**Linux**

# 简介

创始人：**linus**，芬兰人。1991,10,5第一个版本！！

服务器端市场占有: **66%**,

**unix**, **windows server** 2000…….

有很多发型版本：red hat, debian,fedora,**ubantu**........

2个大的分支：

**Redhat 红帽子系列,**

**Debian 大便系列**

**都有商业版本和开源版本**

**redhat已经商业化了，他有一个一模一样的完全免费开源的社区版：centos**

**目前最新版本7.x，我们使用的是7.x**

## Centos

CentOS（Community Enterprise Operating System，中文意思是：社区企业操作系统）是Linux发行版之一，它是来自于Red Hat Enterprise Linux依照[开放源代码](http://baike.baidu.com/view/1708.htm)规定释出的源代码所编译而成。

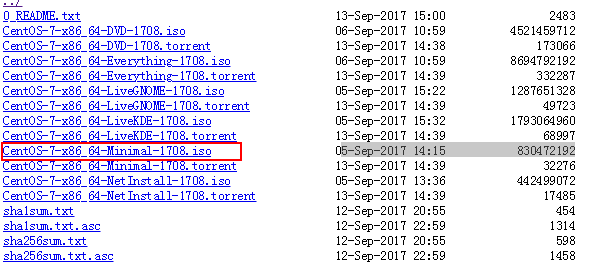
# 下载

官网下载：<https://www.centos.org/download/>



官网镜像表：<https://www.centos.org/download/mirrors/>

国内阿里镜像：<http://mirrors.aliyun.com/centos/7.4.1708/isos/x86_64/>



# 虚拟机

## VMware虚拟机

VMWare虚拟机软件是一个“虚拟[PC](http://baike.baidu.com/subview/13621/4882082.htm)”软件，它使你可以在一台机器上同时运行二个或更多[Windows](http://baike.baidu.com/view/4821.htm)、[DOS](http://baike.baidu.com/subview/365/7971327.htm)、[LINUX](http://baike.baidu.com/view/1634.htm)系统。与“多启动”系统相比，[VMWare](http://baike.baidu.com/view/301083.htm)采用了完全不同的概念。多启动系统在一个时刻只能运行一个系统，在系统切换时需要重新启动机器。

可以下载：

http://www.epinv.com/dl/6304.html

解压密码：www.epinv.com

## 安装简单

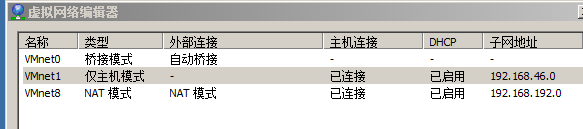
**注意：安装之前关闭所有的杀毒软件！！**

下一步就可以，可以输入序列号！！

许可证: **5A02H-AU243-TZJ49-GTC7K-3C61N**

## 安装后检查

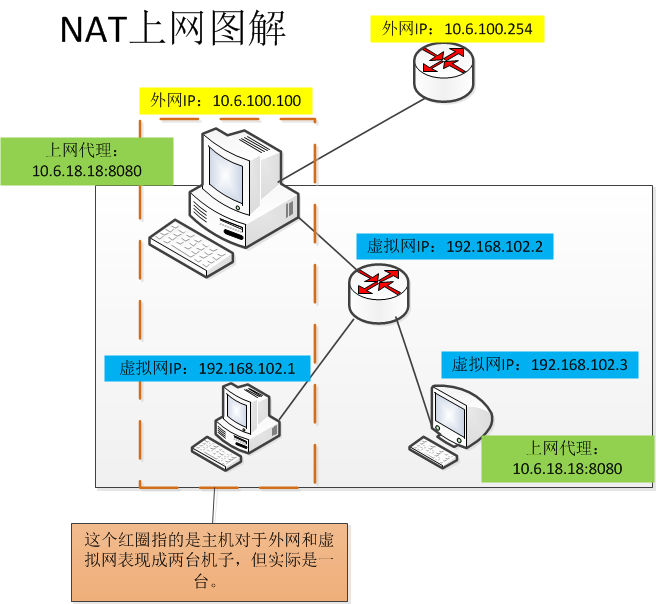




检查服务



## 了解一下虚拟网络nat模式



默认生成的2块网卡：



可以通过cmd里的命令查看所有的网络信息：



这块网卡1连接虚拟网络:



另一块网卡8是本机用来连接虚拟机的：



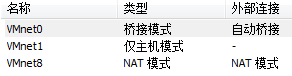
本机：就是你的真实的电脑，例如：windows7。

Vmnet8

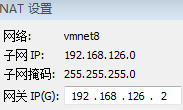
虚拟机

本机

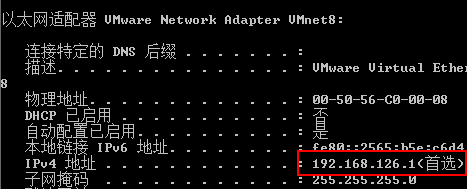
虚拟网络编辑器：



我们采用nat模式：



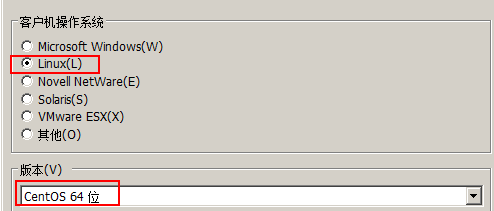
请记住这里的设置！！

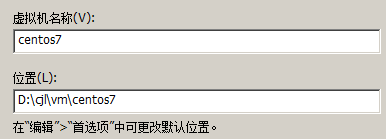


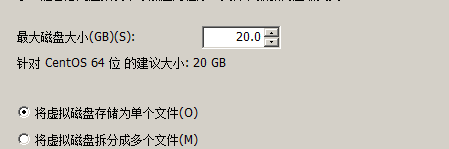
**要保证同一网段哦！！**

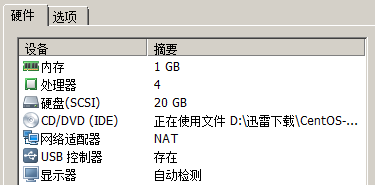
## 创建虚拟机

选择64位









**设置安装的光盘镜像文件：**

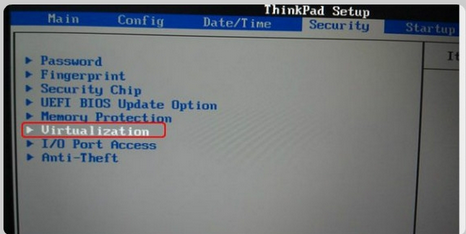
# 安装centos-7

**安装64位要开启虚拟化技术支持！！请注意！！**

需要重启电脑，在开机黑界面，按“F1”键，进入“BIOS”设置开启“虚拟化”技术！！

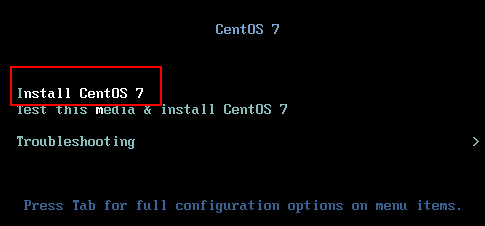
具体可百度，不同的机器，设置不同，前面的按键也不同！！

**Thinkpad电脑主板设置如下图：**





重启后进入如下界面：











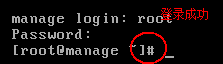






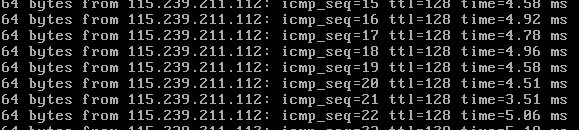


设置“root”密码不少于6位！



## 检查网络是否通畅

是否能连接外网：ping [www.baidu.com](http://www.baidu.com)



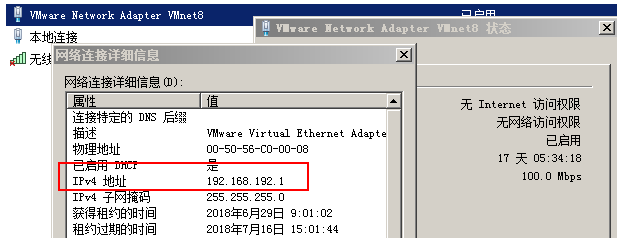
本机是否能访问虚拟机：



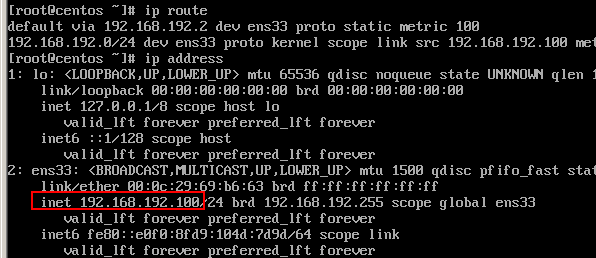
**注意：**如果本机ping 不通虚拟机，请检查这块

网卡的网段设置，是否和虚拟机在同一个网段里。

所谓网段，就是ip的前三个数字



检查虚拟机网络配置：





如果以上情况都正常，还连接不上的话，可能就是你的网络问题（你人为的不当操作引起的，看下面提醒！！）

**特别提醒：如果电脑上有多块网卡，请禁用其他网卡，还有请保持网卡为正常状态，不要有共享等特别的设置！！**

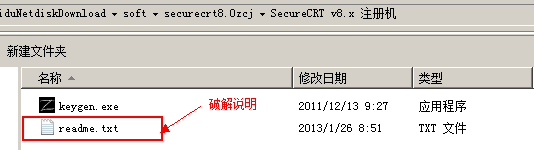
# Linux客户端工具secureCRT

## 下载软件、注册机



安装很简单，默认即可！！

## 按照说明文件破解

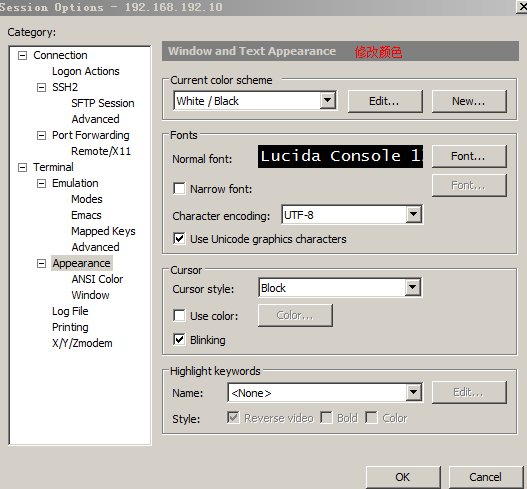


默认安装路径



## 破解后启动

可以根据自己喜好修改颜色



# Linux入门

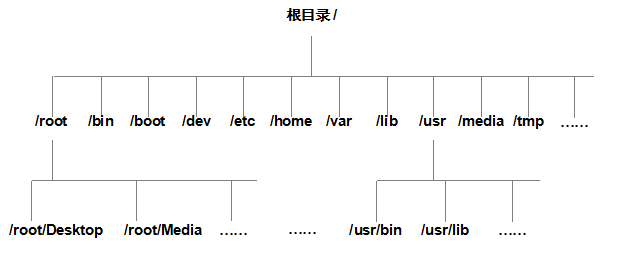
参考书籍：<<**循序渐进linux**>>，<<鸟哥的私房菜>>，《linux就该这么学》

在线网站：<https://www.linuxprobe.com/chapter-00.html>

**理解linux的文件系统的目录结构**（重点）

# 目录结构

在Linux系统中，目录、字符设备、块设备、套接字、打印机等都被抽象成了文件，一直强调的“**Linux系统中一切都是文件**”。



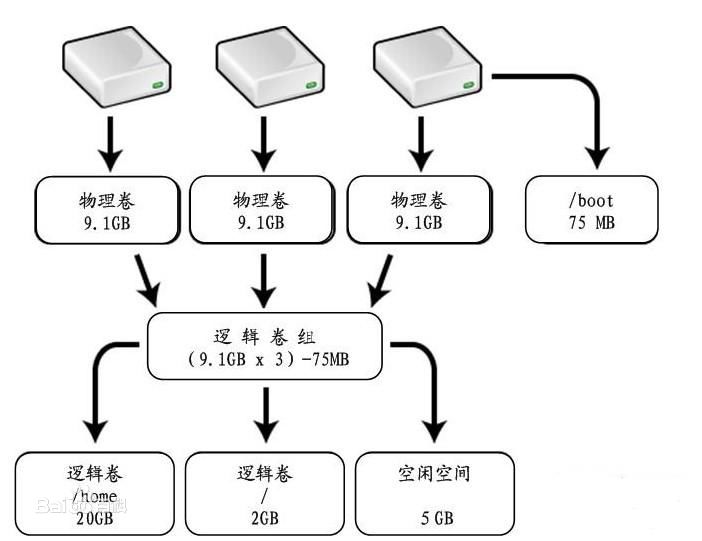
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Linux系统中常见的目录名称以及相应内容   |  |  | | --- | --- | | 目录名称 | 应放置文件的内容 | | /boot | 开机所需文件—内核、开机菜单以及所需配置文件等 | | /dev | 以文件形式存放任何设备与接口 | | /etc | 配置文件 | | /home | 用户主目录 | | /bin | 存放单用户模式下还可以操作的命令 | | /lib | 开机时用到的函数库，以及/bin与/sbin下面的命令要调用的函数 | | /sbin | 开机过程中需要的命令 | | /media | 用于挂载设备文件的目录 | | /opt | 放置第三方的软件 | | /root | 系统管理员的家目录 | | /srv | 一些网络服务的数据文件目录 | | /tmp | 任何人均可使用的“共享”临时目录 | | /proc | 虚拟文件系统，例如系统内核、进程、外部设备及网络状态等 | | /usr/local | 用户自行安装的软件 | | /usr/sbin | Linux系统开机时不会使用到的软件/命令/脚本 | | /usr/share | 帮助与说明文件，也可放置共享文件 | | /var | 主要存放经常变化的文件，如日志 | | /lost+found | 当文件系统发生错误时，将一些丢失的文件片段存放在这里 | |

# 路径

在Linux系统中另外还有一个重要的概念—**路径**。路径指的是如何定位到某个文件，分为**绝对路径**与**相对路径**。绝对路径指的是从根目录（/）开始写起的文件或目录名称，而相对路径则指的是相对于当前路径的写法。

# 分区

**LVM**是 Logical Volume Manager（逻辑[卷管理](https://baike.baidu.com/item/%E5%8D%B7%E7%AE%A1%E7%90%86)）的简写，它是Linux环境下对[磁盘分区](https://baike.baidu.com/item/%E7%A3%81%E7%9B%98%E5%88%86%E5%8C%BA)进行管理的一种机制，它由Heinz Mauelshagen在Linux 2.4[内核](https://baike.baidu.com/item/%E5%86%85%E6%A0%B8)上实现，目前最新版本为：稳定版1.0.5，开发版 1.1.0-rc2，以及LVM2开发版。Linux用户安装[Linux操作系统](https://baike.baidu.com/item/Linux%E6%93%8D%E4%BD%9C%E7%B3%BB%E7%BB%9F)时遇到的一个常见的难以决定的问题就是如何正确地评估各分区大小，以分配合适的硬盘空间。普通的磁盘分区管理方式在逻辑分区划分好之后就无法改变其大小，当一个逻辑分区存放不下某个文件时，这个文件因为受上层文件系统的限制，也不能跨越多个分区来存放，所以也不能同时放到别的磁盘上。而遇到出现某个分区空间耗尽时，解决的方法通常是使用[符号链接](https://baike.baidu.com/item/%E7%AC%A6%E5%8F%B7%E9%93%BE%E6%8E%A5)，或者使用调整分区大小的工具，但这只是暂时解决办法，没有从根本上解决问题。随着Linux的逻辑[卷管理](https://baike.baidu.com/item/%E5%8D%B7%E7%AE%A1%E7%90%86)功能的出现，这些问题都迎刃而解，用户在无需停机的情况下可以方便地调整各个分区大小。



# 文件系统

用户在硬件存储设备中执行的文件建立、写入、读取、修改、转存与控制等操作都是依靠文件系统来完成的。文件系统的作用是合理规划硬盘，以保证用户正常的使用需求。Linux系统支持数十种的文件系统，而最常见的文件系统如下所示。

**Ext3**：是一款日志文件系统，能够在系统异常宕机时避免文件系统资料丢失，并能自动修复数据的不一致与错误。然而，当硬盘容量较大时，所需的修复时间也会很长，而且也不能百分之百地保证资料不会丢失。它会把整个磁盘的每个写入动作的细节都预先记录下来，以便在发生异常宕机后能回溯追踪到被中断的部分，然后尝试进行修复。

**Ext4**：Ext3的改进版本，作为RHEL 6系统中的默认文件管理系统，它支持的存储容量高达1EB（1EB=1,073,741,824GB），且能够有无限多的子目录。另外，Ext4文件系统能够批量分配block块，从而极大地提高了读写效率。

**XFS**：**是一种高性能的日志文件系统，而且是RHEL 7中默认的文件管理系统**，它的优势在发生意外宕机后尤其明显，即可以快速地恢复可能被破坏的文件，而且强大的日志功能只用花费极低的计算和存储性能。并且它最大可支持的存储容量为18EB，这几乎满足了所有需求。

# 文件链接

在**Windows**系统中，快捷方式就是指向原始文件的一个**链接文件**，可以让用户从不同的位置来访问原始的文件；原文件一旦被删除或剪切到其他地方后，会导致链接文件失效。但是，这个看似简单的东西在Linux系统中可不太一样。

在**Linux**系统中存在硬链接和软连接两种文件。

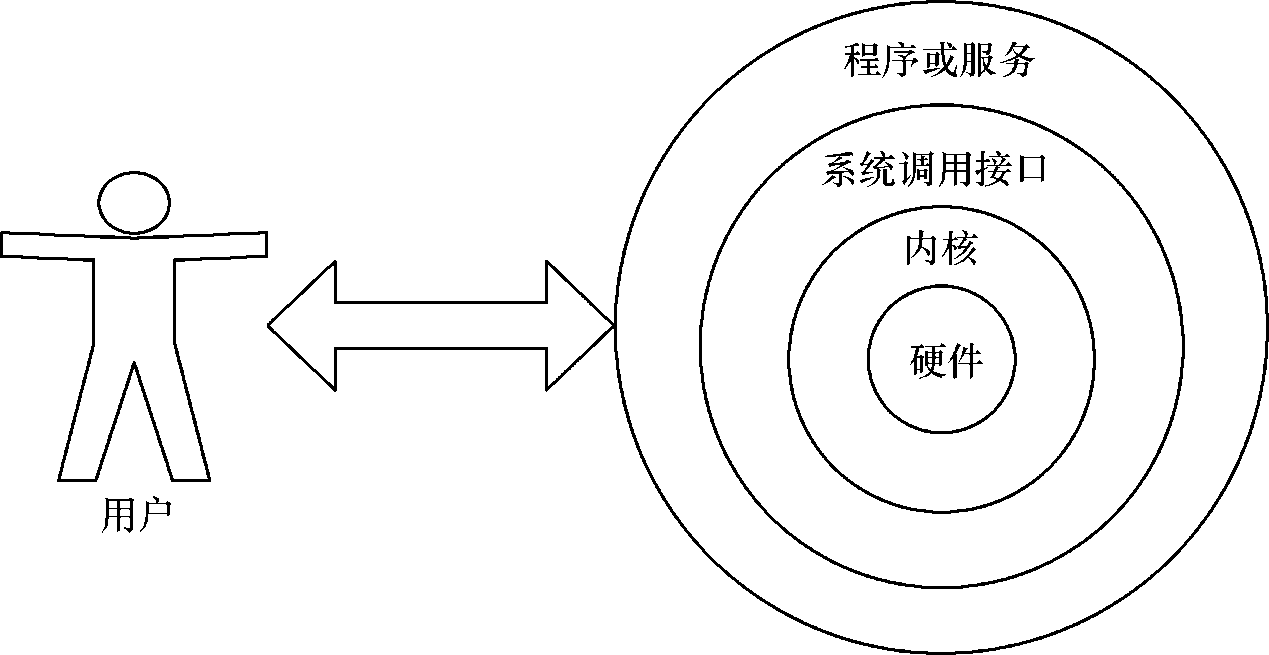
**硬链接（hard link）：**可以将它理解为一个“指向原始文件inode的指针”，系统不为它分配独立的inode和文件。所以，**硬链接文件与原始文件其实是同一个文件，只是名字不同**。我们每添加一个硬链接，该文件的inode连接数就会增加1；而且只有当该文件的inode连接数为0时，才算彻底将它删除。换言之，由于硬链接实际上是指向原文件inode的指针，因此即便原始文件被删除，依然可以通过硬链接文件来访问。需要注意的是，由于技术的局限性，**我们不能跨分区对目录文件进行链接**。

**软链接（也称为符号链接[symbolic link]）：**仅仅包含所链接文件的路径名，因此能链接目录文件，也可以跨越文件系统进行链接。但是，**当原始文件被删除后，链接文件也将失效**，从这一点上来说与Windows系统中的“快捷方式”具有一样的性质。

# Shell

通常来讲，计算机硬件是由运算器、控制器、存储器、输入/输出设备等共同组成的，而让各种硬件设备各司其职且又能协同运行的东西就是系统内核。[Linux系统](https://www.linuxprobe.com/)的内核负责完成对硬件资源的分配、调度等管理任务。由此可见，系统内核对计算机的正常运行来讲是太重要了，因此一般不建议直接去编辑内核中的参数，而是让用户通过基于系统调用接口开发出的程序或服务来管理计算机，以满足日常工作的需要，如图2-1所示。

必须肯定的是，Linux系统中有些图形化工具（比如逻辑卷管理器[Logical Volume Manager，LVM]）确实非常好用，极大地降低了运维人员操作出错的概率，值得称赞。但是，很多图形化工具其实是调用了脚本来完成相应的工作，往往只是为了完成某种工作而设计的，缺乏Linux命令原有的灵活性及可控性。再者，图形化工具相较于Linux命令行界面会更加消耗系统资源，因此经验丰富的运维人员甚至都不会给Linux系统安装图形界面，需要开始运维工作时直接通过命令行模式远程连接过去，不得不说这样做确实挺高效的。



Shell就是这样的一个工具。Shell（也称为终端或壳）充当的是人与内核（硬件）之间的翻译官，用户把一些命令“告诉”终端，它就会调用相应的程序服务去完成某些工作。现在包括[红帽](https://www.linuxprobe.com/)系统在内的许多主流Linux系统默认使用的终端是Bash（Bourne-Again SHell）解释器。主流Linux系统选择Bash解释器作为命令行终端主要有以下4项优势，读者可以在今后的学习和生产工作中细细体会Linux系统命令行的美妙之处，真正从心里爱上它们。

1：通过上下方向键来调取过往执行过的Linux命令；

2：命令或参数仅需输入前几位就可以用Tab键补全；

3：具有强大的批处理脚本；

4：具有实用的环境变量功能。

**Shell有图形化、字符2种类型！！**

# 命令组成

常见执行Linux命令的格式是这样的：

**命令名称 [命令参数] [命令对象]**

注意，**命令名称、命令参数、命令对象之间请用空格键分隔**。

命令对象一般是指要处理的文件、目录、用户等资源，而命令参数可以用**长格式**（完整的选项名称），也可以用**短格式**（单个字母的缩写），两者分别用**--与-**作为前缀。Linux新手不会执行命令大多是因为参数比较复杂，参数值需要随不同的命令和需求情况而发生改变。因此，要想灵活搭配各种参数，执行自己想要的功能，则需要长时间的经验积累了。

                                           命令参数的长格式与短格式示例

|  |  |
| --- | --- |
| 长格式 | man --help |
| 短格式 | man -h |

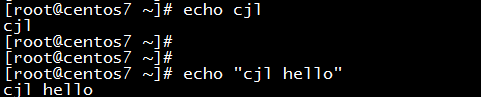
查询命令的网站：<http://man.linuxde.net>

## man(了解)

帮助学习一个命令的，由于文档都是英文的，而且太详细，不太适合新手去学习，了解即可！！

## Echo

echo命令用于在终端输出字符串或变量提取后的值，格式为“echo [字符串 | $变量]”。



下面，我们使用$变量的方式提取变量SHELL的值，并将其输出到屏幕上：



## date

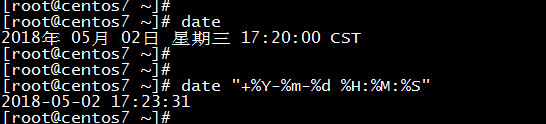
date命令用于显示及设置系统的时间或日期，格式为“date [选项] [+指定的格式]”。

只需在强大的date命令中输入以“+”号开头的参数，即可按照指定格式来输出系统的时间或日期，这样在日常工作时便可以把备份数据的命令与指定格式输出的时间信息结合到一起。例如，把打包后的文件自动按照“年-月-日”的格式打包成“backup-2017-9-1.tar.gz”，用户只需要看一眼文件名称就能大概了解到每个文件的备份时间了。date命令中常见的参数格式及作用如表2-4所示。

                                            date命令中的参数以及作用

|  |  |
| --- | --- |
| 参数 | 作用 |
| %t | 跳格[Tab键] |
| %H | 小时（00～23） |
| %I | 小时（00～12） |
| %M | 分钟（00～59） |
| %S | 秒（00～59） |
| %j | 今年中的第几天 |

按照默认格式查看当前系统时间的date命令如下所示：



## reboot

reboot命令用于重启系统，其格式为reboot。

由于重启计算机这种操作会涉及硬件资源的管理权限，因此默认只能使用root管理员来重启，其命令如下：

## poweroff

poweroff命令用于关闭系统，其格式为poweroff。

该命令与reboot命令相同，都会涉及硬件资源的管理权限，因此默认只有root管理员才可以关闭电脑，

## wget

wget命令用于在终端中下载网络文件，格式为“wget [参数] 下载地址”。

如果您没有Linux系统的管理经验，当前只需了解一下wget命令的参数以及作用，然后看一下下面的演示实验即可，切记不要急于求成。后面章节将逐步讲解Linux系统的配置管理方法，可以等您掌握了网卡的配置方法后再来进行这个实验操作。表2-5所示为wget命令的参数以及参数的作用。

表2-5                                                wget命令的参数以及作用

|  |  |
| --- | --- |
| 参数 | 作用 |
| -b | 后台下载模式 |
| -P | 下载到指定目录 |
| -t | 最大尝试次数 |
| -c | 断点续传 |
| -p | 下载页面内所有资源，包括图片、视频等 |
| -r | 递归下载 |

后面会有机会演示！！

## ps

ps命令用于查看系统中的进程状态，格式为“ps [参数]”。

                                             ps命令的参数以及作用

|  |  |
| --- | --- |
| 参数 | 作用 |
| -a | 显示所有进程（包括其他用户的进程） |
| -u | 用户以及其他详细信息 |
| -x | 显示没有控制终端的进程 |

Linux系统中时刻运行着许多进程，如果能够合理地管理它们，则可以优化系统的性能。在Linux系统中，有5种常见的进程状态，分别为运行、中断、不可中断、僵死与停止，其各自含义如下所示。

**R（运行）：**进程正在运行或在运行队列中等待。

**S（中断）：**进程处于休眠中，当某个条件形成后或者接收到信号时，则脱离该   状态。

**D（不可中断）：**进程不响应系统异步信号，即便用kill命令也不能将其中断。

**Z（僵死）：**进程已经终止，但进程描述符依然存在, 直到父进程调用wait4()系统函数后将进程释放。

**T（停止）：**进程收到停止信号后停止运行。

当执行**ps aux**命令后通常会看到如表2-7所示的进程状态，表2-7中只是列举了部分输出值，而且正常的输出值中不包括中文注释。

表2-7                                                             进程状态

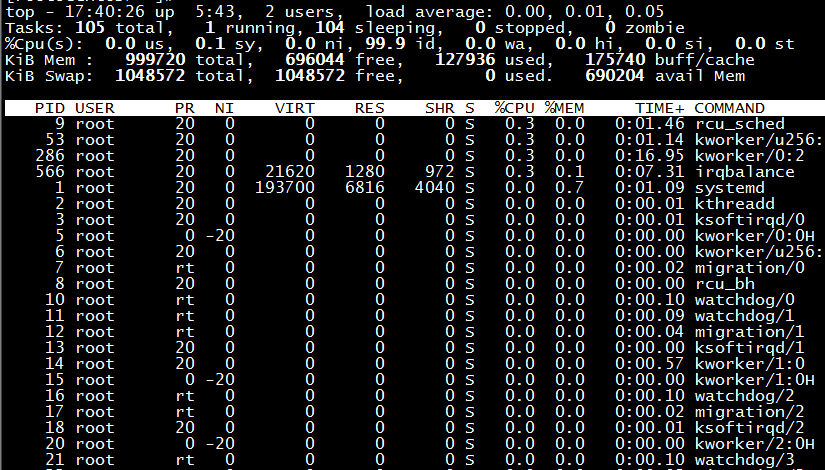
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| USER | PID | %CPU | %MEM | VSZ | RSS | TTY | STAT | START | TIME | COMMAND |
| 进程的所有者 | 进程ID号 | 运算器占用率 | 内存占用率 | 虚拟内存使用量(单位是KB) | 占用的固定内存量(单位是KB) | 所在终端 | 进程状态 | 被启动的时间 | 实际使用CPU的时间 | 命令名称与参数 |
| root | 1 | 0.0 | 0.4 | 53684 | 7628 | ? | Ss | 07:22 | 0:02 | /usr/lib/systemd/systemd |
| root | 2 | 0.0 | 0.0 | 0 | 0 | ? | S | 07:22 | 0:00 | [kthreadd] |
| root | 3 | 0.0 | 0.0 | 0 | 0 | ? | S | 07:22 | 0:00 | [ksoftirqd/0] |
| root | 5 | 0.0 | 0.0 | 0 | 0 | ? | S< | 07:22 | 0:00 | [kworker/0:0H] |
| root | 7 | 0.0 | 0.0 | 0 | 0 | ? | S | 07:22 | 0:00 | [migration/0] |
|  | | | | ……………… | 省略部分输出信息 | ……………… | | | | |

如前面所提到的，在Linux系统中的命令参数有长短格式之分，长格式和长格式之间不能合并，长格式和短格式之间也不能合并，但短格式和短格式之间是可以合并的，合并后仅保留一个-（减号）即可。另外ps命令可允许参数不加减号（-），因此可直接写成ps aux的样子。

## top

top命令用于动态地监视进程活动与系统负载等信息，其格式为top。

top命令相当强大，能够动态地查看系统运维状态，完全将它看作Linux中的“强化版的Windows任务管理器”。top命令的运行界面如图2-6所示。



在图2-5中，top命令执行结果的前5行为系统整体的统计信息，其所代表的含义如下。

第1行：系统时间、运行时间、登录终端数、系统负载（三个数值分别为1分钟、5分钟、15分钟内的平均值，数值越小意味着负载越低）。

第2行：进程总数、运行中的进程数、睡眠中的进程数、停止的进程数、僵死的进程数。

第3行：用户占用资源百分比、系统内核占用资源百分比、改变过优先级的进程资源百分比、空闲的资源百分比等。其中数据均为CPU数据并以百分比格式显示，例如“97.1 id”意味着有97.1%的CPU处理器资源处于空闲。

第4行：物理内存总量、内存使用量、内存空闲量、作为内核缓存的内存量。

第5行：虚拟内存总量、虚拟内存使用量、虚拟内存空闲量、已被提前加载的内存量。

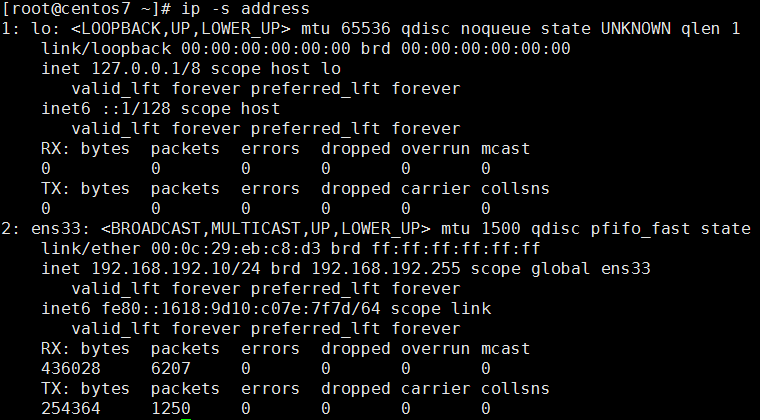
## kill

kill命令用于终止某个指定PID的服务进程，格式为“**kill [参数] [进程PID]**”。

## killall

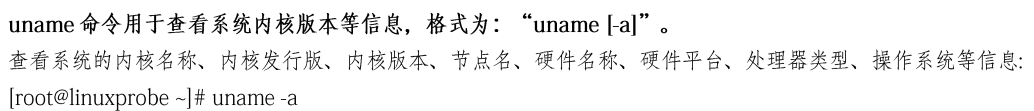
killall命令用于终止某个指定名称的服务所对应的全部进程，格式为：“killall [参数] [进程名称]”。

## ip





## uname



uname命令用于查看系统内核与系统版本等信息，格式为“uname [-a]”。

在使用uname命令时，一般会固定搭配上-a参数来完整地查看当前系统的内核名称、主机名、内核发行版本、节点名、系统时间、硬件名称、硬件平台、处理器类型以及操作系统名称等信息。



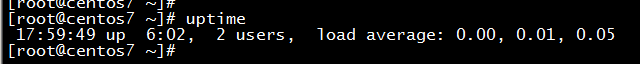
顺带一提，如果要查看当前系统版本的详细信息，则需要查看redhat-release文件，其命令以及相应的结果如下：



## uptime

uptime用于查看系统的负载信息，格式为uptime。

uptime命令真的很棒，它可以显示当前系统时间、系统已运行时间、启用终端数量以及平均负载值等信息。平均负载值指的是系统在最近1分钟、5分钟、15分钟内的压力情况（下面加粗的信息部分）；负载值越低越好，尽量不要长期超过1，在生产环境中不要超过5。



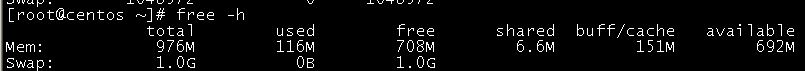
## free

free用于显示当前系统中内存的使用量信息，格式为“free [-h]”。

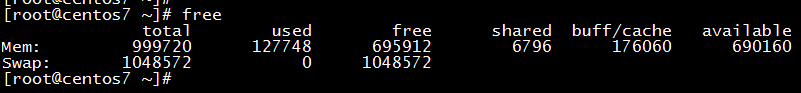
为了保证Linux系统不会因资源耗尽而突然宕机，运维人员需要时刻关注内存的使用量。在使用free命令时，可以结合使用-h参数以更人性化的方式输出当前内存的实时使用量信息。需要注意的是，输出信息中的中文注释是作者自行添加的内容，实际输出时没有相应的参数解释。

[root@linuxprobe ~]# free -h

执行free -h命令后的输出信息



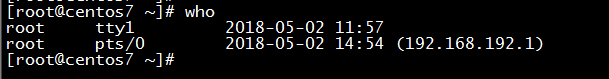
内存总量 已用量 可用量 进程共享的内存量 磁盘缓存的内存量 缓存的内存量



## who

who用于查看当前登入主机的用户终端信息，格式为“who [参数]”。

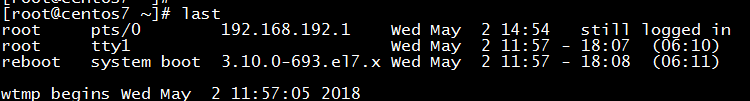
这三个简单的字母可以快速显示出所有正在登录本机的用户的名称以及他们正在开启的终端信息。表2-9所示为执行who命令后的结果。



## last

last命令用于查看所有系统的登录记录，格式为“last [参数]”。

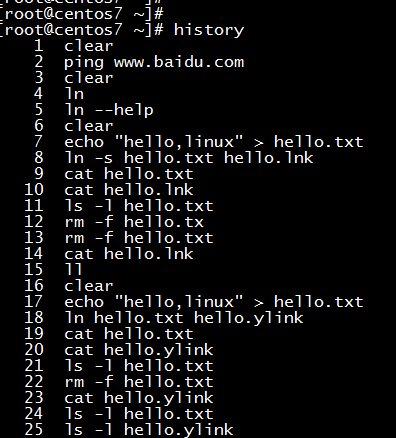
使用last命令可以查看本机的登录记录。但是，由于这些信息都是以日志文件的形式保存在系统中，因此黑客可以很容易地对内容进行篡改。千万不要单纯以该命令的输出信息而判断系统有无被恶意入侵！



## history

history命令用于显示历史执行过的命令，格式为“history [-c]”。

history命令应该是作者最喜欢的命令。执行history命令能显示出当前用户在本地计算机中执行过的最近1000条命令记录。如果觉得1000不够用，还可以自定义/etc/profile文件中的HISTSIZE变量值。在使用history命令时，如果使用-c参数则会清空所有的命令历史记录。还可以使用“!编码数字”的方式来重复执行某一次的命令。总之，history命令有很多有趣的玩法等待您去开发。



## sosreport

sosreport命令用于收集系统配置及架构信息并输出诊断文档，格式为sosreport。

当Linux系统出现故障需要联系技术支持人员时，大多数时候都要先使用这个命令来简单收集系统的运行状态和服务配置信息，以便让技术支持人员能够远程解决一些小问题，亦或让他们能提前了解某些复杂问题。在下面的输出信息中，加粗的部分是收集好的资料压缩文件以及校验码，将其发送给技术支持人员即可。**目前了解即可！！**

# 文件操作

## pwd

pwd命令用于显示用户当前所处的工作目录，格式为“pwd [选项]”。



## cd

cd命令用于切换工作路径，格式为“cd [目录名称]”。

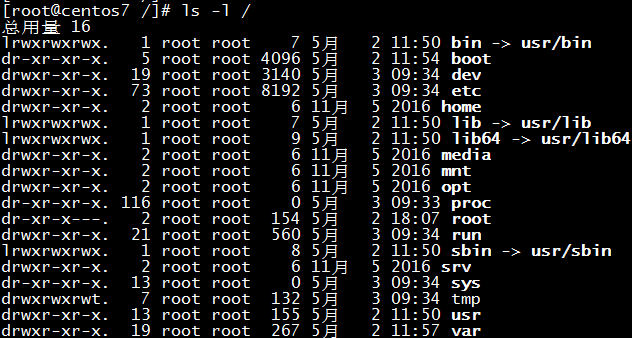
这个命令应该是最常用的一个Linux命令了。可以通过cd命令迅速、灵活地切换到不同的工作目录。除了常见的切换目录方式，还可以使用“cd -”命令返回到上一次所处的目录，使用“cd..”命令进入上级目录，以及使用“cd ~”命令切换到当前用户的家目录，亦或使用“cd ~username”切换到其他用户的家目录。

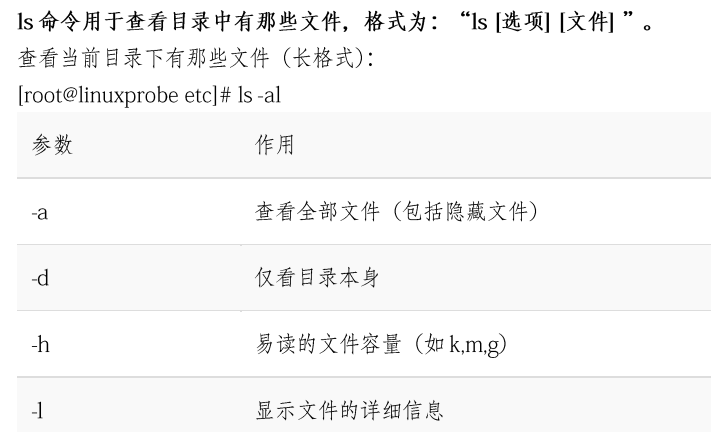


## ls

ls命令用于显示目录中的文件信息，格式为“ls [选项] [文件] ”。

所处的工作目录不同，当前工作目录下的文件肯定也不同。使用ls命令的“-a”参数看到全部文件（包括隐藏文件），使用“-l”参数可以查看文件的属性、大小等详细信息。将这两个参数整合之后，再执行ls命令即可查看当前目录中的所有文件并输出这些文件的属性信息：





如果想要查看目录属性信息，则需要额外添加一个-d参数。例如，可使用如下命令查看/etc目录的权限与属性信息：

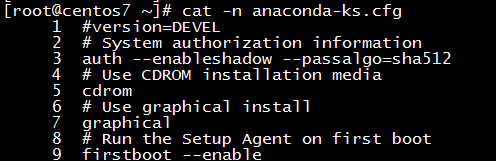


## cat

cat命令用于查看纯文本文件（内容较少的），格式为“cat [选项] [文件]”。

Linux系统中有多个用于查看文本内容的命令，每个命令都有自己的特点，比如这个cat命令就是用于查看内容较少的纯文本文件的。cat这个命令也很好记，因为cat在英语中是“猫”的意思，小猫咪是不是给您一种娇小、可爱的感觉呢？

如果在查看文本内容时还想顺便显示行号的话，不妨在cat命令后面追加一个-n参数：



## more

more命令用于查看纯文本文件（内容较多的），格式为“more [选项]文件”。

如果需要阅读长篇小说或者非常长的配置文件，那么“小猫咪”可就真的不适合了。因为一旦使用cat命令阅读长篇的文本内容，信息就会在屏幕上快速翻滚，导致自己还没有来得及看到，内容就已经翻篇了。因此对于长篇的文本内容，推荐使用more命令来查看。more命令会在最下面使用百分比的形式来提示您已经阅读了多少内容。您还可以使用**空格键**或**回车键**向下翻页：

向下翻页：空格

向下翻行：回车

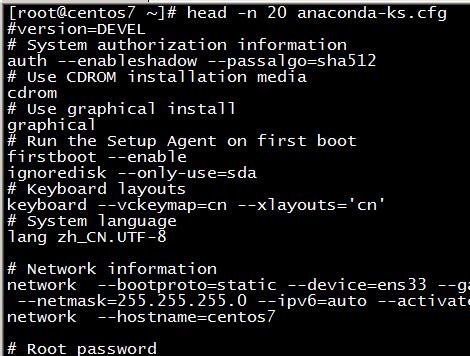
向前翻页：shift+PgUp

向上翻一行：shift + 向上的箭头（或者b键）

## head

head命令用于查看纯文本文档的前N行，格式为“head [选项] [文件]”。

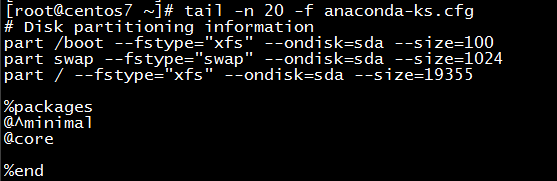
在阅读文本内容时，谁也难以保证会按照从头到尾的顺序往下看完整个文件。如果只想查看文本中前20行的内容，该怎么办呢？head命令可以派上用场了：



## tail

tail命令用于查看纯文本文档的后N行或持续刷新内容，格式为“tail [选项] [文件]”。

我们可能还会遇到另外一种情况，比如需要查看文本内容的最后20行，这时就需要用到tail命令了。tail命令的操作方法与head命令非常相似，只需要执行“tail -n 20 文件名”命令就可以达到这样的效果。**tail命令最强悍的功能是可以持续刷新一个文件的内容，当想要实时查看最新日志文件时，这特别有用，此时的命令格式为“tail -f 文件名”**：



## tr

**tr命令用于替换文本文件中的字符，格式为“tr [原始字符] [目标字符]”。**

在很多时候，我们想要快速地替换文本中的一些词汇，又或者把整个文本内容都进行替换，如果进行手工替换，难免工作量太大，尤其是需要处理大批量的内容时，进行手工替换更是不现实。这时，就可以先使用cat命令读取待处理的文本，然后通过管道符（详见第3章）把这些文本内容传递给tr命令进行替换操作即可。例如，把某个文本内容中的英文全部替换为大写：



## wc

wc命令用于统计指定文本的行数、字数、字节数，格式为“wc [参数] 文本”。

Linux系统中的wc命令用于统计文本的行数、字数、字节数等。如果为了方便自己记住这个命令的作用，也可以联想到上厕所时好无聊，无聊到数完了手中的如厕读物上有多少行字。wc的参数以及相应的作用如表2-10所示。

表2-10                                                   wc的参数以及作用

|  |  |
| --- | --- |
| 参数 | 作用 |
| -l | 只显示行数 |
| -w | 只显示单词数 |
| -c | 只显示字节数 |



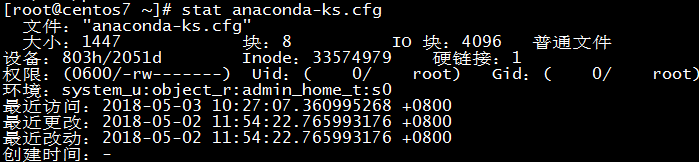
在Linux系统中，passwd是用于保存系统账户信息的文件，要统计当前系统中有多少个用户，可以使用下面的命令来进行查询，是不是很神奇：



## stat

stat命令用于查看文件的具体存储信息和时间等信息，格式为“stat 文件名称”。

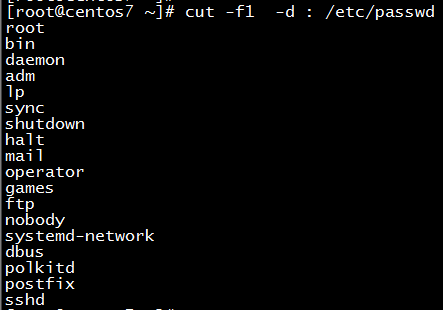
stat命令可以用于查看文件的存储信息和时间等信息，命令stat anaconda-ks.cfg会显示出文件的三种时间状态（已加粗）：Access、Modify、Change。这三种时间的区别将在下面的**touch**命令中详细详解：



## cut

cut命令用于按“列”提取文本字符，格式为“cut [参数] 文本”。

在Linux系统中，如何准确地提取出最想要的数据，这也是我们应该重点学习的内容。一般而言，按基于“行”的方式来提取数据是比较简单的，只需要设置好要搜索的关键词即可。但是如果按列搜索，不仅要使用-f参数来设置需要看的列数，还需要使用-d参数来设置间隔符号。passwd在保存用户数据信息时，用户信息的每一项值之间是采用冒号来间隔的，接下来我们使用下述命令尝试提取出passwd文件中的用户名信息，即提取以冒号（：）为间隔符号的第一列内容：



## diff

diff命令用于比较多个文本文件的差异，格式为“diff [参数] 文件”。

在使用diff命令时，不仅可以使用--brief参数来确认两个文件是否不同，还可以使用-c参数来详细比较出多个文件的差异之处，这**绝对是判断文件是否被篡改的有力神器**。例如，先使用cat命令分别查看diff\_A.txt和diff\_B.txt文件的内容，然后进行比较：

后面再说，了解即可

## touch

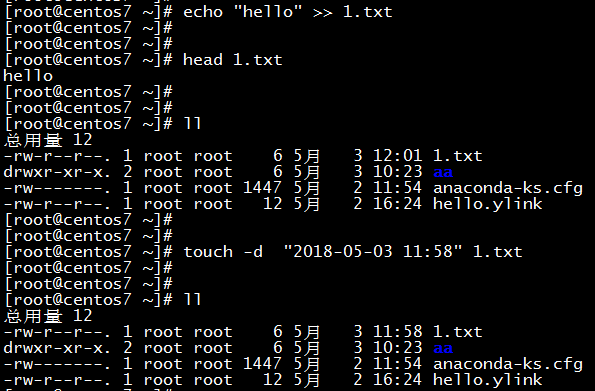
touch命令用于创建空白文件或设置文件的时间，格式为“touch [选项] [文件]”。

在创建空白的文本文件方面，这个touch命令相当简捷，简捷到没有必要铺开去讲。比如，touch linuxprobe命令可以创建出一个名为linuxprobe的空白文本文件。对touch命令来讲，有难度的操作主要是体现在设置**文件内容**的修改时间（mtime）、**文件权限或属性**的更改时间（ctime）与文件的**读取时间**（atime）上面。touch命令的参数及其作用如表2-11所示。

表2-11                                             touch命令的参数及其作用

|  |  |
| --- | --- |
| 参数 | 作用 |
| -a | 仅修改“读取时间”（atime） |
| -m | 仅修改“修改时间”（mtime） |
| -d | 同时修改atime与mtime |

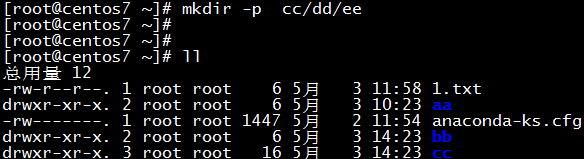
接下来，我们先使用ls命令查看一个文件的修改时间，然后修改这个文件，最后再通过touch命令把修改后的文件时间设置成修改之前的时间（**很多黑客就是这样做的呢**）：



## mkdir

mkdir命令用于创建空白的目录，格式为“mkdir [选项] 目录”。

在Linux系统中，文件夹是最常见的文件类型之一。除了能创建单个空白目录外，mkdir命令还可以结合-p参数来递归创建出具有嵌套叠层关系的文件目录。



## cp

cp命令用于复制文件或目录，格式为“cp [选项] 源文件 目标文件”。大家对文件复制操作应该不陌生，在Linux系统中，复制操作具体分为3种情况：

**如果目标文件是目录，则会把源文件复制到该目录中；**

**如果目标文件也是普通文件，则会询问是否要覆盖它；**

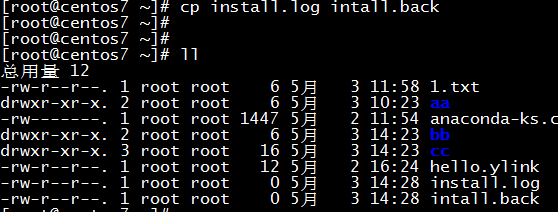
**如果目标文件不存在，则执行正常的复制操作。**

cp命令的参数及其作用如表2-12所示。

表2-12                                               cp命令的参数及其作用

|  |  |
| --- | --- |
| 参数 | 作用 |
| -p | 保留原始文件的属性 |
| -d | 若对象为“链接文件”，则保留该“链接文件”的属性 |
| -r | 递归持续复制（用于目录） |
| -i | 若目标文件存在则询问是否覆盖 |
| -a | 相当于-pdr（p、d、r为上述参数） |

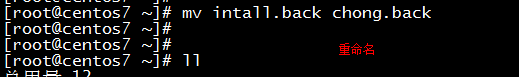
接下来，使用touch创建一个名为install.log的普通空白文件，然后将其复制为一份名为x.log的备份文件，最后再使用ls命令查看目录中的文件：



## mv

mv命令用于剪切文件或将文件重命名，格式为“mv [选项] 源文件 [目标路径|目标文件名]”。

剪切操作不同于复制操作，因为它会默认把源文件删除掉，只保留剪切后的文件。如果在同一个目录中对一个文件进行剪切操作，其实也就是对其进行重命名：



## rm

目录

[1 简介 1](#_Toc5089664)

[1.1 Centos 1](#_Toc5089665)

[2 下载 1](#_Toc5089666)

[3 虚拟机 2](#_Toc5089667)

[3.1 VMware虚拟机 2](#_Toc5089668)

[3.2 安装简单 2](#_Toc5089669)

[3.3 安装后检查 3](#_Toc5089670)

[3.4 了解一下虚拟网络nat模式 4](#_Toc5089671)

[3.5 创建虚拟机 5](#_Toc5089672)

[4 安装centos-7 6](#_Toc5089673)

[4.1 检查网络是否通畅 11](#_Toc5089674)

[5 Linux客户端工具secureCRT 13](#_Toc5089675)

[5.1 下载软件、注册机 13](#_Toc5089676)

[5.2 按照说明文件破解 13](#_Toc5089677)

[5.3 破解后启动 13](#_Toc5089678)

[6 Linux入门 14](#_Toc5089679)

[7 目录结构 14](#_Toc5089680)

[8 路径 17](#_Toc5089681)

[9 分区 17](#_Toc5089682)

[10 文件系统 18](#_Toc5089683)

[11 文件链接 18](#_Toc5089684)

[12 Shell 19](#_Toc5089685)

[13 命令组成 20](#_Toc5089686)

[13.1 man(了解) 20](#_Toc5089687)

[13.2 Echo 20](#_Toc5089688)

[13.3 date 21](#_Toc5089689)

[13.4 reboot 22](#_Toc5089690)

[13.5 poweroff 22](#_Toc5089691)

[13.6 wget 22](#_Toc5089692)

[13.7 ps 23](#_Toc5089693)

[13.8 top 24](#_Toc5089694)

[13.9 kill 25](#_Toc5089695)

[13.10 killall 26](#_Toc5089696)

[13.11 ip 26](#_Toc5089697)

[13.12 uname 26](#_Toc5089698)

[13.13 uptime 27](#_Toc5089699)

[13.14 free 27](#_Toc5089700)

[13.15 who 27](#_Toc5089701)

[13.16 last 28](#_Toc5089702)

[13.17 history 28](#_Toc5089703)

[13.18 sosreport 29](#_Toc5089704)

[14 文件操作 29](#_Toc5089705)

[14.1 pwd 29](#_Toc5089706)

[14.2 cd 30](#_Toc5089707)

[14.3 ls 30](#_Toc5089708)

[14.4 cat 31](#_Toc5089709)

[14.5 more 32](#_Toc5089710)

[14.6 head 32](#_Toc5089711)

[14.7 tail 33](#_Toc5089712)

[14.8 tr 33](#_Toc5089713)

[14.9 wc 34](#_Toc5089714)

[14.10 stat 34](#_Toc5089715)

[14.11 cut 35](#_Toc5089716)

[14.12 diff 35](#_Toc5089717)

[14.13 touch 36](#_Toc5089718)

[14.14 mkdir 37](#_Toc5089719)

[14.15 cp 37](#_Toc5089720)

[14.16 mv 38](#_Toc5089721)

[14.17 rm 38](#_Toc5089722)

[14.18 40](#_Toc5089723)

[14.19 file 41](#_Toc5089724)

[14.20 tar 41](#_Toc5089725)

[14.21 grep 43](#_Toc5089726)

[14.22 find 44](#_Toc5089727)

[14.23 ss 45](#_Toc5089728)

[14.24 useradd 46](#_Toc5089729)

[14.25 id 46](#_Toc5089730)

[14.26 passwd 47](#_Toc5089731)

[14.27 groupadd 47](#_Toc5089732)

[14.28 su 47](#_Toc5089733)

[14.29 ln 47](#_Toc5089734)

[15 管道符、重定向与环境变量 49](#_Toc5089735)

[15.1 管道符 49](#_Toc5089736)

[15.2 输入输出重定向 50](#_Toc5089737)

[15.3 命令行通配符 51](#_Toc5089738)

[15.4 重要的环境变量 52](#_Toc5089739)

[16 Vim编辑器 54](#_Toc5089740)

[16.1 修改主机名 56](#_Toc5089741)

[16.2 修改网卡 57](#_Toc5089742)

[16.3 配置Yum仓库 58](#_Toc5089743)

[17 Shell编程(了解) 61](#_Toc5089744)

[18 计划任务服务程序 66](#_Toc5089745)

[19 用户身份 69](#_Toc5089746)

[20 权限 70](#_Toc5089747)

[21 su命令与sudo服务 72](#_Toc5089748)

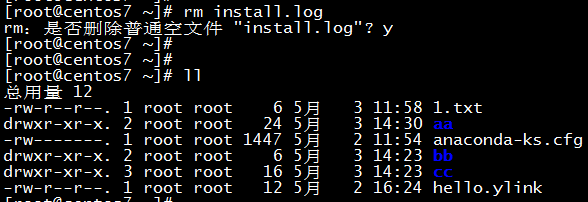
[22 防火墙 77](#_Toc5089749)

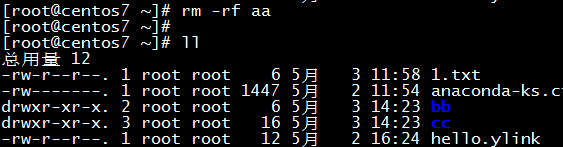
[23 关于开机自启动的说明 84](#_Toc5089750)

## 

rm命令用于删除文件或目录，格式为“rm [选项] 文件”。

在Linux系统中删除文件时，系统会默认向您询问是否要执行删除操作，如果不想总是看到这种反复的确认信息，可在rm命令后跟上-f参数来强制删除。另外，想要删除一个目录，需要在rm命令后面一个-r参数才可以，否则删除不掉。我们来尝试删除前面创建的install.log和linux.log文件：

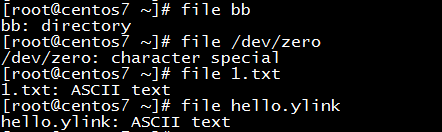




## file

file命令用于查看文件的类型，格式为“file 文件名”。

在Linux系统中，由于文本、目录、设备等所有这些一切都统称为文件，而我们又不能单凭后缀就知道具体的文件类型，这时就需要使用file命令来查看文件类型了。



## tar

tar命令用于对文件进行打包压缩或解压，格式为“tar [选项] [文件]”。

在Linux系统中，常见的文件格式比较多，其中主要使用的是.tar或.tar.gz或.tar.bz2格式，我们不用担心格式太多而记不住，其实这些格式大部分都是由tar命令来生成的。

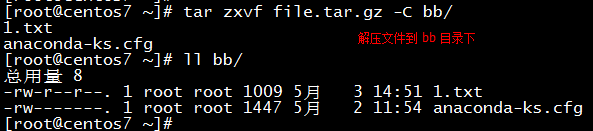
表2-14                                               tar命令的参数及其作用

|  |  |
| --- | --- |
| 参数 | 作用 |
| -c | 创建压缩文件 |
| -x | 解开压缩文件 |
| -t | 查看压缩包内有哪些文件 |
| -z | 用Gzip压缩或解压 |
| -j | 用bzip2压缩或解压 |
| -v | 显示压缩或解压的过程 |
| -f | 目标文件名 |
| -p | 保留原始的权限与属性 |
| -P | 使用绝对路径来压缩 |
| -C | 指定解压到的目录 |

首先，-c参数用于创建压缩文件，-x参数用于解压文件，因此这两个参数不能同时使用。其次，-z参数指定使用Gzip格式来压缩或解压文件，-j参数指定使用bzip2格式来压缩或解压文件。用户使用时则是根据文件的后缀来决定应使用何种格式参数进行解压。在执行某些压缩或解压操作时，可能需要花费数个小时，如果屏幕一直没有输出，您一方面不好判断打包的进度情况，另一方面也会怀疑电脑死机了，因此非常推荐使用-v参数向用户不断显示压缩或解压的过程。-C参数用于指定要解压到哪个指定的目录。-f参数特别重要，它必须放到参数的最后一位，代表要压缩或解压的软件包名称。







## grep

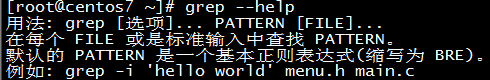
grep命令用于在**文本中执行关键词搜索**，并显示匹配的结果，格式为“grep [选项] [文件]”。grep命令的参数及其作用如表2-15所示。

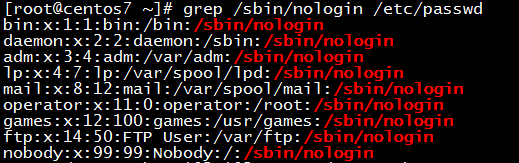
表2-15                                              grep命令的参数及其作用

|  |  |
| --- | --- |
| 参数 | 作用 |
| -b | 将可执行文件(binary)当作文本文件（text）来搜索 |
| -c | 仅显示找到的行数 |
| -i | 忽略大小写 |
| -n | 显示行号 |
| -v | 反向选择——仅列出没有“关键词”的行。 |

grep命令是用途最广泛的文本搜索匹配工具，虽然有很多参数，但是大多数基本上都用不到。我们在这里只讲两个最最常用的参数：-n参数用来显示搜索到信息的行号；-v参数用于反选信息（即没有包含关键词的所有信息行）。这两个参数几乎能完成您日后80%的工作需要，至于其他上百个参数，即使以后在工作期间遇到了，再使用man grep命令查询也来得及。

在Linux系统中，/etc/passwd文件是保存着所有的用户信息，而一旦用户的登录终端被设置成**/sbin/nologin**，则不再允许登录系统，因此可以使用grep命令来查找出当前系统中不允许登录系统的所有用户信息：





## find

find命令用于按照指定条件来**查找文件**，格式为“find [查找路径] 寻找条件 操作”。

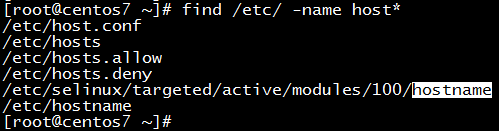
在Linux系统中，搜索工作一般都是通过find命令来完成的，它可以使用不同的文件特性作为寻找条件（如文件名、大小、修改时间、权限等信息），一旦匹配成功则默认将信息显示到屏幕上。find命令的参数以及作用如表2-16所示。

表2-16                                             find命令中的参数以及作用

|  |  |
| --- | --- |
| 参数 | 作用 |
| -name | 匹配名称 |
| -perm | 匹配权限（mode为完全匹配，-mode为包含即可） |
| -user | 匹配所有者 |
| -group | 匹配所有组 |
| -mtime -n +n | 匹配修改内容的时间（-n指n天以内，+n指n天以前） |
| -atime -n +n | 匹配访问文件的时间（-n指n天以内，+n指n天以前） |
| -ctime -n +n | 匹配修改文件权限的时间（-n指n天以内，+n指n天以前） |
| -nouser | 匹配无所有者的文件 |
| -nogroup | 匹配无所有组的文件 |
| -newer f1 !f2 | 匹配比文件f1新但比f2旧的文件 |
| --type b/d/c/p/l/f | 匹配文件类型（后面的字幕字母依次表示块设备、目录、字符设备、管道、链接文件、文本文件） |
| -size | 匹配文件的大小（+50KB为查找超过50KB的文件，而-50KB为查找小于50KB的文件） |
| -prune | 忽略某个目录 |
| -exec …… {}\; | 后面可跟用于进一步处理搜索结果的命令（下文会有演示） |

这里需要重点讲解一下-exec参数重要的作用。这个参数用于把find命令搜索到的结果交由紧随其后的命令作进一步处理，它十分类似于将要讲解的管道符技术，并且由于find命令对参数的特殊要求，因此虽然exec是长格式形式，但依然只需要一个减号（-）。

根据文件系统层次标准（Filesystem Hierarchy Standard）协议，Linux系统中的配置文件会保存到/etc目录中。如果要想获取到该目录中所有以host开头的文件列表，可以执行如下命令：



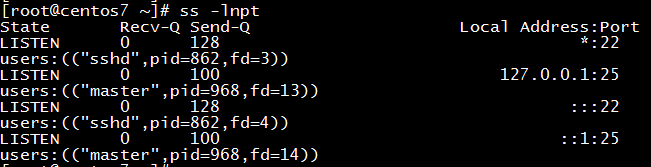
## ss

查看进程端口号占用

一般都是查找进程的id，然后再进程id查找端口占用

查找进程用命令：ps aux

查端口：**ss –lnpt**



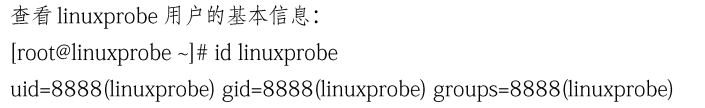
## useradd





## id

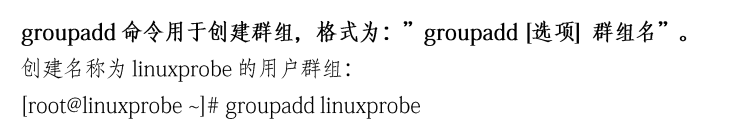
查看用户信息



## passwd



## groupadd



## su

切换用户： su – root

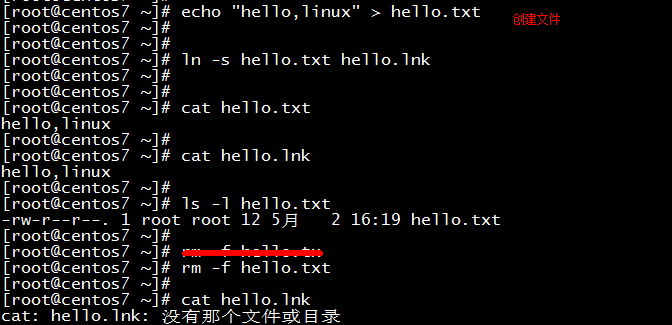
## ln

ln命令用于创建链接文件，格式为“**ln [选项] 目标**”，其可用的参数以及作用如表6-6所示。在使用ln命令时，是否添加-s参数，将创建出性质不同的两种“快捷方式”。因此如果没有扎实的理论知识和实践经验做铺垫，尽管能够成功完成实验，但永远不会明白为什么会成功。

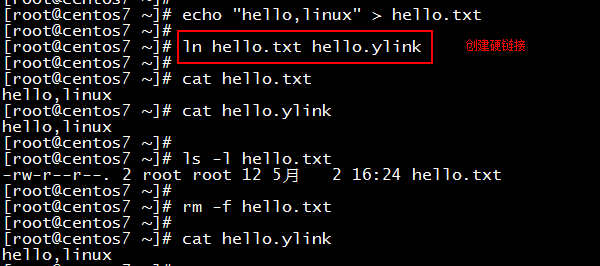
                                          ln命令中可用的参数以及作用

|  |  |
| --- | --- |
| 参数 | 作用 |
| -s | 创建“符号链接”（如果不带-s参数，则默认创建硬链接） |
| -f | 强制创建文件或目录的链接 |
| -i | 覆盖前先询问 |
| -v | 显示创建链接的过程 |

为了更好地理解软链接、硬链接的不同性质，接下来创建一个类似于Windows系统中快捷方式的软链接。这样，当原始文件被删除后，就无法读取新建的链接文件了。



接下来针对一个原始文件创建一个硬链接，即相当于针对原始文件的硬盘存储位置创建了一个指针，这样一来，新创建的这个硬链接就不再依赖于原始文件的名称等信息，也不会因为原始文件的删除而导致无法读取。同时可以看到创建硬链接后，原始文件的硬盘链接数量增加到了2。



# 管道符、重定向与环境变量

Linux运维shell，perl，python



## 管道符

管道命令符的作用也可以用一句话来概括“**把前一个命令原本要输出到屏幕的标准正常数据当作是后一个命令的标准输入**”。

通过匹配关键词/sbin/nologin找出了所有被限制登录系统的用户。在学完本节内容后，完全可以把下面这两条命令合并为一条：

找出被限制登录用户的命令是grep "/sbin/nologin" /etc/passwd；

统计文本行数的命令则是wc -l。

现在要做的就是把搜索命令的输出值传递给统计命令，即把原本要输出到屏幕的用户信息列表再交给wc命令作进一步的加工，因此只需要把管道符放到两条命令之间即可，具体如下。这简直是太方便了！





## 输入输出重定向

简而言之，输入重定向是指把**文件**导入到命令中，而输出重定向则是指把原本要输出到屏幕的数据信息写入到指定文件中。在日常的学习和工作中，相较于输入重定向，我们使用**输出重定向的频率更高**，所以又将输出重定向分为了**标准输出**重定向和**错误输出**重定向两种不同的技术，以及**清空写入**与**追加写入**两种模式。

标准输入重定向（STDIN，文件描述符为0）：默认从键盘输入，也可从其他文件或命令中输入。

标准输出重定向（STDOUT，文件描述符为1）：默认输出到屏幕。

错误输出重定向（STDERR，文件描述符为2）：默认输出到屏幕。

对于输入重定向来讲，用到的符号及其作用如表3-1所示。

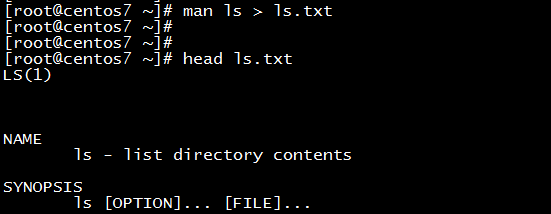
表3-1                                         输入重定向中用到的符号及其作用

|  |  |
| --- | --- |
| 符号 | 作用 |
| 命令 < 文件 | 将文件作为命令的标准输入 |
| 命令 << 分界符 | 从标准输入中读入，直到遇见分界符才停止 |
| 命令 < 文件1 > 文件2 | 将文件1作为命令的标准输入并将标准输出到文件2 |

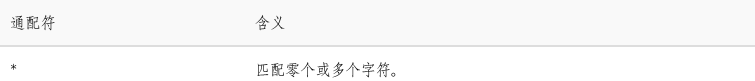
对于输出重定向来讲，用到的符号及其作用如表3-2所示。

表3-2                                         输出重定向中用到的符号及其作用

|  |  |
| --- | --- |
| 符号 | 作用 |
| 命令 > 文件 | 将标准输出重定向到一个文件中（清空原有文件的数据） |
| 命令 2> 文件 | 将错误输出重定向到一个文件中（清空原有文件的数据） |
| 命令 >> 文件 | 将标准输出重定向到一个文件中（追加到原有内容的后面） |
| 命令 2>> 文件 | 将错误输出重定向到一个文件中（追加到原有内容的后面） |
| 命令 >> 文件 2>&1  或 命令 &>> 文件 | 将标准输出与错误输出共同写入到文件中（追加到原有内容的后面） |



## 命令行通配符







## 重要的环境变量

变量是计算机系统用于保存可变值的数据类型。在Linux系统中，变量名称一般都是大写的，这是一种约定俗成的规范。我们可以直接通过变量名称来提取到对应的变量值。Linux系统中的环境变量是用来定义系统运行环境的一些参数，比如每个用户不同的家目录、邮件存放位置等。

要想让Linux系统能够正常运行并且为用户提供服务，需要数百个环境变量来协同工作，我们没有必要逐一查看、学习每一个变量，而是应该在有限的篇幅中精讲最重要的内容。

为了通过环境变量帮助Linux系统构建起能够为用户提供服务的工作运行环境，需要数百个变量协同工作才能完成。前文中曾经讲到，在Linux系统中一切都是文件，Linux命令也不例外。那么，在用户执行了一条命令之后，Linux系统中到底发生了什么事情呢？简单来说，**命令在Linux中的执行分为4个步骤。**

**第1步**：判断用户是否以绝对路径或相对路径的方式输入命令（如/bin/ls），如果是的话则直接执行。

**第2步**：Linux系统检查用户输入的命令是否为“别名命令”，即用一个自定义的命令名称来替换原本的命令名称。可以用alias命令来创建一个属于自己的命令别名，格式为“alias 别名=命令”。若要取消一个命令别名，则是用unalias命令，格式为“unalias 别名”。我们之前在使用rm命令删除文件时，Linux系统都会要求我们再确认是否执行删除操作，其实这就是Linux系统为了防止用户误删除文件而特意设置的rm别名命令.

**第3步**：Bash解释器判断用户输入的是内部命令还是外部命令。内部命令是解释器内部的指令，会被直接执行；而用户在绝大部分时间输入的是外部命令，这些命令交由步骤4继续处理。可以使用“type命令名称”来判断用户输入的命令是内部命令还是外部命令。

**第4步**：系统在多个路径中查找用户输入的命令文件，而定义这些路径的变量叫作PATH，可以简单地把它理解成是“解释器的小助手”，作用是告诉Bash解释器待执行的命令可能存放的位置，然后Bash解释器就会乖乖地在这些位置中逐个查找。PATH是由多个路径值组成的变量，每个路径值之间用**冒号**间隔，对这些路径的增加和删除操作将影响到Bash解释器对Linux命令的查找。

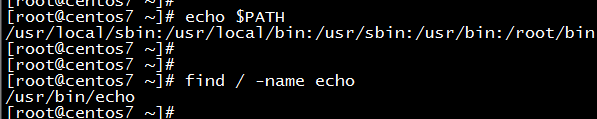


表3-3                                       Linux系统中最重要的10个环境变量

|  |  |
| --- | --- |
| 变量名称 | 作用 |
| HOME | 用户的主目录（即家目录） |
| SHELL | 用户在使用的Shell解释器名称 |
| HISTSIZE | 输出的历史命令记录条数 |
| HISTFILESIZE | 保存的历史命令记录条数 |
| MAIL | 邮件保存路径 |
| LANG | 系统语言、语系名称 |
| RANDOM | 生成一个随机数字 |
| PS1 | Bash解释器的提示符 |
| PATH | 定义解释器搜索用户执行命令的路径 |
| EDITOR | 用户默认的文本编辑器 |

Linux作为一个**多用户**多任务的操作系统，能够为每个用户提供独立的、合适的工作运行环境，因此，一个相同的变量会因为用户身份的不同而具有不同的值。

其实变量是由固定的变量名与用户或系统设置的变量值两部分组成的，我们完全可以自行创建变量，来满足工作需求。例如设置一个名称为WORKDIR的变量，方便用户更轻松地进入一个层次较深的目录：

[root@linuxprobe ~]# mkdir /home/workdir

[root@linuxprobe ~]# WORKDIR=/home/workdir

[root@linuxprobe ~]# cd $WORKDIR

[root@linuxprobe workdir]# pwd

/home/workdir

但是，这样的变量不具有全局性，作用范围也有限，默认情况下不能被其他用户使用。如果工作需要，可以使用export命令将其提升为全局变量，这样其他用户也就可以使用它了：

# Vim编辑器

神之编辑器 emacs

编辑器之神 vi

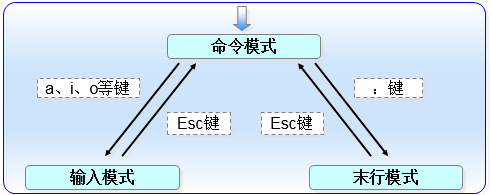
“在[Linux系统](https://www.linuxprobe.com/)中一切都是文件，而配置一个服务就是在修改其配置文件的参数”。而且在日常工作中大家也肯定免不了要编写文档，这些工作都是通过文本编辑器来完成的。

Vim之所以能得到广大厂商与用户的认可，原因在于Vim编辑器中设置了三种模式—命令模式、末行模式和编辑模式，每种模式分别又支持多种不同的命令快捷键，这大大提高了工作效率，而且用户在习惯之后也会觉得相当顺手。要想高效率地操作文本，就必须先搞清这三种模式的操作区别以及模式之间的切换方法（见图4-1）。

命令模式：控制光标移动，可对文本进行复制、粘贴、删除和查找等工作。

输入模式：正常的文本录入。

末行模式：保存或退出文档，以及设置编辑环境。



在每次运行Vim编辑器时，默认进入命令模式，此时需要先切换到输入模式后再进行文档编写工作，而每次在编写完文档后需要先返回命令模式，然后再进入末行模式，执行文档的保存或退出操作。在Vim中，无法直接从输入模式切换到末行模式。Vim编辑器中内置的命令有成百上千种用法，为了能够帮助读者更快地掌握Vim编辑器，表4-1总结了在命令模式中最常用的一些命令。

表4-1                                                      Vim中常用的命令

|  |  |
| --- | --- |
| 命令 | 作用 |
| dd | 删除（剪切）光标所在整行 |
| 5dd | 删除（剪切）从光标处开始的5行 |
| yy | 复制光标所在整行 |
| 5yy | 复制从光标处开始的5行 |
| n | 显示搜索命令定位到的下一个字符串 |
| N | 显示搜索命令定位到的上一个字符串 |
| u | 撤销上一步的操作 |
| p | 将之前删除（dd）或复制（yy）过的数据粘贴到光标后面 |

末行模式主要用于保存或退出文件，以及设置Vim编辑器的工作环境，还可以让用户执行外部的[Linux命令](https://www.linuxprobe.com/)或跳转到所编写文档的特定行数。要想切换到末行模式，在命令模式中输入一个冒号就可以了。末行模式中可用的命令如表4-2所示。

表4-2                                                  末行模式中可用的命令

|  |  |
| --- | --- |
| 命令 | 作用 |
| :w | 保存 |
| :q | 退出 |
| :q! | 强制退出（放弃对文档的修改内容） |
| :wq! | 强制保存退出 |
| :set nu | 显示行号 |
| :set nonu | 不显示行号 |
| :命令 | 执行该命令 |
| :整数 | 跳转到该行 |
| :s/one/two | 将当前光标所在行的第一个one替换成two |
| :s/one/two/g | 将当前光标所在行的所有one替换成two |
| :%s/one/two/g | 将全文中的所有one替换成two |
| ?字符串 | 在文本中从下至上搜索该字符串 |
| /字符串 | 在文本中从上至下搜索该字符串 |

“G”：快速到达文件末尾

“gg”：快速到达文件头部

## 修改主机名

为了便于在局域网中查找某台特定的主机，或者对主机进行区分，除了要有IP地址外，还要为主机配置一个主机名，主机之间可以通过这个类似于域名的名称来相互访问。在Linux系统中，主机名大多保存在/etc/hostname文件中，接下来将/etc/hostname文件的内容修改为“linuxprobe.com”，步骤如下。

**第1步**：使用Vim编辑器修改“/etc/hostname”主机名称文件。

**第2步**：把原始主机名称删除后追加“linuxprobe.com”。注意，使用Vim编辑器修改主机名称文件后，要在末行模式下执行:wq!命令才能保存并退出文档。

**第3步**：保存并退出文档，然后使用hostname命令检查是否修改成功。

[root@linuxprobe ~]# vim /etc/hostname

linuxprobe.com

hostname命令用于查看当前的主机名称，但有时主机名称的改变不会立即同步到系统中，所以如果发现修改完成后还显示原来的主机名称，可重启虚拟机后再行查看：

[root@linuxprobe ~]# hostname

linuxprobe.com

## 修改网卡

网卡IP地址配置的是否正确是两台服务器是否可以相互通信的前提。在Linux系统中，一切都是文件，因此配置网络服务的工作其实就是在编辑网卡配置文件，因此这个小任务不仅可以帮助您练习使用Vim编辑器，而且也为您后面学习Linux中的各种服务配置打下了坚实的基础。当您认真学习完本书后，一定会特别有成就感，因为本书前面的基础部分非常扎实，而后面内容则具有几乎一致的网卡IP地址和运行环境，从而确保您全身心地投入到各类服务程序的学习上，而不用操心系统环境的问题。

如果您具备一定的运维经验或者熟悉早期的Linux系统，则在学习本书时会遇到一些不容易接受的差异变化。在RHEL 5、RHEL 6中，网卡配置文件的前缀为eth，第1块网卡为eth0，第2块网卡为eth1；以此类推。而在RHEL 7中，网卡配置文件的前缀则以ifcfg开始，加上网卡名称共同组成了网卡配置文件的名字，例如ifcfg-eno16777736；好在除了文件名变化外也没有其他大的区别。

现在有一个名称为ifcfg-eno16777736的网卡设备，我们将其配置为开机自启动，并且IP地址、子网、网关等信息由人工指定，其步骤应该如下所示。

**第1步**：首先切换到/etc/sysconfig/network-scripts目录中（存放着网卡的配置文件）。

**第2步**：使用Vim编辑器修改网卡文件ifcfg-eno16777736，逐项写入下面的配置参数并保存退出。由于每台设备的硬件及架构是不一样的，因此请读者使用ifconfig命令自行确认各自网卡的默认名称。

设备类型：TYPE=Ethernet

地址分配模式：BOOTPROTO=static/none

网卡名称：NAME=eno16777736

是否启动：ONBOOT=yes

IP地址：IPADDR=192.168.10.10

子网掩码：NETMASK=255.255.255.0

网关地址：GATEWAY=192.168.10.1

DNS地址：DNS1=192.168.10.1

**第3步**：重启网络服务并测试网络是否联通。

进入到网卡配置文件所在的目录，然后编辑网卡配置文件，在其中填入下面的信息：

[root@linuxprobe ~]# cd /etc/sysconfig/network-scripts/

[root@linuxprobe network-scripts]# vim ifcfg-eno16777736

TYPE=Ethernet

BOOTPROTO=static

NAME=eno16777736

ONBOOT=yes

IPADDR=192.168.10.10

NETMASK=255.255.255.0

GATEWAY=192.168.10.1

DNS1=192.168.10.1

执行重启网卡设备的命令（在正常情况下不会有提示信息），然后通过ping命令测试网络能否联通。由于在Linux系统中ping命令不会自动终止，因此需要手动按下Ctrl-c键来强行结束进程。

[root@linuxprobe network-scripts]# **systemctl restart network**

[root@linuxprobe network-scripts]# ping 192.168.10.10

PING 192.168.10.10 (192.168.10.10) 56(84) bytes of data.

64 bytes from 192.168.10.10: icmp\_seq=1 ttl=64 time=0.081 ms

64 bytes from 192.168.10.10: icmp\_seq=2 ttl=64 time=0.083 ms

64 bytes from 192.168.10.10: icmp\_seq=3 ttl=64 time=0.059 ms

64 bytes from 192.168.10.10: icmp\_seq=4 ttl=64 time=0.097 ms

^C

--- 192.168.10.10 ping statistics ---

4 packets transmitted, 4 received, 0% packet loss, time 2999ms

rtt min/avg/max/mdev = 0.059/0.080/0.097/0.013 ms

## 配置Yum仓库

Yum软件仓库的作用是为了进一步简化RPM管理软件的难度以及自动分析所需软件包及其依赖关系的技术。可以把Yum想象成是一个硕大的软件仓库，里面保存有几乎所有常用的工具，而且只需要说出所需的软件包名称，系统就会自动为您搞定一切。

既然要使用Yum软件仓库，就要先把它搭建起来，然后将其配置规则确定好才行。鉴于第6章才会讲解Linux的存储结构和设备挂载操作，所以我们当前还是将重心放到Vim编辑器的学习上。如果遇到看不懂的参数也不要紧，后面章节会单独讲解。搭建并配置Yum软件仓库的大致步骤如下所示。

**第1步**：进入到/etc/yum.repos.d/目录中（因为该目录存放着Yum软件仓库的配置文件）。

**第2步**：使用Vim编辑器创建一个名为rhel7.repo的新配置文件（文件名称可随意，但后缀必须为.repo），逐项写入下面加粗的配置参数并保存退出（不要写后面的中文注释）。

**[rhel-media]**：Yum软件仓库唯一标识符，避免与其他仓库冲突。

**name=linuxprobe**：Yum软件仓库的名称描述，易于识别仓库用处。

**baseurl=file:///media/cdrom**：提供的方式包括FTP（ftp://..）、HTTP（http://..）、本地（file:///..）。

**enabled=1**：设置此源是否可用；1为可用，0为禁用。

**gpgcheck=1**：设置此源是否校验文件；1为校验，0为不校验。

**gpgkey=file:///media/cdrom/RPM-GPG-KEY-redhat-release**：若上面参数开启校验，那么请指定公钥文件地址。

**第3步**：按配置参数的路径挂载光盘，并把光盘挂载信息写入到/etc/fstab文件中。

**第4步**：使用“yum install httpd -y”命令检查Yum软件仓库是否已经可用。

进入/etc/yum.repos.d目录中后创建Yum配置文件：

[root@linuxprobe ~]# cd /etc/yum.repos.d/

[root@linuxprobe yum.repos.d]# vim rhel7.repo

[rhel7]

name=rhel7

baseurl=file:///media/cdrom

enabled=1

gpgcheck=0

创建挂载点后进行挂载操作，并设置成开机自动挂载（详见第6章）。尝试使用Yum软件仓库来安装Web服务，出现**Complete**！则代表配置正确：

[root@linuxprobe yum.repos.d]# mkdir -p /media/cdrom

[root@linuxprobe yum.repos.d]# mount /dev/cdrom /media/cdrom

mount: /dev/sr0 is write-protected, mounting read-only

[root@linuxprobe yum.repos.d]# vim /etc/fstab

/dev/cdrom /media/cdrom iso9660 defaults 0 0

[root@linuxprobe ~]# yum install httpd

Loaded plugins: langpacks, product-id, subscription-manager

………………省略部分输出信息………………

Dependencies Resolved

================================================================================

Package Arch Version Repository Size

================================================================================

Installing:

httpd x86\_64 2.4.6-17.el7 rhel 1.2 M

Installing for dependencies:

apr x86\_64 1.4.8-3.el7 rhel 103 k

apr-util x86\_64 1.5.2-6.el7 rhel 92 k

httpd-tools x86\_64 2.4.6-17.el7 rhel 77 k

mailcap noarch 2.1.41-2.el7 rhel 31 k

Transaction Summary

================================================================================

Install 1 Package (+4 Dependent packages)

Total download size: 1.5 M

Installed size: 4.3 M

Is this ok [y/d/N]: **y**

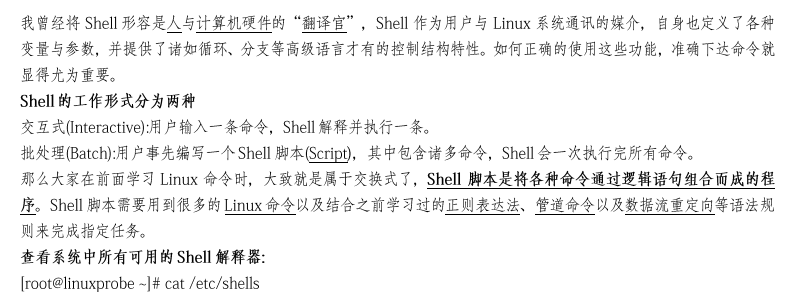
Downloading packages:

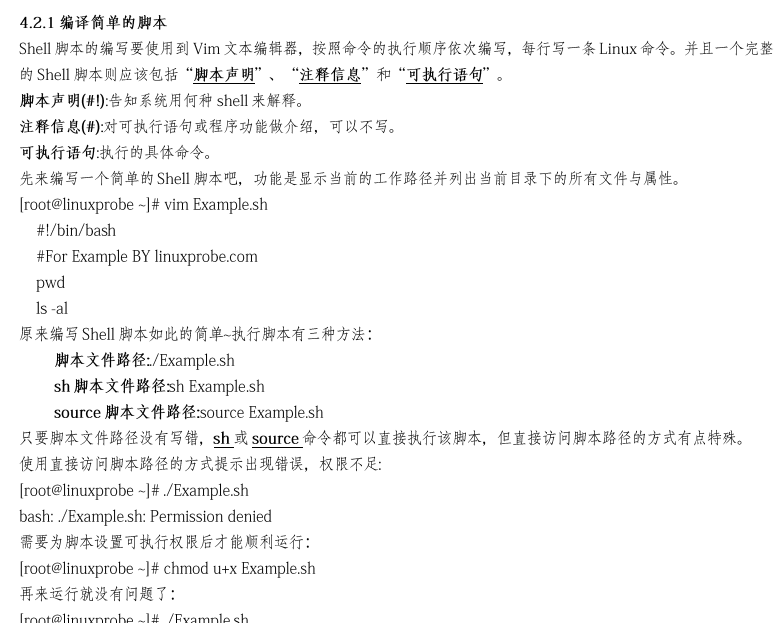
--------------------------------------------------------------------------------

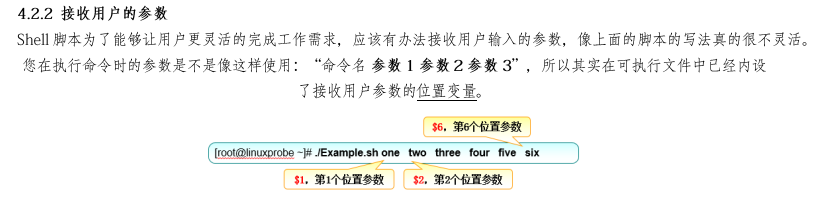
………………省略部分输出信息………………

**Complete!**

# Shell编程(了解)

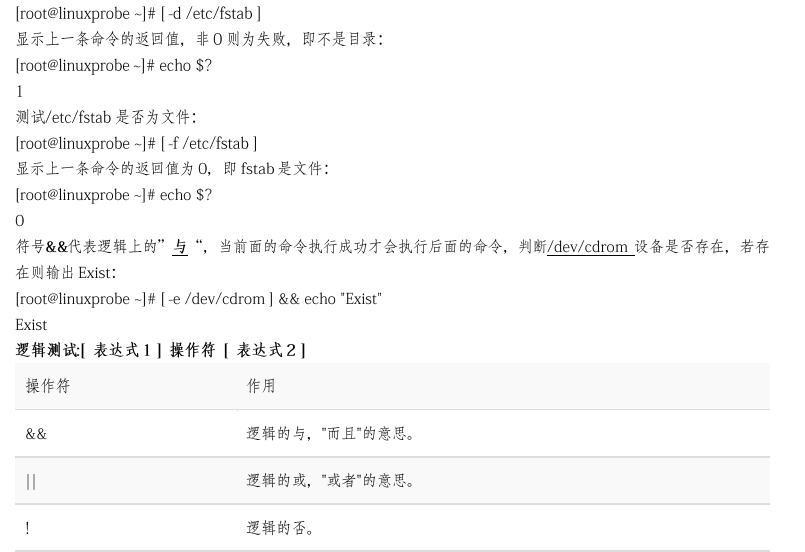


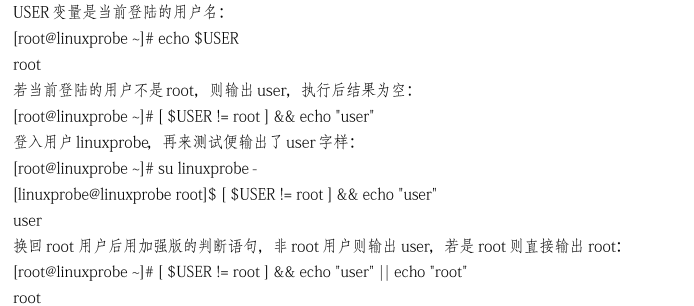








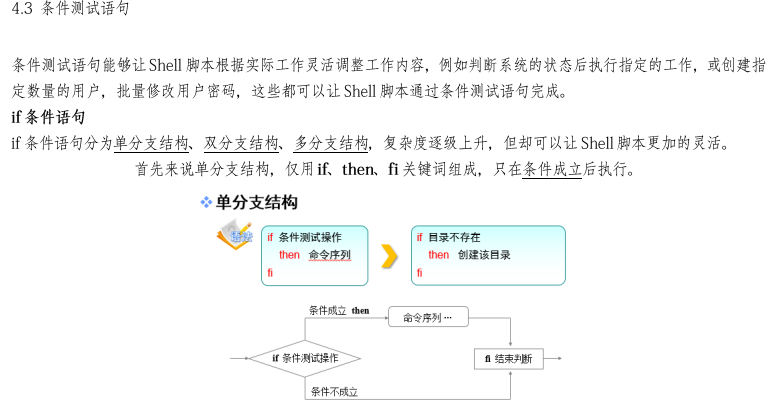


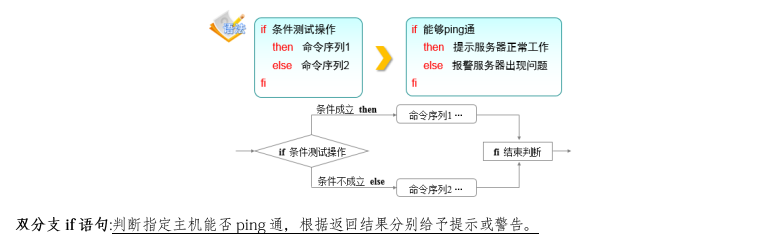


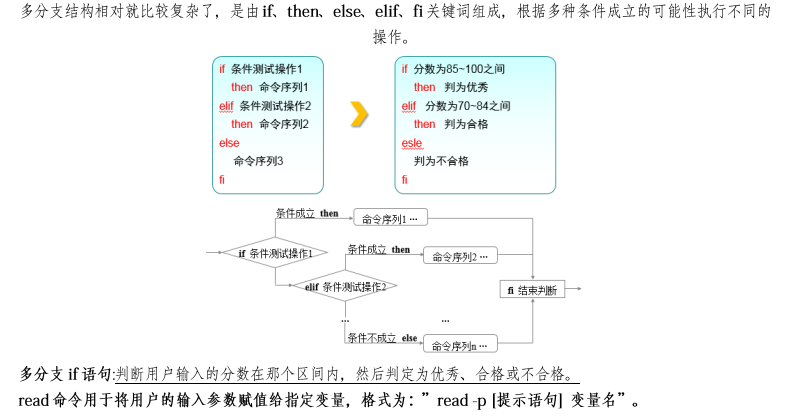


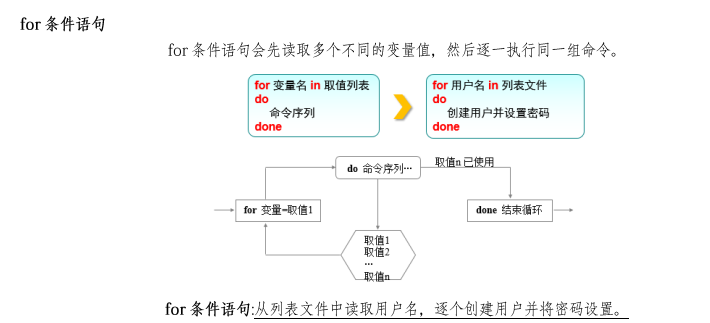


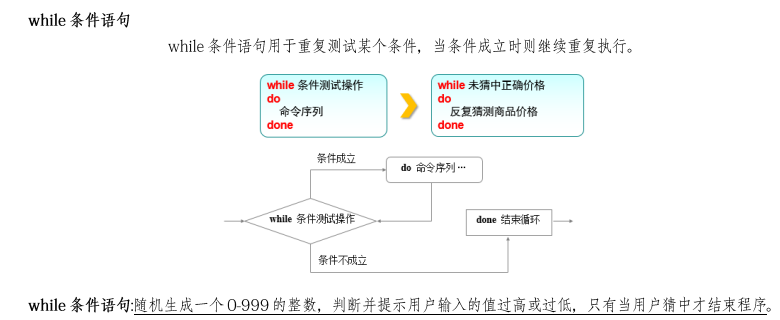


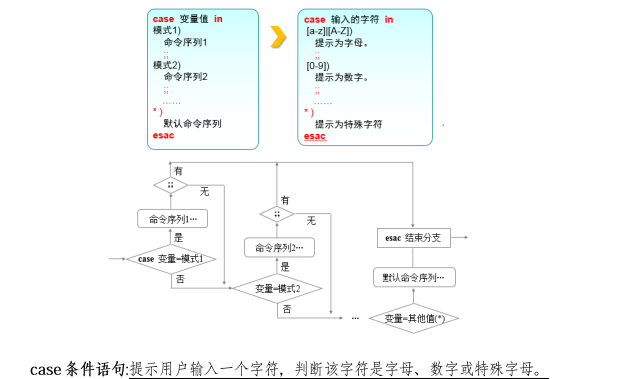












# 计划任务服务程序

经验丰富的系统运维工程师可以使得Linux在无需人为介入的情况下，在指定的时间段自动启用或停止某些服务或命令，从而实现运维的自动化。尽管我们现在已经有了功能彪悍的脚本程序来执行一些批处理工作，但是，如果仍然需要在每天凌晨两点敲击键盘回车键来执行这个脚本程序，这简直太痛苦了（当然，也可以训练您的小猫在半夜按下回车键）。接下来，刘遄老师将向大家讲解如何设置服务器的计划任务服务，把周期性、规律性的工作交给系统自动完成。

计划任务分为一次性计划任务与长期性计划任务，大家可以按照如下方式理解。

一次性计划任务：今晚11点30分开启网站服务。

长期性计划任务：每周一的凌晨3点25分把/home/wwwroot目录打包备份为backup.tar.gz。

顾名思义，一次性计划任务只执行一次，一般用于满足临时的工作需求。我们可以用at命令实现这种功能，只需要写成“at 时间”的形式就可以。如果想要查看已设置好但还未执行的一次性计划任务，可以使用“at -l”命令；要想将其删除，可以用“atrm 任务序号”。在使用at命令来设置一次性计划任务时，默认采用的是交互式方法。例如，使用下述命令将系统设置为在今晚23:30分自动重启网站服务。

[root@linuxprobe ~]# at 23:30

at > systemctl restart httpd

at > 此处请同时按下Ctrl+d来结束编写计划任务

job 3 at Mon Apr 27 23:30:00 2015

[root@linuxprobe ~]# at -l

3 Mon Apr 27 23:30:00 2016 a root

如果读者想挑战一下难度更大但简捷性更高的方式，可以把前面学习的管道符（任意门）放到两条命令之间，让at命令接收前面echo命令的输出信息，以达到通过非交互式的方式创建计划一次性任务的目的。

[root@linuxprobe ~]# echo "systemctl restart httpd" | at 23:30

job 4 at Mon Apr 27 23:30:00 2015

[root@linuxprobe ~]# at -l

3 Mon Apr 27 23:30:00 2016 a root

4 Mon Apr 27 23:30:00 2016 a root

如果我们不小心设置了两个一次性计划任务，可以使用下面的命令轻松删除其中一个：

[root@linuxprobe ~]# atrm 3

[root@linuxprobe ~]# at -l

4 Mon Apr 27 23:30:00 2016 a root

如果我们希望Linux系统能够周期性地、有规律地执行某些具体的任务，那么Linux系统中默认启用的**crond**服务简直再适合不过了。创建、编辑计划任务的命令为“crontab -e”，查看当前计划任务的命令为“crontab -l”，删除某条计划任务的命令为“crontab -r”。另外，如果您是以管理员的身份登录的系统，还可以在crontab命令中加上-u参数来编辑他人的计划任务。

在正式部署计划任务前，请先跟刘遄老师念一下口诀“分、时、日、月、星期 命令”。这是使用crond服务设置任务的参数格式（其格式见表4-6）。需要注意的是，如果有些字段没有设置，则需要使用星号（**\***）占位，如图4-23所示。



表4-6                                       使用crond设置任务的参数字段说明

|  |  |
| --- | --- |
| 字段 | 说明 |
| 分钟 | 取值为0～59的整数 |
| 小时 | 取值为0～23的任意整数 |
| 日期 | 取值为1～31的任意整数 |
| 月份 | 取值为1～12的任意整数 |
| 星期 | 取值为0～7的任意整数，其中0与7均为星期日 |
| 命令 | 要执行的命令或程序脚本 |

假设在每周一、三、五的凌晨3点25分，都需要使用tar命令把某个网站的数据目录进行打包处理，使其作为一个备份文件。我们可以使用crontab -e命令来创建计划任务。为自己创建计划任务无需使用-u参数，具体的实现效果的参数如crontab -l命令结果所示：

[root@linuxprobe ~]# crontab -e

no crontab for root - using an empty one

crontab: installing new crontab

[root@linuxprobe ~]# crontab -l

**25 3 \* \* 1,3,5 /usr/bin/tar -czvf backup.tar.gz /home/wwwroot**

需要说明的是，除了用逗号（,）来分别表示多个时间段，例如“8,9,12”表示8月、9月和12月。还可以用减号（-）来表示一段连续的时间周期（例如字段“日”的取值为“12-15”，则表示每月的12～15日）。以及用除号（/）表示执行任务的间隔时间（例如“\*/2”表示每隔2分钟执行一次任务）。

如果在crond服务中需要同时包含多条计划任务的命令语句，应每行仅写一条。例如我们再添加一条计划任务，它的功能是每周一至周五的凌晨1点钟自动清空/tmp目录内的所有文件。尤其需要注意的是，在crond服务的计划任务参数中，所有命令一定要用绝对路径的方式来写，如果不知道绝对路径，请用whereis命令进行查询，rm命令路径为下面输出信息中加粗部分。

[root@linuxprobe ~]# whereis rm

rm: **/usr/bin/rm** /usr/share/man/man1/rm.1.gz /usr/share/man/man1p/rm.1p.gz

[root@linuxprobe ~]# crontab -e

crontab: installing new crontab

[root@linuxprobe ~]# crontab -l

25 3 \* \* 1,3,5 /usr/bin/tar -czvf backup.tar.gz /home/wwwroot

**0 1 \* \* 1-5 /usr/bin/rm -rf /tmp/\***

在本节最后，刘遄老师再来啰嗦几句在工作中使用计划服务的注意事项。

在crond服务的配置参数中，可以像Shell脚本那样以#号开头写上注释信息，这样在日后回顾这段命令代码时可以快速了解其功能、需求以及编写人员等重要信息。

计划任务中的“分”字段必须有数值，绝对不能为空或是\*号，而“日”和“星期”字段不能同时使用，否则就会发生冲突。

# 用户身份

设计Linux系统的初衷之一就是为了满足多个用户同时工作的需求，因此Linux系统必须具备很好的安全性。第1章在安装RHEL 7操作系统时，特别要求设置root管理员密码，这个root管理员就是存在于所有类UNIX系统中的超级用户。它拥有最高的系统所有权，能够管理系统的各项功能，如添加/删除用户、启动/关闭服务进程、开启/禁用硬件设备等。虽然以root管理员的身份工作时不会受到系统的限制，但俗语讲“能力越大，责任就越大”，因此一旦使用这个高能的root管理员权限执行了错误的命令可能会直接毁掉整个系统。使用与否，确实需要好好权衡一下。

在学习时是否要使用root管理员权限来控制整个系统呢？面对这个问题，网络上有很多文章建议以普通用户的身份来操作—这是一个更安全也更“无责任”的回答。今天，[刘遄](https://www.linuxprobe.com/)老师就要冒天下之大不韪给出自己的心得—强烈推荐大家在学习时使用root管理员权限！

这种为root管理员正名的决绝态度在网络中应该还是很少见的，我之所以力荐root管理员权限，原因很简单。因为在Linux的学习过程中如果使用普通用户身份进行操作，则在配置服务之后出现错误时很难判断是系统自身的问题还是因为权限不足而导致的；这无疑会给大家的学习过程徒增坎坷。更何况我们的实验环境是使用VMware虚拟机软件搭建的，可以将安装好的系统设置为一次快照，这即便系统彻底崩溃了，您也可以在5秒钟的时间内快速还原出一台全新的系统，而不用担心数据丢失。

总之，[刘遄](https://www.linuxprobe.com/)老师在培训时都推荐每位学生使用root管理员权限来学习Linux系统，等到工作时再根据生产环境决定使用哪个用户权限；这些仅与选择相关，而非技术性问题。

另外，很多图书或培训机构的老师会讲到，Linux系统中的管理员就是root。这其实是错误的，Linux系统的管理员之所以是root，并不是因为它的名字叫root，而是因为该用户的身份号码即UID（User IDentification）的数值为0。在Linux系统中，UID就相当于我们的身份证号码一样具有唯一性，因此可通过用户的UID值来判断用户身份。在RHEL 7系统中，用户身份有下面这些。

管理员UID为0：系统的管理员用户。

系统用户UID为1～999： Linux系统为了避免因某个服务程序出现漏洞而被黑客提权至整台服务器，默认服务程序会有独立的系统用户负责运行，进而有效控制被破坏范围。

普通用户UID从1000开始：是由管理员创建的用于日常工作的用户。

需要注意的是，UID是不能冲突的，而且管理员创建的普通用户的UID默认是从1000开始的（即使前面有闲置的号码）。

为了方便管理属于同一组的用户，Linux系统中还引入了用户组的概念。通过使用用户组号码（GID，Group IDentification），我们可以把多个用户加入到同一个组中，从而方便为组中的用户统一规划权限或指定任务。假设有一个公司中有多个部门，每个部门中又有很多员工。如果只想让员工访问本部门内的资源，则可以针对部门而非具体的员工来设置权限。例如，可以通过对技术部门设置权限，使得只有技术部门的员工可以访问公司的数据库信息等。

另外，在Linux系统中创建每个用户时，将自动创建一个与其同名的基本用户组，而且这个基本用户组只有该用户一个人。如果该用户以后被归纳入其他用户组，则这个其他用户组称之为扩展用户组。一个用户只有一个基本用户组，但是可以有多个扩展用户组，从而满足日常的工作需要。

# 权限

尽管在Linux系统中一切都是文件，但是每个文件的类型不尽相同，因此Linux系统使用了不同的字符来加以区分，常见的字符如下所示。

 -：普通文件。

d：目录文件。

l：链接文件。

b：块设备文件。

c：字符设备文件。

p：管道文件。

在Linux系统中，每个文件都有所属的所有者和所有组，并且规定了文件的所有者、所有组以及其他人对文件所拥有的可读（r）、可写（w）、可执行（x）等权限。对于一般文件来说，权限比较容易理解：“可读”表示能够读取文件的实际内容；“可写”表示能够编辑、新增、修改、删除文件的实际内容；“可执行”则表示能够运行一个脚本程序。但是，对于目录文件来说，理解其权限设置来就不那么容易了。很多资深Linux用户其实也没有真正搞明白。

刘遄老师在这里给大家详细讲解一下目录文件的权限设置。对目录文件来说，“可读”表示能够读取目录内的文件列表；“可写”表示能够在目录内新增、删除、重命名文件；而“可执行”则表示能够进入该目录。

文件的读、写、执行权限可以简写为rwx，亦可分别用数字4、2、1来表示，文件所有者，所属组及其他用户权限之间无关联，如表5-5所示。



文件权限的数字法表示基于字符表示（rwx）的权限计算而来，其目的是简化权限的表示。例如，若某个文件的权限为**7**则代表可读、可写、可执行（4+2+1）；若权限为6则代表可读、可写（4+2）。我们来看这样一个例子。现在有这样一个文件，其所有者拥有可读、可写、可执行的权限，其文件所属组拥有可读、可写的权限；而且其他人只有可读的权限。那么，这个文件的权限就是rwxrw-r--，数字法表示即为764。不过大家千万别再将这三个数字相加，计算出7+6+4=17的结果，这是小学的数学加减法，不是Linux系统的权限数字表示法，三者之间没有互通关系。

Linux系统的文件权限相当复杂，但是用途很广泛，建议大家把它彻底搞清楚之后再学习下一节的内容。现在来练习一下。请各位读者分别计算数字表示法764、642、153、731所对应的字符表示法，然后再把rwxrw-r--、rw--w--wx、rw-r--r--转换成数字表示法。

下面我们利用上文讲解的知识，一起分析图5-1中所示的文件信息。



在图5-1中，包含了文件的类型、访问权限、所有者（属主）、所属组（属组）、占用的磁盘大小、修改时间和文件名称等信息。通过分析可知，该文件的类型为普通文件，所有者权限为可读、可写（rw-），所属组权限为可读（r--），除此以外的其他人也只有可读权限（r--），文件的磁盘占用大小是34298字节，最近一次的修改时间为4月2日的凌晨23分，文件的名称为install.log。

# su命令与sudo服务

各位读者在实验环境中很少遇到安全问题，并且为了避免因权限因素导致配置服务失败，从而建议使用root管理员来学习本书，但是在生产环境中还是要对安全多一份敬畏之心，不要用root管理员去做所有事情。因为一旦执行了错误的命令，可能会直接导致系统崩溃，这样一来，不但客户指责、领导批评，没准奖金也会鸡飞蛋打。但转头一想，尽管Linux系统为了安全性考虑，使得许多系统命令和服务只能被root管理员来使用，但是这也让普通用户受到了更多的权限束缚，从而导致无法顺利完成特定的工作任务。

su命令可以解决切换用户身份的需求，使得当前用户在不退出登录的情况下，顺畅地切换到其他用户，比如从root管理员切换至普通用户：

[root@linuxprobe ~]# id

uid=0(root) gid=0(root) groups=0(root)

[root@linuxprobe ~]# su - linuxprobe

Last login: Wed Jan 4 01:17:25 EST 2017 on pts/0

[linuxprobe@linuxprobe ~]$ id

uid=1000(linuxprobe) gid=1000(linuxprobe) groups=1000(linuxprobe) context=unconfined\_u:unconfined\_r:unconfined\_t:s0-s0:c0.c1023

细心的读者一定会发现，上面的su命令与用户名之间有一个减号（-），这意味着完全切换到新的用户，即把环境变量信息也变更为新用户的相应信息，而不是保留原始的信息。强烈建议在切换用户身份时添加这个减号（-）。

另外，当从root管理员切换到普通用户时是不需要密码验证的，而从普通用户切换成root管理员就需要进行密码验证了；这也是一个必要的安全检查：

[linuxprobe@linuxprobe root]$ su root

Password:

[root@linuxprobe ~]# su - linuxprobe

Last login: Mon Aug 24 19:27:09 CST 2017 on pts/0

[linuxprobe@linuxprobe ~]$ exit

logout

[root@linuxprobe ~]#

尽管像上面这样使用su命令后，普通用户可以完全切换到root管理员身份来完成相应工作，但这将暴露root管理员的密码，从而增大了系统密码被黑客获取的几率；这并不是最安全的方案。

刘遄老师接下来将介绍如何使用sudo命令把特定命令的执行权限赋予给指定用户，这样既可保证普通用户能够完成特定的工作，也可以避免泄露root管理员密码。我们要做的就是合理配置sudo服务，以便兼顾系统的安全性和用户的便捷性。sudo服务的配置原则也很简单—在保证普通用户完成相应工作的前提下，尽可能少地赋予额外的权限。

sudo命令用于给普通用户提供额外的权限来完成原本root管理员才能完成的任务，格式为“sudo [参数] 命令名称”。sudo服务中可用的参数以及相应的作用如表5-7所示。

表5-7                                          sudo服务中的可用参数以及作用

|  |  |
| --- | --- |
| 参数 | 作用 |
| -h | 列出帮助信息 |
| -l | 列出当前用户可执行的命令 |
| -u 用户名或UID值 | 以指定的用户身份执行命令 |
| -k | 清空密码的有效时间，下次执行sudo时需要再次进行密码验证 |
| -b | 在后台执行指定的命令 |
| -p | 更改询问密码的提示语 |

总结来说，sudo命令具有如下功能：

限制用户执行指定的命令：

记录用户执行的每一条命令；

配置文件（/etc/sudoers）提供集中的用户管理、权限与主机等参数；

验证密码的后5分钟内（默认值）无须再让用户再次验证密码。

当然，如果担心直接修改配置文件会出现问题，则可以使用sudo命令提供的visudo命令来配置用户权限。这条命令在配置用户权限时将禁止多个用户同时修改sudoers配置文件，还可以对配置文件内的参数进行语法检查，并在发现参数错误时进行报错。

只有root管理员才可以使用visudo命令编辑sudo服务的配置文件。

**visudo: >>> /etc/sudoers: syntax error near line 111 <<<**

What now?

Options are:

(e)dit sudoers file again

(x)it without saving changes to sudoers file

(Q)uit and save changes to sudoers file (DANGER!)

使用visudo命令配置sudo命令的配置文件时，其操作方法与Vim编辑器中用到的方法一致，因此在编写完成后记得在末行模式下保存并退出。在sudo命令的配置文件中，按照下面的格式将第99行（大约）填写上指定的信息：

**谁可以使用  允许使用的主机=（以谁的身份）  可执行命令的列表**

[root@linuxprobe ~]# visudo

96 ##

97 ## Allow root to run any commands anywhere

98 root ALL=(ALL) ALL

**99 linuxprobe ALL=(ALL) ALL**

在填写完毕后记得要先保存再退出，然后切换至指定的普通用户身份，此时就可以用sudo -l命令查看到所有可执行的命令了（下面的命令中，验证的是该普通用户的密码，而不是root管理员的密码，请读者不要搞混了）：

[root@linuxprobe ~]# su - linuxprobe

Last login: Thu Sep 3 15:12:57 CST 2017 on pts/1

[linuxprobe@linuxprobe ~]$ sudo -l

[sudo] password for linuxprobe:**此处输入linuxprobe用户的密码**

Matching Defaults entries for linuxprobe on this host:

requiretty, !visiblepw, always\_set\_home, env\_reset, env\_keep="COLORS

DISPLAY HOSTNAME HISTSIZE INPUTRC KDEDIR LS\_COLORS", env\_keep+="MAIL PS1

PS2 QTDIR USERNAME LANG LC\_ADDRESS LC\_CTYPE", env\_keep+="LC\_COLLATE

LC\_IDENTIFICATION LC\_MEASUREMENT LC\_MESSAGES", env\_keep+="LC\_MONETARY

LC\_NAME LC\_NUMERIC LC\_PAPER LC\_TELEPHONE", env\_keep+="LC\_TIME LC\_ALL

LANGUAGE LINGUAS \_XKB\_CHARSET XAUTHORITY",

secure\_path=/sbin\:/bin\:/usr/sbin\:/usr/bin

**User linuxprobe may run the following commands on this host:**

**(ALL) ALL**

接下来是见证奇迹的时刻！作为一名普通用户，是肯定不能看到root管理员的家目录（/root）中的文件信息的，但是，只需要在想执行的命令前面加上sudo命令就可以了：

[linuxprobe@linuxprobe ~]$ ls /root

ls: cannot open directory /root: Permission denied

[linuxprobe@linuxprobe ~]$ **sudo** ls /root

anaconda-ks.cfg Documents initial-setup-ks.cfg Pictures Templates

Desktop Downloads Music Public Videos

效果立竿见影！但是考虑到生产环境中不允许某个普通用户拥有整个系统中所有命令的最高执行权（这也不符合前文提到的权限赋予原则，即尽可能少地赋予权限），因此ALL参数就有些不合适了。因此只能赋予普通用户具体的命令以满足工作需求，这也受到了必要的权限约束。如果需要让某个用户只能使用root管理员的身份执行指定的命令，切记一定要给出该命令的绝对路径，否则系统会识别不出来。我们可以先使用whereis命令找出命令所对应的保存路径，然后把配置文件第99行的用户权限参数修改成对应的路径即可：

[linuxprobe@linuxprobe ~]$ exit

logout

[root@linuxprobe ~]# whereis cat

cat: **/usr/bin/cat** /usr/share/man/man1/cat.1.gz /usr/share/man/man1p/cat.1p.gz

[root@linuxprobe ~]# visudo

96 ##

97 ## Allow root to run any commands anywhere

98 root ALL=(ALL) ALL

**99 linuxprobe ALL=(ALL) /usr/bin/cat**

在编辑好后依然是先保存再退出。再次切换到指定的普通用户，然后尝试正常查看某个文件的内容，此时系统提示没有权限。这时再使用sudo命令就可以顺利地查看文件内容了：

[root@linuxprobe ~]# su - linuxprobe

Last login: Thu Sep 3 15:51:01 CST 2017 on pts/1

[linuxprobe@linuxprobe ~]$ cat /etc/shadow

cat: /etc/shadow: Permission denied

[linuxprobe@linuxprobe ~]$ sudo cat /etc/shadow

root:$6$GV3UVtX4ZGg6ygA6$J9pBuPGUSgZslj83jyoI7ThJla9ZAULku3BcncAYF00Uwk6Sqc4E36MnD1hLtlG9QadCpQCNVJs/5awHd0/pi1:16626:0:99999:7:::

bin:\*:16141:0:99999:7:::

daemon:\*:16141:0:99999:7:::

adm:\*:16141:0:99999:7:::

lp:\*:16141:0:99999:7:::

sync:\*:16141:0:99999:7:::

shutdown:\*:16141:0:99999:7:::

halt:\*:16141:0:99999:7:::

mail:\*:16141:0:99999:7:::

operator:\*:16141:0:99999:7:::

games:\*:16141:0:99999:7:::

ftp:\*:16141:0:99999:7:::

nobody:\*:16141:0:99999:7:::

………………省略部分文件内容………………

大家千万不要以为到这里就结束了，刘遄老师还有更压箱底的宝贝。不知大家是否发觉在每次执行sudo命令后都会要求验证一下密码。虽然这个密码就是当前登录用户的密码，但是每次执行sudo命令都要输入一次密码其实也挺麻烦的，这时可以添加NOPASSWD参数，使得用户执行sudo命令时不再需要密码验证：

[linuxprobe@linuxprobe ~]$ exit

logout

[root@linuxprobe ~]# whereis poweroff

poweroff: **/usr/sbin/poweroff** /usr/share/man/man8/poweroff.8.gz

[root@linuxprobe ~]# visudo

96 ##

97 ## Allow root to run any commands anywhere

98 root ALL=(ALL) ALL

**99 linuxprobe ALL=NOPASSWD: /usr/sbin/poweroff**

这样，当切换到普通用户后再执行命令时，就不用再频繁地验证密码了，我们在日常工作中也就痛快至极了。

[root@linuxprobe ~]# su - linuxprobe

Last login: Thu Sep 3 15:58:31 CST 2017 on pts/1

[linuxprobe@linuxprobe ~]$ poweroff

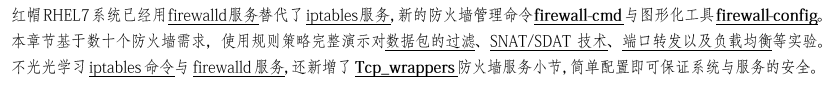
User root is logged in on seat0.

Please retry operation after closing inhibitors and logging out other users.

Alternatively, ignore inhibitors and users with 'systemctl poweroff -i'.

[linuxprobe@linuxprobe ~]$ sudo poweroff

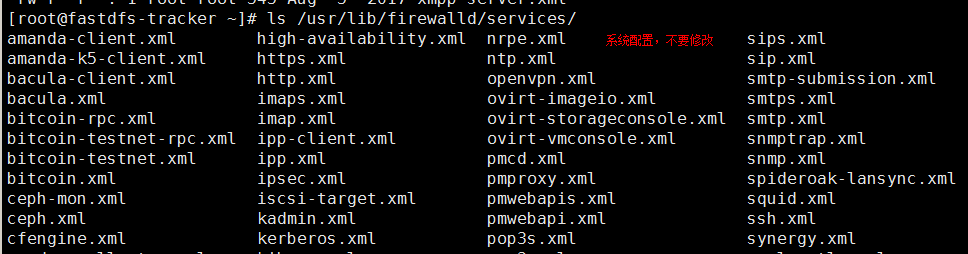
# 防火墙



官方文档介绍地址：

<https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/Security_Guide/sec-Using_Firewalls.html#sec-Introduction_to_firewalld1>

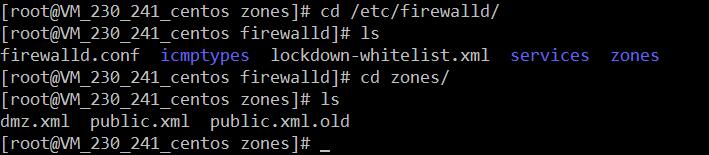
中文博客：<http://blog.csdn.net/xlgen157387/article/details/52672988>



**用户配置目录**

/etc/firewalld/

* 1



**3、如何自定义添加端口**

用户可以通过修改配置文件的方式添加端口，也可以通过命令的方式添加端口，注意，修改的内容会在/etc/firewalld/ 目录下的配置文件中还体现。

* 3.1、命令的方式添加端口

firewall-cmd --permanent --add-port=9527/tcp

参数介绍：

1、firewall-cmd：是Linux提供的操作firewall的一个工具；

2、--permanent：表示设置为持久；

3、--add-port：标识添加的端口；

另外，firewall中有Zone的概念，可以将具体的端口制定到具体的zone配置文件中。

例如：添加8010端口

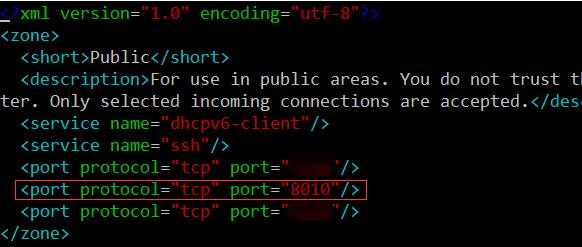
firewall-cmd --zone=public --permanent --add-port=8010/tcp

* 1

--zone=public：指定的zone为public；

* 1

添加结果如下：



如果–zone=dmz 这样设置的话，会在dmz.xml文件中新增一条。

* 3.2、修改配置文件的方式添加端口

<?xml version="1.0" encoding="utf-8"?>

<zone>

<short>Public</short>

<description>For use in public areas.</description>

<rule family="ipv4">

<source address="122.10.70.234"/>

<port protocol="udp" port="514"/>

<accept/>

</rule>

<rule family="ipv4">

<source address="123.60.255.14"/>

<port protocol="tcp" port="10050-10051"/>

<accept/>

</rule>

<rule family="ipv4">

<source address="192.249.87.114"/> 放通指定ip，指定端口、协议

<port protocol="tcp" port="80"/>

<accept/>

</rule>

<rule family="ipv4"> 放通任意ip访问服务器的9527端口

<port protocol="tcp" port="9527"/>

<accept/>

</rule>

</zone>

上述的一个配置文件可以很好的看出：

1、添加需要的规则，开放通源ip为122.10.70.234，端口514，协议tcp；

2、开放通源ip为123.60.255.14，端口10050-10051，协议tcp；/3、开放通源ip为任意，端口9527，协议tcp；

**1重启、关闭、开启firewalld.service服务**

service firewalld restart 重启

service firewalld start 开启

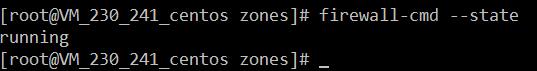
service firewalld stop 关闭

**2、查看firewall服务状态**

systemctl status firewalld

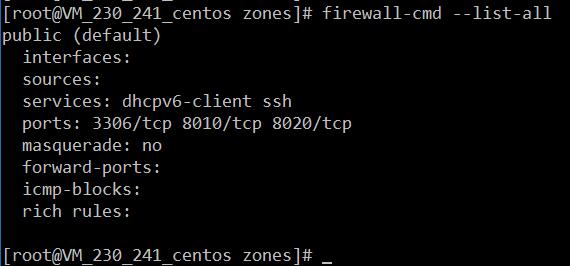
**3、查看firewall的状态**

firewall-cmd --state



**4、查看防火墙规则**

firewall-cmd --list-all



命令行终端是一种极富效率的工作方式，firewall-cmd是firewalld防火墙配置管理工具的CLI（命令行界面）版本。它的参数一般都是以“长格式”来提供的，大家不要一听到长格式就头大，因为RHEL 7系统支持部分命令的参数补齐，其中就包含这条命令（很酷吧）。也就是说，现在除了能用Tab键自动补齐命令或文件名等内容之外，还可以用Tab键来补齐表8-3中所示的长格式参数了（这太棒了）。

表8-3                                   firewall-cmd命令中使用的参数以及作用

|  |  |
| --- | --- |
| 参数 | 作用 |
| --get-default-zone | 查询默认的区域名称 |
| --set-default-zone=<区域名称> | 设置默认的区域，使其永久生效 |
| --get-zones | 显示可用的区域 |
| --get-services | 显示预先定义的服务 |
| --get-active-zones | 显示当前正在使用的区域与网卡名称 |
| --remove-source= | 将源自此IP或子网的流量导向指定的区域 |
| --remove-source= | 不再将源自此IP或子网的流量导向某个指定区域 |
| --add-interface=<网卡名称> | 将源自该网卡的所有流量都导向某个指定区域 |
| --change-interface=<网卡名称> | 将某个网卡与区域进行关联 |
| --list-all | 显示当前区域的网卡配置参数、资源、端口以及服务等信息 |
| --list-all-zones | 显示所有区域的网卡配置参数、资源、端口以及服务等信息 |
| --add-service=<服务名> | 设置默认区域允许该服务的流量 |
| --add-port=<端口号/协议> | 设置默认区域允许该端口的流量 |
| --remove-service=<服务名> | 设置默认区域不再允许该服务的流量 |
| --remove-port=<端口号/协议> | 设置默认区域不再允许该端口的流量 |
| --reload | 让“永久生效”的配置规则立即生效，并覆盖当前的配置规则 |
| --panic-on | 开启应急状况模式 |
| --panic-off | 关闭应急状况模式 |

与Linux系统中其他的防火墙策略配置工具一样，使用firewalld配置的防火墙策略默认为运行时（Runtime）模式，又称为当前生效模式，而且随着系统的重启会失效。如果想让配置策略一直存在，就需要使用永久（Permanent）模式了，方法就是在用firewall-cmd命令正常设置防火墙策略时添加--permanent参数，这样配置的防火墙策略就可以永久生效了。但是，永久生效模式有一个“不近人情”的特点，就是使用它设置的策略只有在系统重启之后才能自动生效。如果想让配置的策略立即生效，需要手动执行firewall-cmd --reload命令。

接下来的实验都很简单，但是提醒大家一定要仔细查看刘遄老师使用的是Runtime模式还是Permanent模式。如果不关注这个细节，就算是正确配置了防火墙策略，也可能无法达到预期的效果。

查看firewalld服务当前所使用的区域：

[root@linuxprobe ~]# firewall-cmd --get-default-zone

public

查询eno16777728网卡在firewalld服务中的区域：

[root@linuxprobe ~]# firewall-cmd --get-zone-of-interface=eno16777728

public

把firewalld服务中eno16777728网卡的默认区域修改为external，并在系统重启后生效。分别查看当前与永久模式下的区域名称：

[root@linuxprobe ~]# firewall-cmd --permanent --zone=external --change-interface=eno16777728

success

[root@linuxprobe ~]# firewall-cmd --get-zone-of-interface=eno16777728

public

[root@linuxprobe ~]# firewall-cmd --permanent --get-zone-of-interface=eno16777728

external

把firewalld服务的当前默认区域设置为public：

[root@linuxprobe ~]# firewall-cmd --set-default-zone=public

success

[root@linuxprobe ~]# firewall-cmd --get-default-zone

public

启动/关闭firewalld防火墙服务的应急状况模式，阻断一切网络连接（当远程控制服务器时请慎用）：

[root@linuxprobe ~]# firewall-cmd --panic-on

success

[root@linuxprobe ~]# firewall-cmd --panic-off

success

查询public区域是否允许请求SSH和HTTPS协议的流量：

[root@linuxprobe ~]# firewall-cmd --zone=public --query-service=ssh

yes

[root@linuxprobe ~]# firewall-cmd --zone=public --query-service=https

no

把firewalld服务中请求HTTPS协议的流量设置为永久允许，并立即生效：

[root@linuxprobe ~]# firewall-cmd --zone=public --add-service=https

success

[root@linuxprobe ~]# firewall-cmd --permanent --zone=public --add-service=https

success

[root@linuxprobe ~]# firewall-cmd --reload

success

把firewalld服务中请求HTTP协议的流量设置为永久拒绝，并立即生效：

[root@linuxprobe ~]# firewall-cmd --permanent --zone=public --remove-service=http

success

[root@linuxprobe ~]# firewall-cmd --reload

success

把在firewalld服务中访问8080和8081端口的流量策略设置为允许，但仅限当前生效：

[root@linuxprobe ~]# firewall-cmd --zone=public --add-port=8080-8081/tcp

success

[root@linuxprobe ~]# firewall-cmd --zone=public --list-ports

8080-8081/tcp

把原本访问本机888端口的流量转发到22端口，要且求当前和长期均有效：

流量转发命令格式为firewall-cmd --permanent --zone=**<区域>** --add-forward-port=port=<源端口号>:proto=**<协议>**:toport=**<目标端口号>**:toaddr=**<目标IP地址>**

[root@linuxprobe ~]# firewall-cmd --permanent --zone=**public** --add-forward-port=port=888:proto=**tcp**:toport=**22**:toaddr=**192.168.10.10**

success

[root@linuxprobe ~]# firewall-cmd --reload

success

在客户端使用ssh命令尝试访问192.168.10.10主机的888端口：

[root@client A ~]# ssh -p 888 192.168.10.10

The authenticity of host '[192.168.10.10]:888 ([192.168.10.10]:888)' can't be established.

ECDSA key fingerprint is b8:25:88:89:5c:05:b6:dd:ef:76:63:ff:1a:54:02:1a.

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

Warning: Permanently added '[192.168.10.10]:888' (ECDSA) to the list of known hosts.

root@192.168.10.10's password:**此处输入远程root管理员的密码**

Last login: Sun Jul 19 21:43:48 2017 from 192.168.10.10

firewalld中的富规则表示更细致、更详细的防火墙策略配置，它可以针对系统服务、端口号、源地址和目标地址等诸多信息进行更有针对性的策略配置。它的优先级在所有的防火墙策略中也是最高的。比如，我们可以在firewalld服务中配置一条富规则，使其拒绝192.168.10.0/24网段的所有用户访问本机的ssh服务（22端口）：

[root@linuxprobe ~]# firewall-cmd --permanent --zone=public --add-rich-rule="rule family="ipv4" source address="192.168.10.0/24" service name="ssh" reject"

success

[root@linuxprobe ~]# firewall-cmd --reload

success

在客户端使用ssh命令尝试访问192.168.10.10主机的ssh服务（22端口）：

[root@client A ~]# ssh 192.168.10.10

Connecting to 192.168.10.10:22...

Could not connect to '192.168.10.10' (port 22): Connection failed.

# 关于开机自启动的说明

可以使用前面修改/etc/rc.local文件添加命令的方式，也可以通过创建服务这种更专业的方式来开机自启动！！（**前提是别人已经写好了支持安装服务的脚本！！**）

这需要使用chkconfig命令（选学内容）

http://www.cnblogs.com/jimeper/archive/2013/03/12/2955687.html

|  |
| --- |
| [**linux自定义开机启动服务和chkconfig使用方法**](http://www.cnblogs.com/jimeper/archive/2013/03/12/2955687.html)  linux自定义开机启动服务和chkconfig使用方法  1. 服务概述 在linux操作系统下，经常需要创建一些服务，这些服务被做成shell脚本，这些服务需要在系统启动的时候自动启动，关闭的时候自动关闭。 将需要自动启动的脚本/etc/rc.d/init.d目录下，然后用命令chkconfig --add filename将自动注册开机启动和关机关闭。实质就是在rc0.d-rc6.d目录下生成一些文件连接，这些连接连接到/etc/rc.d /init.d目录下指定文件的shell脚本。  2. 手工创建服务 在/etc/rc.d/init.d目录下创建shell脚本，文件名auto\_run。 设置脚本的运行权限chmod +x auto\_run。 然后在rc0.d-rc6.d目录下分别创建文件连接。 ln -s /etc/rc.d/init.d/auto\_run /etc/rc.d/rc2.d/S99auto\_run ln -s /etc/rc.d/init.d/auto\_run /etc/rc.d/rc3.d/S99auto\_run ln -s /etc/rc.d/init.d/auto\_run /etc/rc.d/rc5.d/S99auto\_run ln -s /etc/rc.d/init.d/auto\_run /etc/rc.d/rc0.d/K01auto\_run ln -s /etc/rc.d/init.d/auto\_run /etc/rc.d/rc6.d/K01auto\_run  这样系统在启动的时候，就会运行auto\_run 并加上start参数，等同于执行命令auto\_run start。 在系统关闭的时候，就会运行auto\_run，并加上stop参数，等同于运行命令auto\_run stop。  创建连接的6条命令可以用命令chkconfig --add auto\_run来完成，这样就简单多了，还不容易出错。 下面就介绍一下chkconfig命令。 3. chkconfig命令用法 语法： chkconfig --list [name] chkconfig --add name chkconfig --del name chkconfig [--level levels] name <on|off|reset> chkconfig [--level levels] name    说明: chkconfig提供一种简单的命令行工具来帮助管理员对/etc/rc[0-6].d目录层次下的众多的符号链接进行直接操作。 此命令使用是由chkconfig命令在IRIX操作系统提供授权。不用在/etc/rc[0-6].d目录下直接维护配置信息，而是直接在/etc/rc[0-6]下管理链接文件。在运行级别的目录下的配置信息通知在将会初始启动哪些服务。 Chkconfig有五个很明确的功能：为管理增加一个新的功能、删除一个功能、列出当前服务的启动信息、改变一个服务的启动信息和检测特殊服务的启动状态。 当chkconfig没有参数运行时，它将显示其使用方法。如果只给出了一个服务名，它将检测这个服务名是否已经被配置到了当前运行级别中。如果已经配置，返回真，否则返回假。--level选项可以被用做查询多个运行级别而不仅仅是一个。 如果在服务名之后指定了on,、off或reset，chkconfig将改变指定服务的启动信息。On或off标记服务被打开或停止，尤其是在运行级别被改变时。Reset标记重置服务的启动信息。 默认情况下，on或off选项仅对2、3、4、5有影响，而 reset影响所有的运行级。--level选项可以被用于指定哪个运行级别接收影响。 注意：对于每个服务，每一个运行级都有一个开始角本和一个结束角本。当开或关一个运行级时，init不会重新开始一个已经运行的服务，也不会重新停止一个未运行的服务。 选项： --level levels 指定一个运行级别适合的操作。范围为0-7。 --add name 增加一个新的服务。 --del name 删除一个服务 --list name 显示服务的情况 RUNLEVEL FILES 每个通过chkconfig管理的服务在其init.d目录下的角本中都需要两行或多行的注释。第一行告诉chkconfig在默认情况下什么运行级别的服务可以开始，也就是所说的开始或结束的优先级别。如果服务没有默认的级别，建造将在所有运行级别中启动。a – 将用于代替运行级列表。第二个注释行包括对此服务的描述，可以通过反斜线符号扩展为多行。  示例，auto\_run的前三行如下： #!/bin/sh #chkconfig: 2345 80 90 #description:auto\_run 第一行，告诉系统使用的shell,所以的shell脚本都是这样。 第二行，chkconfig后面有三个参数2345,80和90告诉chkconfig程序，需要在rc2.d~rc5.d目录下，创建名字为 S80auto\_run的文件连接，连接到/etc/rc.d/init.d目录下的的auto\_run脚本。第一个字符是S，系统在启动的时候，运行脚本auto\_run，就会添加一个start参数，告诉脚本，现在是启动模式。同时在rc0.d和rc6.d目录下，创建名字为K90auto\_run的文件连接，第一个字符为K，系统在关闭系统的时候，会运行auto\_run，添加一个stop，告诉脚本，现在是关闭模式。 注意上面的三行中，第二，第三行是必须的，否则在运行chkconfig --add auto\_run时，会报错。 4. 常见的错误 “服务不支持 chkconfig”： 请注意检查脚本的前面，是否有完整的两行： #chkconfig: 2345 80 90 #description:auto\_run 在脚本前面这两行是不能少的，否则不能chkconfig命令会报错误。 如果运行chkconfig老是报错，如果脚本没有问题，我建议，直接在rc0.d~rc6.d下面创建到脚本的文件连接来解决，原理都是一样的。 |