\_list是一个意思，

max\_clients=100 /默认没有这一项

#最大用户在线数量

max\_per\_ip=2 /默认没有这一项

#每ip最大线程

anon\_max\_rate=30000 /默认没有这一项

#匿名用户最大传输速度 单位: bytes/秒

local\_max\_rate=50000 /默认没有这一项

#本地用户最大传输速度 单位：bytes/秒

user\_config\_dir=/etc/userconf /默认没有这一项

#个别用户配置目录(用来设定特殊帐号),例如我想让blue这个用户的传输速度是100KB

就可以在/etc/userconf/下创建文本文件blue(与用户名相同),加入local\_max\_rate=100000即可 /默认没有这一项

anon\_root=/var/ftp /默认没有这一项

#设定匿名用户登陆后所在的目录

local\_root=/var/local\_user /默认没有这一项

#设定所有本地用户登陆后的目录，如不设置此项，则本地用户登陆后位于各自家目录下。

use\_localtime=YES /默认没有这一项

#使用本地时间而不是GMT

nopriv\_user=vsftpd /默认没有这一项

write\_enable=YES 可写

local\_umask=033 文件上传后的属性掩码

async\_abor\_enable=YES 异步传输功能

chroot\_local\_user=YES 把用户限制在自己的主目录

chroot\_list\_enable=YES 可以配置用户，允许访问主目录外的文件

chroot\_list\_file=/etc/vsftpd/chroot\_list 配置文件

listen=YES 监听

listen\_port=18221 修改监听端口，默认是 21

pam\_service\_name=vsftpd 认证，默认就有的

userlist\_enable=YES 默认就有的

tcp\_wrappers=YES 默认就有的

guest\_enable=YES 允许虚拟用户

guest\_username=ftpuser 虚拟用户名，和上面一致

virtual\_use\_local\_privs=YES 设置虚拟用户权限

user\_config\_dir=/etc/vsftpd/vconf 虚拟用户配置目录

max\_clients=20 最大连接数

# 四、MySQL

# 五、Mavercisk OS X

## 5.1、VMware11安装Mac OS X10提示不可恢复错误(vcpu-0)

VMware11安装Mac OS X10提示不可恢复错误(vcpu-0)怎么办?某论坛一天浏览苹果Mac系统页面的网友，竟然有一千七百多，看样子越来越多的朋友开始琢磨使用虚拟机安装Mac OS了。这不这位朋友在使用VMware11，安装Mac OS X10.10的时候，出现如下提示界面：

VMware Workstation不可恢复错误：(vcpu-0)

vcpu-0:VEREIFY vmcore/vmm/main/physMem\_monitor.c.1123

日志文件位于“E:OS X\vmware.log”中。

你可以请求支持。

要收集数据提供过给VMware技术支持，请选择“帮助”菜单中的“收集支持数据”。

也可以直接在VMwarestation文件夹中运行“vm-support”脚本。

出现这种情况，只要找到并打开vmx文件，

使用记事本打开后，添加“smc.version = 0”后保存，

问题即可解决

## 5.2、新建文件

很多mac用户是windows用户过来的，当他们用mac时不可避免的会把原来使用windows的习惯带到mac中，已达到使用windows经验的延续，实际上这是非常错误的，因为这是两个完全不兼容的系统，除了操作系统理论层面的东西，他们之间，几乎没有交集，就像很多人讲历史上的一些官员名称，总是和今天的官员做对比，实际上这是不准确的，比如三公这个概念，就经常有很多人说相当于今天的什么什么，其实呢，你很难找到一个和他相提并论的职位。不信，我先来给你介绍一下这个三公的概念，这可不是今天的三公消费，是指中国古代官场的最高官阶的三种职位，称为三公，位列三公，就是说你这官做到了顶点，位极人臣，那么我们看看什么是三公，讲历史了，呵呵：

据杜佑的《通典》记载：

夏、商以前，云天子无爵，三公无官。

周以太师、太傅、太保曰三公。

汉以丞相、大司马、御史大夫为三公。

后汉又以太尉、司徒、司空为三公。

魏、晋、宋、齐、梁、陈、后魏、北齐皆以太尉、司徒、司空为三公。

后周以太师、太傅、太保为三公。

隋以太尉、司徒、司空为三公。

大唐因之。”

翻译一下：

在夏朝和商朝以前并没有三公这个职称的。

周朝时候，“三公”是指：太师、太傅以及太保。

秦朝、西汉时候，改以丞相、太尉、御史大夫为“三公”。

西汉末至东汉初期演变为：大司马、大司徒、大司空，汉光武帝建武二十七年改为太尉、司徒、司空。

后来，魏、晋、宋、齐、梁、陈等皆沿用以太尉、司徒、司空作为“三公”。

到了后周，又改回原来周朝的称呼：仍以太师、太傅以及太保为“三公”。

隋朝再次称“三公”为太尉、司徒、司空。

唐朝时候就延续隋朝制度，基本上没有大的变动。

历代三公

周朝：

周立太师、太傅、太保为三公（“师，天子所师法；傅，傅相天子；保，保安天子于德义。”）。自周衰，官失而百职乱，战国并争，各有变易。

秦朝：

通说为：

（1）丞相，最高行政长官，辅助皇帝处理政务，同时负责对文武百官的管理。

（2）太尉，最高军政长官，负责管理全国军事事务，但他平时没有军权，战时也要听从皇帝的命令，而且要有皇帝的符节才能调动军队，军权实际上也是掌握在皇帝手里。

（3）御史大夫，执掌全臣奏章，下达皇帝诏令，负责监察百官的，也是副丞相。

根据《汉书·百官公卿表》的记载，丞相金印紫绶，御史大夫银印青绶（御史大夫，秦官，位上卿），从其印绶和称为“上卿”来看，御史大夫不是公，而属卿一级，因此秦代并未有三公之称呼。

西汉：

初承秦制，中央的最高官职仍是丞相、太尉、御史大夫，可见汉初和秦一样，也并没有并列的三公，不过一般还是沿袭旧号，把最高官吏称为三公。

另说：设置三公。汉武帝元狩四年，置大司马，以取代太尉之职。后元二年（前87年）春，汉世宗孝武帝刘彻驾崩，大司马大将军霍光受汉武帝遗诏，与车骑将军金日磾、左将军上官桀、御史大夫桑弘羊等人成为汉昭帝刘弗陵的辅命大臣，共同执掌朝政。从此，霍光掌握了汉朝政府的最高权力，三公中的丞相地位急剧下滑。“汉哀帝元寿二年，置太傅，位在三公上； 平帝元始元年置太师、太保，太师位在太傅上，太保次太傅。其后或废或置，大抵无所统职。

“太傅，古官，高后元年初置，金印紫绶。后省，八年复置。哀帝元寿二年复置。位在三公上。太师、太保，皆古官，平帝元始元年皆初置，金印紫绶。太师位在太傅上，太保次太傅。”[1]

东汉：

西汉末至东汉初，以大司马、大司徒、大司空为三公。至汉光武帝建武二十七年，省大司马，又置太尉，以太仆赵熹为之，而与司徒、司空为三公。汉献帝初，刘虞、李傕为大司马，而太尉如故，二职始两置矣，大司马位在三公之上。汉制，三公府分部九卿，太尉所部太常、卫尉、光禄三卿。汉献帝建安十三年，省三公官，置丞相。魏初又置，而兼置大司马，大司马一般由大将军升迁而来。明黄道周《节寰袁公（袁可立）传》：“张然明破诸羌，静幽并，耻为王曹所卖，发愤申陈窦之冤，卒不得为三公。”

三国：

三国时期：太尉、司徒、司空为三公，将太师、太傅、太保特称三师，以正其名。然非道德崇重则不居其位，无其人则阙之，故后世历代多以为赠官。太傅、大司马、大将军为上公，位在三公之上。其中太傅和三公（太尉、司徒、司空）一般为虚职，大司马和大将军掌实权。吴设丞相、大司马（另有左、右大司马）、上大将军，其中上大将军位在大将军之上，大司马之下。

晋朝：

晋以相国、丞相与八公为诸公官。《通典》卷二《职官二》“三公总叙”条说：“后魏以太师、太傅、太保谓之三师，上公也。大司马、大将军谓之二大，太尉、司徒、司空谓之三公。”

以司马望为太尉。历宋、齐、梁、陈、后魏、北齐，并为三公，置府僚。

隋唐：

置太尉、司徒、司空为三公，正一品。《唐六典》载：“三公，论道之官也。盖以佐天子，理阴阳，平邦国，无所不统，故不以一职名其官。然周、汉已来，代存其任。自隋文帝罢三公府僚，皇朝因之，其或亲王拜者，亦但存其名位耳。”

唐代分为中书省，门下，尚书省。职责分别为决策，审议，执行。

后周：

依《周官》，以太师、太傅、太保为三公，不置府僚。

宋朝：

北宋，承唐制，以太师、太傅、太保为三师，太尉、司徒、司空为三公。至宋徽宗，依三代旧制，改三公为太师、太傅、太保，罢司徒、司空、太尉。

元朝：

由于太尉等衔加授过滥，元朝中人已渐渐不再称它们为三公，而以三公指代前朝一般称为“上公”、“三师”的另外三个更高的荣誉头衔——太师、太傅、太保。

至此，后世历代皆以太师、太傅、太保为三公。在明仁宗之后，三公皆为虚衔，为勋戚文武大臣加官、赠官。

从上面就可以看出，历史上的三公其含义并不相同，其权利大小也相差很多，不管是名称上的不同，太尉和司马都是所谓的军事上的最高长官，但本身就是矛盾的，三国时期，就同时有这两个官阶，那时太尉已经不是三公了，很多人都认为像太尉这样的官阶，就是今天的国防部长，其实这也是不对的，因为，太尉在和平时期，是没有兵权的，只有战争时期，他才有指挥权，并且还需要皇帝的节符，而今天的国防部长，显然比这个权力要大多了，只能说为了理解这个官阶，做个比喻罢了，另外像御史大夫，被说成今天的中纪委，这就是相差更多了。今天中纪委，只是一个纪律监督部门，而御史大夫，还有人大的功能，所以你要想很好的理解这些官阶的实际功能，你必须用当时的环境来了解，不可粗暴的比喻。

言归正传，linux和windows也是一样，不能用windows的概念简单来理解linux，因为，他们的结构完全不同，很多系统级概念都是不同的，所以不能简单用实现的结果，来说明他们功能上的相似性。

比如文件系统，windows有文件类型的概念，通过不同的后缀名来区别不同的文件，并且文件头信息也是不一样的，而linux则没有类似的概念，有的同学说，linux文件也有后缀名呀，但linux文件后缀名并不是代表文件类型，而是一种让查看者方便查询的标识，也就是说他并没有什么实际意义，比如说windows中文本文件的后缀名是txt，可执行文件后缀名是exe，而在linux中，任何文件都可以是文本文件，都可以用编辑器打开，当然你要有相应的权限，任何文件也都是可执行，同样也是权限，也就是linux中是用权限来区分文件的，而不是后缀名，比如我们给一个txt文件赋予可执行权限，那么，这个txt文件就是可执行的，当然他也需要相应的解释器，像mac这样的类unix系统，为了方便用户是用，也把一些常用的文件关联了一些应用程序，这一点和windows很像，但不同的是，只要有执行权限，你就可以执行。

那么回到我们最开始说的这个问题，在mac系统中创建一个文件，也许是出于安全性的考虑，mac系统，并没有提供创建一个文件的快捷方式，你只能通过应用程序来创建这好像有点繁琐。

如果你是linux用户，那么这就不是问题了，我们通过命令行来直接创建，别忘了，mac支持大多数的linux命令，那就好办了

touch filename

vi filename

vim filename

nano filename

通常我们用第一个命令就可以了，这个非常方便，可以一次创建多个文件，如果你要可执行，别忘了赋权限。

后边的都是linux标准编辑器，其实和用应用程序创建文件差不多了，只是这些编辑器不用你安装了，因为他们都是linux标准编辑器。

很多同学一定很奇怪，创建文件为什么起这个名字touch，而不是用create，new这样的名字，其实touch根本就不是用来创建文件的，他的中文意思是触摸，他实际上是用来更新文件的访问和修改时间，下面给出touch命令的原始含义，你一看就知道了，所以。。。。

语法

**touch [ -a ][ -c ][ -m ][ -f ] [ -r RefFile ] [ Time | -t Time ] { File ... | Directory ... }**

touch 命令更新由 Directory 参数指定的每个目录下的由 File 参数指定的每个文件的访问和修改时间。如果没有指定 Time变量值，touch 命令就使用当前时间。如果指定了一个不存在的文件，touch 命令就创建此文件，除非指定了 -c 标志。

touch 命令的返回码是时间没有被成功修改的文件数目（包括不存在的文件和没有创建的文件）。

注:

任何超出 2038 年（包含2038年）的日期都是无效的。

示例

1. 要更新文件的访问和修改时间，请输入：

touch program.c

这会将 program.c 文件的上次访问和修改时间设置为当前日期和时间。如果 program.c 文件不存在，touch 命令就创建一个同名的空文件。

2. 要避免创建新文件，请输入：

touch -c program.c

3. 要仅更新修改时间，请输入：

touch -m \*.o

这会更新当前目录下的以 .o 扩展名结尾的文件的上次修改时间（不更新访问时间）。touch 命令经常以此方式使用，来改变 make 命令的结果。

4. 要明确设置访问和修改时间，请输入：

touch -c -t 02171425 program.c

这会将访问和修改时间设置为当前年份的 2 月 17 日的 14:25（2:25 p.m.）。

5. 要使用另一文件的时间戳记而不用当前时间，请输入：

touch -r file1 program.c

这会将 program.c 文件的时间戳记设置成与 file1 文件一样。

6. 要使用指定时间而不是当前时间处理文件，请输入：

touch -t 198503030303.55 program.c

这会将 program.c 文件的时间戳记设置为 1985 年 3 月 3 日上午 3:03:55。

这样Windows下就这么一个简单的工作，在linux下啰嗦了这么多，实际上，是想告诉你，不要试图用windows来解释linux，那样你永远也学不会用linux的方式思考，你必须站在linux的角度，来处理问题，而不是windows。

就像touch的本意并不是创建文件，而是当要更新的文件不存在时，他会创建一个同名的文件，显然这不是必须的，你可以通过-c参数来阻止它创建新文件，呵呵呵。

## 5.3、mac的一些快捷键

#### 1、打开一个副本

这个操作比较简单，就是command+n，但是使用的时候要注意的是，你要创建那个应用的副本，你就要先把他激活，比如我们像打开多个qq应用，你在launchpad中是不能执行多次的，也就是说，你想打开一个副本，是打不开的，你用command+n则可以打开多个应用副本。

#### 2、截图

mac的截图，和qq截图差不多，command+shift+3|4，3是直接全屏截图，4是出现选择框截图，默认截图是直接存在桌面上的，你可以修改存储路径，来指定任意位置。

MAC OS X系统默认的截图路径是桌面文件夹，默认的截图格式是 PNG 图片格式，如何自定义设置呢？

截图保存路径

打开终端（Terminal）并输入如下命令：

defaults write com.apple.screencapture location /path/

/path/即你想要设置的保存路径，比如将截图文件存储在用户目录的图片（Pictures）文件夹，命令则为：

defaults write com.apple.screencapture location ~/Pictures/

接下来，为了使得修改生效，输入下面这条命令：

killall SystemUIServer

当然，如果你习惯将截图保存在桌面的话，/path/ 对应的路径为 ~/Desktop/。

截图文件格式

OS X 系统默认截图文件格式为 png，通过下面的终端（Terminal）命令可以修改截图文件格式为 jpg ：

defaults write com.apple.screencapture type jpg

另外，还可以将命令中的 jpg 修改为 gif 或 pdf，以及改回默认的 png 格式。同样别忘了输入上面那条使得修改生效的命令。

#### 3、复制、粘帖、剪切

基本上mac的command就相当于windows的Ctrl键，所以

全选：command+a

剪切：command+x

复制：command+c

粘帖：command+v，说明一下，Word中因为有格式的区别，所以粘帖还有一些选项，比如有无格式等，可以通过Ctrl+command+v来调出选择框。

Ctrl键通常用^号来替换。

#### 4、窗口最小化，关闭应用程序

这个对于窗口的控制，只有最小化，没有最大化，用command+m

关闭窗口用command+w或者command+q

当 Mac OS X 的应用开始“无敌风火轮（转菊花）”时可真让人无奈啊，它就这么卡死了。这时要么等应用恢复响应，或者干脆强制退出它，这里介绍 6 种强制退出应用的快捷键操作，有时候能省不少时间。

1、通过键盘强制退出当前能够响应的 Mac 应用

按住 Command + Option + Shift + Esc 键一两秒，直到应用被强制退出。这是退出有前台界面的应用的最快方法了。

2、调出“强制退出应用”窗口

按下 Command + Option + Esc 键，在弹出的“强制退出应用”窗口中点选要退出的应用名称，并点按“强制退出”按钮，该应用就会立刻退出。这可以当做一个简化版的活动监视器来使用，这套快捷键也比较好记。

3、从 Dock 强制退出应用

按住 Option 并右键点按 Dock 上的应用图标，在菜单中会出现“强制退出”选项，点选此项便可以将应用退出，不会弹出任何提示。

4、从苹果菜单中强制退出应用

按住 Shift 键并点按应用的菜单（左上角）调出“强制退出 [应用名称]”。这种方法也十分容易记住，不过并不是最好用的方法，因为有时候应用失去响应时，它的菜单也会无法使用。

5、使用活动监视器来退出应用

按 Command + 空格键来调出 Spotlight，输入 Activity Monitor 便可启动活动监视器。点选你要退出的进程名称或进程 ID，然后点退出（kill），失去响应的应用通常会显示成红色，然后点按红色的“退出进程”按钮。这基本上等于 Windows 系统中的“任务管理器”，或者可以看成之前的“强制退出应用”窗口的完整版。如果前边的方式都无效，这种方法大部分情况下是能够凑效的。

Activity Monitor

6、使用终端 kill 命令

如果所有办法都没法退出某个应用，还有一种选择：使用终端命令。尤其是退出某些底层进程时。启动终端（应用程序 – 实用工具 – 终端.app），并输入下边的命令：

killall [processname]

例如，“killall Safari”就会将 Safari 进程的所有实例都“杀死”。如果你知道进程 ID（可以通过 ps 或 ps aux 命令查询），则可以通过进程 ID 来进行强制退出：

kill -9 [pid]

终端里的 kill 命令几乎在所有场合都会有效，并且伴有一些副作用，例如不会按照 Versions、窗口恢复、自动保存等的方法进行退出，这样可能有时会导致数据丢失。

“附赠”一条：在 iOS 上强制退出应用

除了长按 Home 键调出简易的后台管理之外，还有一个方法：按住电源钮，在出现滑动并关机的画面时，按住 Home 键直到当前的应用被强制退出。

5、浏览器中新建标签页

command+t

6、复制帮助文档

man ps |col -b > ps.txt

# 六、HTML5 CSS3 现代网页制作

# 七、shell编程

## 7.1、VI下删除文本中的^M

关于回车与换行：

很久以前，老式的电传打字机使用两个字符来另起新行。一个字符把滑动架移回首位 (称为回车,ASCII码为0D)，另一个字符把纸上移一行 (称为换行,ASCII码为0A)。当计算机问世以后，存储器曾经非常昂贵。有些人就认定没必要用两个字符来表示行尾。UNIX 开发者决定他们可以用一个字符\n来表示行尾，Linux沿袭Unix，也是。Apple 开发者规定了用\r来表示行尾。开发 MS-DOS以及Windows 的那些家伙则决定沿用老式的\r\n

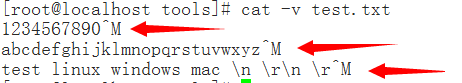
三种行尾格式如下:

Unix/linux : \n #0A

Windows : \r\n #0D0A

Mac : \r #0D

这意味着，如果你试图把一个文件从一种系统移到另一种系统，那么你就有换行符方面的麻烦。因为MS-DOS及Windows是回车＋换行来表示换行，因此在Linux下用Vim查看在Windows下写的代码，行尾有“^M”符号。通常你是看不到这个^M的，你用vi或者cat，查看，并没有发现这个^M，如果你用cat -v查看你就会发现行尾有这个了。



这里，你要注意了，这个^M可不是“^”和“M”这两个字符，

而是Ctrl+V+M 或者 [Ctrl+V]+[Ctrl+M]

由于换行符的不同，导致你查看文件出现问题，比如windows文本在linux下查看，行尾会出现^M，这会导致判断行尾出现错误。

而linux文本在windows下查看，你会发现，他都是连在一起的，没有分行。

那么我们首先说一下，如何把windows的文本中的^M去掉。

其实这个很简单，有很多方法可以去掉，这里把所有的方法给大家介绍一下：

（1）是用VI的命令： 使用vi打开文本文件

vi dos.txt

命令模式下输入

:set fileformat=unix

:w

(2) VI下使用正则表达式替换

g/^M/s/^M//

或者

%s/^M//g

(3) 使用sed 工具

sed ’s/^M//’ filename > tmp\_filename

(4)既然window下的回车符多了‘ ’，那么当然通过删除‘ ’ ，也可以实现：

tr -d " " filename

http://www.cnblogs.com/wangkangluo1/archive/2012/05/31/2528059.html

（5）最后一个方法是本人最常用的方法，个人觉得最方便

在终端下敲命令：

$ dos2unix filename

直接转换成unix格式，就OK了！～

7.1.2、

7.1.3、

7.1.4、

在Vim中解决这个问题，很简单，在Vim中利用替换功能就可以将“^M”都删掉，键入如下替换命令行：

# 待编辑区

## [配置 yum 源的两种方法](http://blog.csdn.net/rachel_luo/article/details/8692488)

分类： [Linux 学习](http://blog.csdn.net/rachel_luo/article/category/1103154) 2013-03-19 16:40 256人阅读 [评论](http://blog.csdn.net/rachel_luo/article/details/8692488#comments)(0) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/rachel_luo/article/details/8692488#report)

[yum](http://www.csdn.net/tag/yum)[yum源配置](http://www.csdn.net/tag/yum%e6%ba%90%e9%85%8d%e7%bd%ae)

## 由于 [****redhat****](http://space.itpub.net/25313300/viewspace-708509)的yum在线更新是收费的，如果没有注册的话不能使用，如果要使用，需将redhat的yum卸载后，重启[****安装****](http://space.itpub.net/25313300/viewspace-708509)，再配置其他源，以下为详细过程：

**1.删除redhat原有的yum**   
rpm -aq|grep yum|xargs rpm -e --nodeps   
  
2.下载yum安装文件   
wget http://mirrors.163.com/centos/6/os/x86\_64/Packages/yum-3.2.27-14.el6.centos.noarch.rpm   
wget http://mirrors.163.com/centos/6/os/x86\_64/Packages/yum-metadata-parser-1.1.2-14.1.el6.x86\_64.rpm   
wget http://mirrors.163.com/centos/6/os/x86\_64/Packages/yum-plugin-fastestmirror-1.1.26-11.el6.noarch.rpm   
wget http://mirrors.163.com/centos/6/os/x86\_64/Packages/python-iniparse-0.3.1-2.1.el6.noarch.rpm   
  
3.进行安装yum   
  
rpm -ivh python-iniparse-0.3.1-2.1.el6.noarch.rpm   
rpm -ivh yum-metadata-parser-1.1.2-14.1.el6.x86\_64.rpm   
rpm -ivh yum-3.2.27-14.el6.centos.noarch.rpm um-plugin-fastestmirror-1.1.26-11.el6.noarch.rpm    
注意最后两个包必需同时安装，否则会相互依赖   
  
4.更新repo文件

mv /etc/yum.repos.d/rhel-debuginfo.repo /etc/yum.repos.d/rhel-debuginfo.repo.repo.bak

vi /etc/yum.repos.d/rhel-debuginfo.repo

内容为：

[base]  
name=CentOS-$releasever - Base  
baseurl=http://mirrors.163.com/centos/6.0/os/$basearch/  
gpgcheck=1  
gpgkey=http://mirrors.163.com/centos/RPM-GPG-KEY-CentOS-6  
  
#released updates  
[updates]  
name=CentOS-$releasever - Updates  
baseurl=http://mirrors.163.com/centos/6.0/updates/$basearch/  
gpgcheck=1  
gpgkey=http://mirrors.163.com/centos/RPM-GPG-KEY-CentOS-6  
  
#packages used/produced in the build but not released  
#[addons]  
#name=CentOS-$releasever - Addons  
#baseurl=http://mirrors.163.com/centos/$releasever/addons/$basearch/  
#gpgcheck=1  
#gpgkey=http://mirrors.163.com/centos/RPM-GPG-KEY-CentOS-6  
#additional packages that may be useful  
[extras]  
name=CentOS-$releasever - Extras  
baseurl=http://mirrors.163.com/centos/6.0/extras/$basearch/  
gpgcheck=1  
gpgkey=http://mirrors.163.com/centos/RPM-GPG-KEY-CentOS-6  
#additional packages that extend functionality of existing packages  
[centosplus]  
name=CentOS-$releasever - Plus  
baseurl=http://mirrors.163.com/centos/6.0/centosplus/$basearch/  
gpgcheck=1  
enabled=0

5、yum clean all

6、yum install vim #测试一下可不可以用

【设置本地镜像作为yum源】

1、mkdir /mnt/cdrom

2   mount /dev/cdrom /mnt/cdrom  #挂着镜像，光盘

3  mkdir /home/redhat-iso

4  cp -Rf /mnt/cdrom/\* /home/redhat-iso   #将光盘内的内容copy到某个路径下

5  vi /etc/yum.repos.d/rhel-debuginfo.repo  #编辑repo文件

添加一下内容：

[rhel\_6\_iso]  
name=local iso   
baseurl=file:///home/redhat-iso/  
gpgcheck=1  
gpgkey=file:///home/redhat-iso/RPM-GPG-KEY-CentOS-6 #该处的gpgkey文件应该改成iso镜像中的文件名

【关于yum的一点资料】

**一、yum是什么**  
yum = Yellow dog Updater, Modified  
主要功能是更方便的添加/删除/更新RPM包.  
它能自动解决包的倚赖性问题.  
它能便于[**管理**](http://space.itpub.net/25313300/viewspace-708509)大量系统的更新问题

**二、 yum特点**  
\*可以同时配置多个资源库(Repository)  
\*简洁的配置文件(/etc/yum.conf，/etc/yum.repos.d 下的文件)  
\*自动解决增加或删除rpm包时遇到的倚赖性问题  
\*使用方便  
\*保持与RPM数据库的一致性  
  
**三、 yum安装**  
Centos/RHEL  
1. #rpm -ivh yum-2.4.3-4.el4.centos.noarch.rpm  
  
**四、yum配置**  
yum的配置文件  
yum的一切配置信息都储存在一个叫yum.conf的配置文件中，通常位于/etc目 录下，这是整个yum系统的重中之重，所以有必要详细介绍。下面是一个从网上找来的yum.con文件，让我们以此为例，进行说明。

|  |
| --- |
| [main]  cachedir=/var/cache/yum  debuglevel=2  logfile=/var/log/yum.log  pkgpolicy=newest  distroverpkg=redhat-release  tolerant=1  exactarch=1  retries=1   [base]  name=Fedora Core $releasever - $basearch - Base  baseurl=http://download.atrpms.net/mirrors/fedoracore/$releasever/$basearch/os  <http://rpmfind.net/linux/fedora/cor...er/$basearch/os>  <http://mirror.clarkson.edu/pub/dist...er/$basearch/os>   [updates-released]  name=Fedora Core $releasever - $basearch - Released Updates  baseurl=http://download.atrpms.net/mirrors/fedoracore/updates/$releasever/$basearch  <http://redhat.linux.ee/pub/fedora/l...sever/$basearch>  <http://fr2.rpmfind.net/linux/fedora...sever/$basearch>   [Fedora.us]  name=Fedora.us - $basearch - Extras  baseurl=http://fedora.linux.duke.edu/fedorax86\_64/fedora.us/$releasever/$basearch/RPMS.stable   [Dag Wieers]  name=Dag RPM Repository for Fedora Core  baseurl=http://apt.sw.be/fedora/$releasever/en/$basearch/dag   [Livna]  Name=Livna RPM, Fedora Core $releasever - $basearch  baseurl=http://rpm.livna.org/fedora/$releasever/$basearch/RPMS.stable   [freshrpms]  name=FreshRPMs  baseurl=http://ayo.freshrpms.net/fedora/linux/$releasever/$basearch/freshrpms/  <http://ftp.us2.freshrpms.net/linux/...arch/freshrpms/> |

**第一部分**(这是yum的全局性设置。默认一般不必改动。)  
  
[main]  
cachedir：yum缓存的目录，yum在此存储下载的rpm包和[**数据库**](http://space.itpub.net/25313300/viewspace-708509)，一般是/var/cache/yum。  
debuglevel：除错级别，0-10,默认是2。  
logfile：yum的日志文件，默认是/var/log/yum.log。  
pkgpolicy：包的策略。一共有两个选项，newest和last，这个作用是如果你设置了多个repository，而同一[**软件**](http://space.itpub.net/25313300/viewspace-708509)在不同的repository中同时存在，yum应该安装哪一个，如果是newest，则yum会安装最新的那个版本。如果是last，则yum会将服务器id以字母表排序，并选择最后的那个[**服务器**](http://space.itpub.net/25313300/viewspace-708509)上的软件安装。一般都是选newest。  
distroverpkg：指定一个软件包，yum会根据这个包判断你的发行版本，默认是redhat-release，也可以是安装的任何针对自己发行版的rpm包。  
exactarch，有两个选项1和0,代表是否只升级和你安装软件包cpu体系一致的包，如果设为1，则如你安装了一个i386的rpm，则yum不会用1686的包来升级。  
retries，网络连接发生错误后的重试次数，如果设为0，则会无限重试。  
tolerent，也有1和0两个选项，表示yum是否容忍命令行发生与软件包有关的错误，比如你要安装1,2,3三个包，而其中3此前已经安装了，如果你设为1,则yum不会出现错误信息。默认是0。  
  
除了上述之外，还有一些可以添加的选项，如  
  
exclude=，排除某些软件在升级名单之外，可以用通配符，列表中各个项目要用空格隔开，这个对于安装了诸如美化包，中文补丁的朋友特别有用。  
gpgchkeck= 有1和0两个选择，分别代表是否是否进行gpg校验，如果没有这一项，默认好像也是检查的。  
  
**第二部分：**  
  
配置repository服务器了，这是最令人激动的，有了好的repository，就如家门口开了大卖场，要什么东西稍微跑跑腿就行，对了这还是个免费的大卖场。  
所有服务器设置都应该遵循如下格式：  
  
1. [serverid]  
2. name=Some name for this [**server**](http://space.itpub.net/25313300/viewspace-708509)  
3. baseurl=url://path/to/repository/  
  
其中serverid是用于区别各个不同的repository，必须有一个独一无二的名称。  
name，是对repository的描述，支持像$releasever $basearch这样的变量;  
baseurl是服务器设置中最重要的部分，只有设置正确，才能从上面获取软件。它的格式是：  
  
1. baseurl=url://server1/path/to/repository/  
2. url://server2/path/to/repository/  
3. url://server3/path/to/repository/  
  
其中url支持的协议有 http:// ftp:// file://三种。baseurl后可以跟多个url，你可以自己改为速度比较快的镜像站，但baseurl只能有一个，也就是说不能像如下格式：  
  
1. baseurl=url://server1/path/to/repository/  
2. baseurl=url://server2/path/to/repository/  
3. baseurl=url://server3/path/to/repository/  
  
其中url指向的目录必须是这个repository header目录的上一级，它也支持$releasever $basearch这样的变量。  
url之后可以加上多个选项，如gpgcheck、exclude、failovermethod等，比如：  
  
1. [updates-released]  
2. name=Fedora Core $releasever - $basearch - Released Updates  
3. baseurl=http://download.atrpms.net/mirrors/fedoracore/updates/$releasever/$basearch  
4. <http://redhat.linux.ee/pub/fedora/linux/core/updates/>$releasever/$basearch  
5. <http://fr2.rpmfind.net/linux/fedora/core/updates/>$releasever/$basearch  
6. gpgcheck=1  
7. exclude=gaim  
8. failovermethod=priority  
  
其中gpgcheck，exclude的含义和[main]部分相同，但只对此服务器起作用，failovermethode 有两个选项roundrobin和priority，意思分别是有多个url可供选择时，yum选择的次序，roundrobin是随机选择，如果连接失 败则使用下一个，依次循环，priority则根据url的次序从第一个开始。如果不指明，默认是roundrobin。  
  
几个变量  
  
**$releasever**，发行版的版本，从[main]部分的distroverpkg获取，如果没有，则根据redhat-release包进行判断。  
**$arch**，cpu体系，如i686,athlon等  
**$basearch**，cpu的基本体系组，如i686和athlon同属i386，alpha和alphaev6同属alpha。  
  
对yum.conf设定完成，我们就可以好好体验yum带来的方便了。  
  
**五、RHEL/CENTOS的yum源**  
1、修改配置文件  
2、导入GPG KEY  
  
导入每个reposity的GPG key，前面说过，yum可以使用gpg对包进行校验，确保下载包的完整性，所以我们先要到各个repository站点找到gpg key，一般都会放在首页的醒目位置，一些名字诸如 RPM-GPG-KEY.txt之类的纯文本文件，把它们下载，然后用rpm --import xxx.txt命令将它们导入，最好把发行版自带GPG-KEY也导入。rpm --import /usr/share/doc/redhat-release-\*/RPM-GPG-KEY 官方软件升级用的上。  
  
**六、扩展你的rpm包  
1、rpmforge**  
Centos/RHEL默认的yum软件仓库非常有限,仅仅限于发行版本那几张盘里面的常规包和一些软件包的更新，利用RpmForge，可以增加非常多的第三方rpm软件包。  
  
**获取**  
<http://dag.wieers.com/rpm/packages/rpmforge-release/>  
找到与系统平台架构匹配的rpmforge包安装之即可。安装完后.默认会在系统的/etc/yum.repos.d/下新增两个文件,mirrors-rpmforge和rpmforge.repo  
  
**安装**  
# rpm -ivh rpmforge-release-0.3.6-1.el4.rf.i386.rpm  
# rpm --import <http://dag.wieers.com/rpm/packages/RPM-GPG-KEY.dag.txt>  
  
**2、加速yum更新速度**  
安装yum的fast mirror插件，可以加快 CentOS/RHEL yum的速度和提高稳定性，效果显著。  
centos5   
 #yum -y install yum-fastestmirror  
centos4  
 yum -y install yum-plugin-fastestmirror

**七、 yum使用**  
注:当第一次使用yum或yum资源库有更新时,yum会自动下载所有所需的headers放置于/var/cache/yum目录下,所需时间可能较长.

**系统更新(更新所有可以升级的rpm包,包括kernel)**  
#yum -y update

**每天定期执行系统更新**  
#chkconfig yum on  
#service yum start

**\*rpm包的更新，检查可更新的rpm包**  
#yum check-update

**更新所有的rpm包**  
#yum update

**更新指定的rpm包,如更新kernel和kernel source**  
#yum update kernel kernel-source

**大规模的版本升级,与yum update不同的是,连旧的淘汰的包也升级**  
#yum upgrade

**\*rpm包的安装和删除**  
安装rpm包,如xmms-mp3  
#yum install xmms-mp3

**删除rpm包,包括与该包有倚赖性的包**  
#yum remove licq  
注:同时会提示删除licq-gnome,licq-qt,licq-text

**\*yum暂存(/var/cache/yum/)的相关参数**  
清除暂存中rpm包文件  
#yum clean packages

**清除暂存中rpm头文件**  
#yum clean headers

**清除暂存中旧的rpm头文件**  
#yum clean oldheaders

**清除暂存中旧的rpm头文件和包文件**  
#yum clean 或#yum clean all  
注:相当于yum clean packages + yum clean oldheaders

**\*rpm包列表**  
列出资源库中所有可以安装或更新的rpm包  
#yum list

**列出资源库中特定的可以安装或更新以及已经安装的rpm包**  
#yum list mozilla  
#yum list mozilla\*  
注:可以在rpm包名中使用匹配符,如列出所有以mozilla开头的rpm包

**列出资源库中所有可以更新的rpm包**  
#yum list updates

**列出已经安装的所有的rpm包**  
#yum list installed

**列出已经安装的但是不包含在资源库中的rpm包**  
#yum list extras  
注:通过其它网站下载安装的rpm包  
\*rpm包信息显示(info参数同list)

**列出资源库中所有可以安装或更新的rpm包的信息**  
#yum info

**列出资源库中特定的可以安装或更新以及已经安装的rpm包的信息**  
#yum info mozilla  
#yum info mozilla\*  
注:可以在rpm包名中使用匹配符,如列出所有以mozilla开头的rpm包的信息

**列出资源库中所有可以更新的rpm包的信息**  
#yum info updates

**列出已经安装的所有的rpm包的信息**  
#yum info installed

**列出已经安装的但是不包含在资源库中的rpm包的信息**  
#yum info extras  
注:通过其它网站下载安装的rpm包的信息

**\*搜索rpm包**  
搜索匹配特定字符的rpm包  
#yum search mozilla  
注:在rpm包名,包描述等中搜索

**搜索有包含特定文件名的rpm包**  
#yum provides realplay

**八、yum常用问题解决**  
**1、如果网速慢的话可以通过增加yum的超时时间，这样就不会总是因为超时而退出。**  
#vi /etc/yum.conf  
#加上这么一句  
timeout=120

**2、yum Existing lock错误的解决办法**  
如果系统启动的时候， yum出现Existing lock /var/run/yum.pid: another copy is running as pid 3380. Aborting.可以用下面的办法解决：  
方法一  
etc/init.d/yum-updatesd stop

方法二  
#rm -f /var/run/yum.pid  
主要原因就是yum在自动更新，只要关掉它就可以了。

先安装fastestmirror，让系统yum的时候自动选择最快的镜像：

#yum install yum-fastestmirror -y

接下来说说RPMForge。

[RPMForge](http://rpmforge.net/)是一个安全、稳定并且更新及时的软件仓库，通过安装RPMForge可以是CentOS在进行yum update的时候多一个选择，有时候甚至是唯一的选择，比如想要升级rsync到3.0.x版本，如果懒得自己编译安装，就得依靠RPMForge了。

1、先安装priority这个yum的插件，这个插件用来设定安装软件时候软件仓库的优先级，一般是默认先从官方base或者镜像安装，然后从社区用户contribute的软件中安装，再从第三方软件仓库中安装。  
#yum install yum-priorities

2、手动编辑 /etc/yum.repos.d/ 目录中后缀为.repos的文件来设置软件仓库的优先级  
priority=N(N是整数，范围从1-99，数字越小代表优先级越高)

官方推荐配置是  
[base], [addons], [updates], [extras] 这几部分 priority=1  
[centosplus],[contrib] 这几部分priority=2  
Third Party Repos such as rpmforge … priority=N  (where N is > 10 and based on your preference)

3、现在开始安装rpmforge的软件仓库  
先下载rpmforge的安装包  
      i386  wget <http://apt.sw.be/redhat/el5/en/i386/RPMS.dag/rpmforge-release-0.3.6-1.el5.rf.i386.rpm>

      x86\_64 wget <http://apt.sw.be/redhat/el5/en/x86_64/RPMS.dag/rpmforge-release-0.3.6-1.el5.rf.x86_64.rpm>

安装

     rpm -i rpmforge-release-0.3.6-1.el5.rf.\*.rpm

更改 /etc/yum.repos.d/rpmforge.repo 配置文件，添加  
priority=1（或者2.3.4….）

现在就可以使用rpmforge这个软件仓库了。比如yum update rsync，嘿嘿，成功升级到3.0.5了。

# [yum命令不能使用的相关错误](http://blog.csdn.net/rachel_luo/article/details/8692744)

分类： [Linux 学习](http://blog.csdn.net/rachel_luo/article/category/1103154) 2013-03-19 17:18 4282人阅读 [评论](http://blog.csdn.net/rachel_luo/article/details/8692744#comments)(0) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/rachel_luo/article/details/8692744#report)

[yum](http://www.csdn.net/tag/yum)[EPEL](http://www.csdn.net/tag/EPEL)[Segmentation fault](http://www.csdn.net/tag/Segmentation%20fault)[DNS](http://www.csdn.net/tag/DNS)

yum (Yellowdog Updater Modified)是一个集与查找，安装，更新和删除程序的Linux软件。它运行在RPM包兼容的Linux发行版本上，如：RedHat, Fedora, SUSE, CentOS, Mandriva。

最近安装了CentOS5.5操作系统，在使用yum命令安装的时候出现如下错误：

一、yum 安装软件时，报错：No package XXX available.

[root@localhost ~]# yum -y install redis  
Loaded plugins: fastestmirror  
Loading mirror speeds from cached hostfile  
 \* addons: mirrors.163.com  
 \* base: mirrors.163.com  
 \* extras: mirrors.163.com  
 \* updates: mirrors.163.com  
Setting up Install Process  
No package redis available.  
Nothing to do

解决方法：

1）.先去更新一下yum仓库：  
#yum -y update

2）.如果还是不行，可能是你安装的时候包名不够详细，可以通过yum的查找参数来列出对应软件的相关包名，拿vim的安装为例：

# yum search vim  
Loaded plugins: fastestmirror  
Loading mirror speeds from cached hostfile  
\* base: mirror.hosting90.cz  
\* extras: mirror.hosting90.cz  
\* updates: mirror.hosting90.cz  
===================== Matched: vim ===================  
vim-X11.x86\_64 : The VIM version of the vi editor for the X Window System.  
vim-common.x86\_64 : The common files needed by any version of the VIM editor.  
vim-enhanced.x86\_64 : A version of the VIM editor which includes recent  
: enhancements.

vim-minimal.x86\_64 : A minimal version of the VIM editor.

找到包名后，执行以下命令即可：

#yum -y install vim-enhanced.x86\_64  
以上命令就能执行成功了。

其他程序也同样可以先进行yum查找，再通过匹配详细包名来进行安装，就不会出现上述错误了。

3）如果还是不行，应该是 EPEL未被启动，且指示的 rpm包不可用  
具体原因：使用yum搜索某些rpm包，找不到包是因为CentOS是RedHat企业版编译过来的，去掉了所有关于版权问题的东西。安装EPEL后可以 很好的解决这个问题。EPEL(Extra Packages for Enterprise Linux )即企业版Linux的扩展包，提供了很多可共Centos使用的组件，安装完这个以后基本常用的rpm都可以找到。  
下载与安装EPEL的rpm文件包  
1.下载：  
#wget http://mirrors.ustc.edu.cn/fedora/epel/5/i386/epel-release-5-4.noarch.rpm  
2.安装：  
#rpm -ivh epel-release-5-4.noarch.rpm  
安装完毕后，可以使用yum安装软件。

注意：

1. epel的版本要与centos的版本相对应，否则安装失败，即centos5版本要对应epel5版本。

2. rpm安装完后，rpm -ql  epel\*，确提示未安装，此时要导入rpm的签名信息

#rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-6 （root身份执行）

二、YUM 提示 Segmentation fault （段错误）  
本来用得好好的yum命令，突然不能用了，报出：Segmentation fault （段错误）。  
通过一番查找，原因终于找到：是有人在服务器上安装了新版本的zlib包，正巧我执行过”#yum clean all”命令。  
解决办法：  
#vi /etc/ld.so.conf，注释掉对应的行：/usr/local/zlib/lib  
#ldconfig  
#yum update  
yum完全恢复正常。

说明：yum使用的是老版本的 zlib 库。如果有用户再一次清除了 yum 缓存，那么再使用yum安装软件时依然会报段错误。

三、执行yum update，报如下错误：  
Loaded plugins: fastestmirror  
Loading mirror speeds from cached hostfile  
Could not retrieve mirrorlist http://mirrorlist.centos.org/?release=6&arch=x86\_64&repo=os error was  
14: PYCURL ERROR 6 - "Couldn't resolve host 'mirrorlist.centos.org'"

Error: Cannot find a valid baseurl for repo: base

原因是：DNS未配置好；  
解决方案：重新配置DNS；  
#vi /etc/resolv.conf  
编辑文件内容为：  
search localdomain  
nameserver 202.101.172.35  
nameserver 202.101.172.47    
保存并退出；  
#service network restart

重启网卡。

2、

# [哈佛大学凌晨4点半的景象---我的学习笔记寄语](http://blog.csdn.net/johnstrive/article/details/6203208)



在网上看到这篇文章，觉得很有感触，

或许这再一次印证任何人每一次的成功背后都有不为人知的付出和汗水。哈佛老师经常给学生这样的告诫：如果你想在进入社会后，在任何时候任何场合下都能得心应手并且得到应有的评价，那么你在哈佛的学习期间，就没有晒太阳的时间。

作为闻名于世的学府，哈佛大学培养了许多名人，他们中有33位诺贝尔奖获得者、7位美国总统以及各行各业的职业精英。究竟是什么使哈佛成为精英的摇篮？哈佛学子接受了什么样的精神和理念？这些问题吸引着成千上万的人去探知其中的答案。

哈佛图书馆墙上的20条训言似乎已经给出了答案。短短数语，引发深思，给人启迪。哈佛图书馆的二十条训言：

1.此刻打盹，你将做梦；而此刻学习，你将圆梦。

2.我荒废的今日，正是昨日殒身之人祈求的明日。

3.觉得为时已晚的时候，恰恰是最早的时候。

4.勿将今日之事拖到明日。

5.学习时的苦痛是暂时的，未学到的痛苦是终生的。

6.学习这件事，不是缺乏时间，而是缺乏努力。

7.幸福或许不排名次，但成功必排名次。

8.学习并不是人生的全部。但既然连人生的一部分―――学习也无法征服，还能做什么呢？

9.请享受无法回避的痛苦。

10.只有比别人更早、更勤奋地努力，才能尝到成功的滋味。

11.谁也不能随随便便成功，它来自彻底的自我管理和毅力。

12.时间在流逝。

13.现在流的口水，将成为明天的眼泪。

14.狗一样地学，绅士一样地玩。

15.今天不走，明天要跑。

16.投资未来的人，是忠于现实的人。

17.受教育程度代表收入。

18.一天过完，不会再来。

19.即使现在，对手也不停地翻动书页。

20.没有艰辛，便无所获。

此刻打盹，你将做梦；而此刻学习，你将圆梦

哈佛老师经常给学生这样的告诫：如果你想在进入社会后，在任何时候任何场合下都能得心应手并且得到应有的评价，那么你在哈佛的学习期间，就没有晒太阳的时间。在哈佛广为流传的一句格言是“忙完秋收忙秋种，学习，学习，再学习。”

　　人的时间和精力都是有限的，所以，要利用时间抓紧学习，而不是将所有的业余时间都用来打瞌睡。

　　有的人会这样说：“我只是在业余时间打盹而已，业余时间干吗把自己弄得那么紧张？”爱因斯坦就曾提出：“人的差异在于业余时间。”我的一位在哈佛任教的朋友也告诉我说，只要知道一个青年怎样度过他的业余时间，就能预言出这个青年的前程怎样。

　　20世纪初，在数学界有这样一道难题，那就是2的76次方减去1的结果是不是人们所猜想的质数。很多科学家都在努力地攻克这一数学难关，但结果并不如愿。1903年，在纽约的数学学会上，一位叫做科尔的科学家通过令人信服的运算论证，成功地证明了这道难题。

　　人们在惊诧和赞许之余，向科尔问道：“您论证这个课题一共花了多少时间？”科尔回答：“3年内的全部星期天。”

同样，加拿大医学教育家奥斯勒也是利用业余时间作出成就的典范。奥斯勒对人类最大的贡献，就是成功地研究了第三种血细胞。他为了从繁忙的工作中挤出时间读书，规定自己在睡觉之前必须读15分钟的书。不管忙碌到多晚，都坚持这一习惯不改变。这个习惯他整整坚持了半个世纪，共读了1000多本书，取得了令人瞩目的成绩。

·我荒废的今日，正是昨天殒身之人祈求的明日

　　闻名于世的约翰霍普金斯学院的创始人、牛津大学医学院的讲座教授、被英国国王册封为爵士的威廉。奥斯勒在年轻时，也曾为自己的前途感到迷茫。一次，他在读书时看到了一句话，给了他很大的启发。这句话是“最重要的就是不要去看远方模糊的事，而是做手边清楚的事。”

　　对此，哈佛提醒学生说“我荒废的今日，正是昨天殒身之人祈求的明日”。明天再美好，也不如抓住眼下的今天多做点实事。

　　获得哈佛大学荣誉学位的发明家、科学家本杰明。富兰克林有一次接到一个年轻人的求教电话，并与他约好了见面的时间和地点。当年轻人如约而至时，本杰明的房门大敞着，而眼前的房子里却乱七八糟、一片狼藉，年轻人很是意外。

　　没等他开口，本杰明就招呼道：“你看我这房间，太不整洁了，请你在门外等候一分钟，我收拾一下，你再进来吧。”然后本杰明就轻轻地关上了房门。

　　不到一分钟的时间，本杰明就又打开了房门，热情地把年轻人让进客厅。这时，年轻人的眼前展现出另一番景象―――房间内的一切已变得井然有序，而且有两杯倒好的红酒，在淡淡的香气里漾着微波。

　　年轻人在诧异中，还没有把满腹的有关人生和事业的疑难问题向本杰明讲出来，本杰明就非常客气地说道：“干杯！你可以走了。”

手持酒杯的年轻人一下子愣住了，带着一丝尴尬和遗憾说：“我还没向您请教呢……”

“这些……难道还不够吗？”本杰明一边微笑一边扫视着自己的房间说，“你进来又有一分钟了。”

　　“一分钟……”年轻人若有所思地说，“我懂了，您让我明白用一分钟的时间可以做许多事情，可以改变许多事情的深刻道理。”

　　珍惜眼前的每一分每一秒，也就珍惜了所拥有的今天。哈佛的这句话实际上揭示了一种人生哲学，那就是人生要以珍惜的态度把握时间，从今天开始，从现在做起。

·觉得为时已晚的时候，恰恰是最早的时候

安曼曾经是纽约港务局的工程师，工作多年后按规定退休。开始的时候，他很是失落。但他很快就高兴起来，因为他有了一个伟大的想法。他想创办一家自己的工程公司，要把办公楼开到全球各个角落。

安曼开始一步一个脚印地实施着自己的计划，设计的建筑遍布世界各地。在退休后的三十多年里，他实践着自己在工作中没有机会尝试的大胆和新奇的设计，不停地创造着一个又一个令世人瞩目的经典：埃塞俄比亚首都亚的斯亚贝巴机场，华盛顿杜勒斯机场，伊朗高速公路系统，宾夕法尼亚州匹兹堡市中心建筑群……这些作品被当作大学建筑系和工程系教科书上常用的范例，也是安曼伟大梦想的见证。86岁的时候，他完成最后一个作品―――当时世界上最长的悬体公路桥―――纽约 韦拉扎诺海峡桥。

　　生活中，很多事情都是这样，如果你愿意开始，认清目标，打定主意去做一件事，永远不会嫌晚。

·今天不走，明天要跑

在哈佛，教授们会时常提醒学生们要做好时间管理，并列举如下事例：

当今世界上最大的化学公司―――杜邦公司的总裁格劳福特。格林瓦特，每天挤出一小时来研究蜂鸟，并用专门的设备给蜂鸟拍照。权威人士把他写的关于蜂鸟的书称为自然历史丛书中的杰出作品。

　　休格。布莱克在进入美国议会前，并未受过高等教育。他从百忙中每天挤出一小时到国会图书馆去博览群书，包括政治、历史、哲学、诗歌等方面的书，数年如一日，就是在议会工作最忙的日子里也从未间断过。后来他成了美国最高法院的法官。

　 　一位名叫尼古拉的希腊籍电梯维修工对现代科学很感兴趣，他每天下班后到晚饭前，总要花一小时时间来攻读核物理学方面的书籍。随着知识的积累，一个念头跃入他的脑海。1948年，他提出了建立一种新型粒子加速器的计划。这种加速器比当时其他类型的加速器造价便宜而且更强有力。他把计划递交给美国原子能委员会做试验，又再经改进，这台加速器为美国节省了7000万美元。尼古拉得到了1万美元的奖励，还被聘请到加州大学放射实验室工作。

　　在人生的道路上，你停步不前，但有人却在拼命赶路。也许当你站立的时候，他还在你的后面向前追赶，但当你再一回望时，已看不到他的身影了，因为，他已经跑到你的前面了，现在需要你来追赶他了。所以，你不能停步，你要不断向前，不断超越。

·狗一样地学，绅士一样地玩

　　我们说要珍惜时间，努力为实现理想而打拼，但有一点要注意，那就是不要一味地拼命，也要有适度的休息和放松。对此，哈佛有个很贴切的说法，叫做“狗一样地学，绅士一样地玩”。话虽略显粗俗，但揭示的道理却很深刻。

　 　在哈佛，虽然学习强度很大，学生们承受着很大的学习压力，但他们也不提倡学生把所有的时间都用来学习。他们认为，学要尽力，玩也不能忽视。哈佛的学生也说，哈佛的课余生活要胜过正规学习。而哈佛也意识到适度的课外活动不但不会背离教育使命，而且还会给教育使命以支持。因此，他们提出要像“绅士一样地玩”。

　　在哈佛，学生们除了紧张地学习，还会参加学校组织的多种艺术活动，比如音乐会、戏剧演出、舞蹈表演及各种艺术展览等，此外，哈佛每年还会举办艺术节，以活跃学生的业余生活。这些充满着浓厚艺术氛围的活动不仅让学生接受了艺术教育和熏陶，而且提高了学生的艺术修养和审美能力。

　　哈佛的理念就是要求你在紧张的学习和工作后，能够暂时地完全忘记它们，像投入工作那样投入玩耍，尽情地放松。的确，在你尽心休闲的时候，所得到的体力和精力的恢复会为你下一阶段的奋斗增添无穷的动力。所以，在前进的路上，你不仅要勤奋努力，更要学会放松。

·现在流的口水，将成为明天的眼泪

成功与安逸是不可兼得的，选择了其一，就必定放弃了另一结局。正像哈佛所提醒的那样：现在流的口水，将成为明天的眼泪。今天不努力，明天必定遭罪。

我的邻居查尔斯曾经在哈佛度过4年的大学时光，他现在就职于纽约的一家软件公司，做他最擅长的行政管理工作,九九读书人。不久前，他的公司被一家法国公司兼并了。在兼并合同签订的当天，公司的新总裁宣布：“我们不会随意裁员，但如果你的法语太差，导致无法和其他员工交流，那么，不管是多高职位的人，我们都不得不请你离开。这个周末我们将进行一次法语考试，只有考试及格的人才能继续在这里工作。”

　　散会后，几乎所有的人都拥向了图书馆，他们这时才意识到要赶快补习法语了。只有查尔斯像平常一样直接回家了，同事们都认为他已经准备放弃这份工作了，毕竟，哈佛的学习背景和公司管理层的工作经验会帮助他轻而易举地找到另一份不错的工作。

然而，令所有人都想不到的是，考试结果出来后，这个在大家眼中没有希望的人却考了最高分。原来，查尔斯在毕业后来到这家公司后，他在工作中发现与法国人打交道的机会特别多，不会法语会使自己的工作受到很大的限制，所以，他很早就开始自学法语了。他利用可利用的一切时间，每天坚持学习，最终学有所获。

　　在哈佛，你从来看不到学生在偷懒，在消磨时间。当若干年后回想起曾经的梦想时，希望带给你的是无尽的欣慰笑容，而不是因蹉跎而流下的悔恨泪水。

·投资未来的人，是忠于现实的人

　　作为世界知名的学府，哈佛十分强调要有长远眼光，为未来投资。要投资未来，就要定好未来的投资方向，也就是要及早地设定人生目标。没有目标，就谈不到发展，更谈不上成功。

哈佛大学曾进行过这样一项跟踪调查，对象是一群在智力、学历和环境等方面条件差不多的年轻人。调查结果发现：27%的人没有目标；60%的人目标模糊；10%的人有着清晰但比较短期的目标；其余3%的人有着清晰而长远的目标.

以后的岁月，他们行进在各自的人生旅途中。25年后，哈佛再次对这群学生进行了跟踪调查。结果是这样的：

　 　3%的人，在25年间朝着一个方向不懈努力，几乎都成为社会各界的成功人士，其中不乏行业领袖和社会精英；10%的人，他们的短期目标不断地实现，成为各个领域中的专业人士，大都生活在社会的中上层；60%的人，他们安稳地生活与工作，但都没有什么特别成绩，几乎都生活在社会的中下层；剩下27%的人， 他们的生活没有目标，过得很不如意，并且常常在抱怨他人，抱怨社会，当然，也抱怨自己。

其实，他们之间的差别仅仅在于：25年前，他们中的一些人就已经知道自己最想要做的是什么，而另一些人则不清楚或不很清楚。这个调查生动地说明了明确生活目标对于人生成功的重要意义。