day1

【学习目标】

1.掌握基础常用的 linux 命令 1.5-2

2.熟练使用 shell 脚本编程 1.5

3.学习 Makefile 的编写 1

【linux 发展历史】

一、 Uninx 的起源

1969 年，由 KenThompson 在 AT&T 贝尔实验室实现的。使用的是用汇编语言。

1970 年，KenThompson 和 DennisRitchie(丹尼斯。里奇)使用 C 语言对整个系统进行了再加工和编写，使得 Unix 能够很容易的移植到其他硬件的计算机上。

二、 Unix 的推广-从学校到企业

起初 AT&T 没有把 Unix 作为正式商品，以分发许可证的方法，对 Unix 仅仅收取很少的费用，就把 Unix 的源代码被散发到各个大学。由于 Unix 收费少，因此，很多厂商就选择了 Unix 作为他们生产的计算机使用的操作系统。

三、 Unix 的两大分支：AT&T UNIX System V 和 BSD Unix。

到了 70 年代中后期，在 Unix 发展到了版本 6 之后，AT&T 认识到了 Unix 的价值，成立了 Unix 系统实验室（UnixSystemLab,USL）来继续发展 Unix。

几乎在同时，加州大学伯克利分校计算机系统研究小组（CSRG）对 Unix 进行研究，做了大量改进工作，组成一个完整的 Unix 系统 ——BSDUnix（BerkeleySoftwareDistribution），向外发行。

BSDUnix 有很大的影响力，例如美国国防部的项目—ARPANET（阿帕网），ARPANET 今天发展成为了 Internet，而 BSDUnix 中最先实现了 TCP/IP 协议(网络编程)，使 Internet 和 Unix 紧密结合在一起。

AT&T 吸收了 BSDUnix 中已有的各种先进特性，并结合其本身的特点，推出了 UnixSystemV 版本之后，形成了两大分支。

在 1992 年，Unix 系统实验室指控 BSDI—— 一家发行商业 BSDUnix 的公司，违反了 AT&T 的许可权，发布自己的 Unix 版本。

后来，Unix 系统实验室被 AT&T 卖给了 Novell 公司，Novell 不打算陷入这样的法律纷争中，因此就采用了比较友好的做法。伯克利的 CSRG 被允许自由发布 BSD，但是其中来自于 AT&T 的代码必须完全删除。

Unix 主要有 Sun 的 Solaris、IBM 的 AIX， HP 的 HP-UX，以及 x86 平台的的 SCO Unix/Unixware。

四、 自由软件基金会（Free Software Foundation，FSF）

是一个倡导自由软件的国际性非盈利组织。由 Richard Stallman 在 1984 年建立。

五、 GNU 计划

（GNU's Not Unix）是由 Richard Stallman 在 1983 年 9 月 27 日公开发起的。它的目标是发展一个类似 UNIX，完全自由的操作系统。

六、 GPL

(General Public License，通用公共许可协议)是一种版权形式，是 Richard Stallman 在开放源代码软件发行的实践中，逐渐总结出的一套保护自由软件的条款，称之为 GPL。当人们提起商业软件版权时，总会用到 Copyright，而在 GPL 中，人们则使用“CopyLeft”。

中心意思：自由软件由开发者提供源代码，任何用户都有权使用、拷贝、扩散、修改该软件，同时用户也有义务将自己修改过的程序代码公开。允许用户在分发过程中收取一定的费用。但是，用户在再分发时，要保证新用户能取得源代码的权力。保证新用户与自己相同，在得到软件时，同时得到同自己一样的权力。在 GPL 下，不存在“盗版”。但有一点，用户不能将软件据为己有(申请软件产品“专利”等)，因为这将侵犯 GPL 版权。

自由软件之父-------Richard Stallman

七、 GNU/linux

到 90 年代，已经发现或者完成了构建一个操作系统所需的，除了内核之外的所有主要成分。

1991 年,由 Linus Torvalds 开发了一个自由的内核。1992 年，把 Linux 和几乎完成的 GNU 系统结合起来，就构成了一个完整的操作系统：一个基于 Linux 的 GNU 系统-GNU/linux. 值得注意的是 Linux 并没有包括 Unix 源码。它是按照公开的 POSIX 标准重新编写的。

Linux 商业化的有 RedHat Linux 、SuSe Linux、slakeware Linux、国内的红旗等，还有 Turbo Linux.

八、 Unix 和 Linux 的区别和联系

联系:

1）Linux 思想源于 Unix.(Linus Torvalds 以 Unix 为原型，开发的 linux)

2）Linux 产品成功的模仿了 UNIX 系统和功能

两大区别:

1）linux 是开发源代码的自由软件．而 unix 是对源代码实行知识产权保护的传统商业软件。

2）UNIX 系统大多是与硬件配套的，而 Linux 则可运行在多种硬件平台上.

（Unix 针对大型应用。在性能上，linux 没有那么全面,主要针对个人和小型应用。）

使用 Unix 的环境，比如银行、电信、民航部门，那一般都是固定机型的 Unix。比如电信里 SUN 的居多，民航里 HP 的居多，银行里 IBM 的居多。

【操作系统的功能】

向上提供接口，向下管理硬件

层次架构

应用层：应用程序、命令\可执行文件

---

shell 解析器

内核层:

系统调用接口：

1.文件管理：实现文件数据的读写或者文件的打开、关闭等

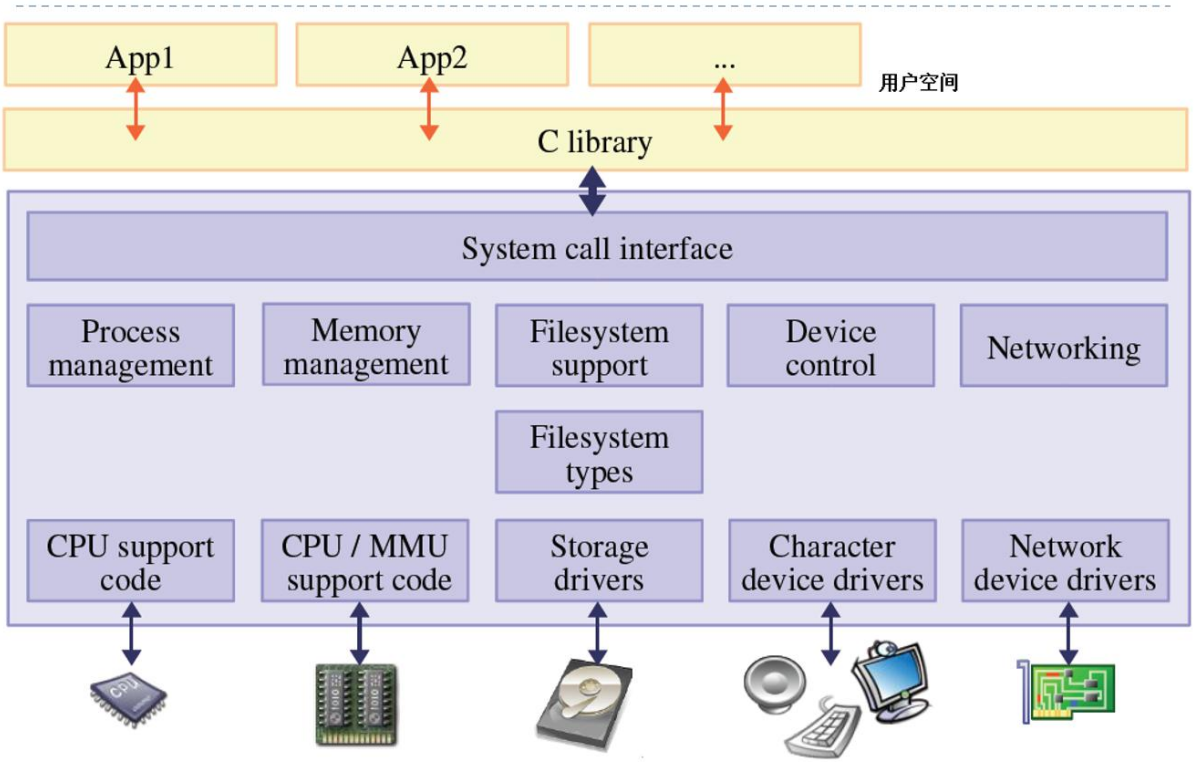
2.进程管理：实现进程的创建、关闭、资源回收等

3.网络管理：实现网络设备的相互连接和数据的读写

4.内存管理：内存资源的申请、释放等。。。

5.设备管理：设备驱动相关接口

硬件层：显示屏、灯、键盘、鼠标



Linux 内核支持多用户、多任务模式运行

多用户：同时有多个用户访问系统

多任务：某个时刻有多个程序运行

linux 内核版本的查看：uname -r

5.4.0-125-generic

主版本-次版本号-修订次数-编译次数-注释

Shell 是一个命令行解释器，它使得用户能够与操作系统进行交互。

shell 示例：

sudo shutdown -h now //立即关机

sudo shutdown -r now //立即重启

sudo shutdown -h +45 //45 分钟之后关机

使用 vi 编辑器的必要性：

当操作系统出现问题甚至半崩溃，vi 编辑器就是这种情况下唯一可以使用的文本编辑工具

【软件包的管理】

软件包发展：

最初，基于 Linux 系统的开发者在完成应用程序开发后，将很多二进制文件发给用户，用户使用之前需要将很多程序逐个安装。

因此，Debian Linux 首先提出“软件包”的管理机制——Deb 软件包，将应用程序的二进制文件、配置文档、man/info 帮助页面等文件合并打包在一个文件中，用户使用软件包管理器直接操作软件包，完成获取、安装、卸载、查询等操作。

随即，Redhat Linux 基于这个理念推出了自己的软件包管理机制——Rpm 软件包。当然，Redhat 采用了自己的打包格式生成 Rpm 包文件，由 Rpm 包管理器负责安装、维护、查询，甚至是软件包版本管理。不过，由于 Redhat Linux 系统的普及，Rpm 软件包被广泛使用。甚至出现第三方开发的软件管理工具，专门管理 Rpm 格式的软件包。

deb 软件包介绍

Debian 包文件包含了二进制可执行文件、库文件、配置文件和 man/info 帮助页面等文档。通常 Debian 包文件的后缀为“.deb”，因此称为“Deb 软件包”。Ubuntu 有两种类型的软件包：二进制软件包（deb）和源码包（deb-src）。

二进制软件包（Binary Packages）：它包含可执行文件、库文件、配置文件、man/info 页面、版权声明和其它文档。

源码包（Source Packages）：包含软件源代码、版本修改说明、构建指令以及编译工具等。先由 tar 工具归档为.tar.gz 文件，然后再打包成.dsc 文件。

在用户不确定一个软件包类型时，可以使用 file 命令查看文件类型。例如下面命令用于证实一个软件包的文件类型是否是 Deb 软件包文件：

```
file sl_3.03-17build2_amd64.deb
sl_3.03-17build2_amd64.deb: Debian binary package (format 2.0)
```

sl_3.03-17build2_amd64.deb

名字_版本.修订版本-编译次数_架构.deb

deb 软件包管理工具分类：

Linux 为用户提供了不同层次和类型的软件包管理工具，根据用户交互方式的不同，可以将常见的软件包管理工具分为三类。

1.命令行:在命令行模式下完成软件包管理任务。为完成软件包的获取、查询、软件包依赖性检查、安装、卸载等任务，需要使用各自不同的命令 dpkg、apt

2.文本窗口界面:在文本窗口模式中，使用窗口和菜单可以完成软件包管理任务

3.图形界面

dpkg 和 apt:

1. dpkg

> 是最早的 Deb 包管理工具，它在 Debian 一提出包管理模式后就诞生了。使用 dpkg 可以实现软件包的安装、编译、卸载、查询，以及应用程序打包等功能。但是由于当时 Linux 系统规模和 Internet 网络条件的限制，没有考虑到操作系统中软件包存在如此复杂的依赖关系，以及帮助用户获取软件包（获取存在依赖关系的软件包）。因而，为了解决软件包依赖性问题和获取问题，就出现了 APT 工具。

dpkg 是 Ubuntu Linux 中最基本的命令行软件包管理工具，用于安装、编译、卸载和查询 Deb 软件包。

缺陷:

第一，不能主动从镜像站点获取软件包；

第二，安装软件包时，无法检查软件包的依赖关系。

因此，在对一个软件组件的依赖关系不清楚的情况下，建议使用 APT 软件包管理器。

除非用户对软件包的依赖关系非常清楚，再使用 dpkg。

dpkg 相关命令：

dpkg -i <package> 安装一个在本地文件系统上存在的 Debian 软件包

dpkg -r <package> 移除一个已经安装的软件包

dpkg -P <package> 移除已安装软件包及配置文件

dpkg -L <package> 列出安装的软件包清单

dpkg -s <package> 显出软件包的安装状态

dpkg-reconfigure <package> 重新配置一个已经安装的软件包

2. APT

APT 系列工具可能是 Deb 软件包管理工具中功能最强大的。Ubuntu 将所有的开发软件包存放在 Internet 上的许许多多镜像站点上。用户可以选择其中最适合自己的站点作为软件源。然后，在 APT 工具的帮助下，就可以完成所有的软件包的管理工作，包括维护系统中的软件包数据库、自动检查软件包依赖关系、安装和升级软件包、从软件源镜像站点主动获取相关软件包等。常用的 APT 实用程序有：apt-get、apt-cache、apt-file、apt-cdrom 等。

APT（Advanced Packaging Tool）是 Ubuntu Linux 中功能最强大的命令行软件包管理工具，用于获取、安装、编译、卸载和查询 Deb 软件包，以及检查软件包依赖关系。

工作原理：

Ubuntu 采用集中式的软件仓库机制，将各式各样的软件包分门别类地存放在软件仓库中，进行有效地组织和管理。然后，将软件仓库置于许许多多的镜像服务器中，并保持基本一致。这样，所有的 Ubuntu 用户随时都能获得最新版本的安装软件包。因此，对于用户，这些镜像服务器就是他们的软件源（reposity）。

使用 APT 工具的前置条件：

1. 保证虚拟机可以访问外网 d'k

1.保证虚拟机能上网：ping www.baidu.com

2.不能上网，做下面配置：

a.虚拟机-》设置-》网络适配器-》net 模式

b.左下角拓展选项-》设置-》网络

c.点击'+'->添加

d.点击右上角网络图标-》有线连接-》选择自己刚才那个配置

注意：如果有的同学网络图标不显示，则需要重启网络相关服务：

1.sudo service network-manager stop

2.sudo rm /var/lib/NetworkManager/NetworkManager.state

3.sudo service network-manager start

4.sudo gedit /etc/NetworkManager/NetworkManager.conf

在运行完上一行的命令后会弹出一个文本文件，将其中的 false 改成 true

5.sudo service network-manager restart

注意：如果上面方法还不行：

编辑-》虚拟网络编辑器-》还原默认网络

2. /etc/apt/sources.list 文件中镜像源是准确可使用的

1.备份源文件

sudo cp /etc/apt/sources.list /etc/apt/sources.list.bak

2.打开文件

sudo gedit /etc/apt/sources.list

3.把文件内容删除，换上以下内容：

deb http://mirrors.aliyun.com/ubuntu/ bionic main restricted universe multiverse
deb http://mirrors.aliyun.com/ubuntu/ bionic-security main restricted universe multiverse
deb http://mirrors.aliyun.com/ubuntu/ bionic-updates main restricted universe multiverse
deb http://mirrors.aliyun.com/ubuntu/ bionic-proposed main restricted universe multiverse
deb http://mirrors.aliyun.com/ubuntu/ bionic-backports main restricted universe multiverse
deb-src http://mirrors.aliyun.com/ubuntu/ bionic main restricted universe multiverse
deb-src http://mirrors.aliyun.com/ubuntu/ bionic-security main restricted universe multiverse
deb-src http://mirrors.aliyun.com/ubuntu/ bionic-updates main restricted universe multiverse
deb-src http://mirrors.aliyun.com/ubuntu/ bionic-proposed main restricted universe multiverse
deb-src http://mirrors.aliyun.com/ubuntu/ bionic-backports main restricted universe multiverse

4.保存退出

5.终端执行：

sudo apt-get update

6.下载软件

给大家的测试软件：

sudo apt-get install fortune

2.cowsay 命令

用 ASCII 字符打印牛，羊等动物，还有个 cowthink，这个是奶牛想，那个是奶牛说，哈哈，差不多

安装 sudo apt-get install cowsay

运行 cowsay "I am not a cow, hahaha"

APT 常用的管理命令：

1. apt-get：用于管理软件包，包括安装、卸载、升级等操作
2. apt-cache：用于查询软件包信息
3. apt-show-versions：用于显示系统中软件包版本信息；

APT-GET 的使用：

格式：

apt-get subcommands [-d | -f | -m | -q | --purge | --reinstall | - b | - s | - y | - u | - h | -v] pkg

subcommands:

update:下载更新软件包列表信息

upgrade:将系统中所有软件包升级到最新的版本

install:下载所需软件包并进行安装配置

remove:卸载软件包

source：下载源码包

clean：清空安装包存放缓存区的内容

注意：apt 工具获取的软件包存放的位置：/var/cache/apt/archives

选项：

-d:仅仅下载软件包，不安装

-f:修复系统中存在的软件包依赖性问题

--purge:与 remove 子命令一起使用，完全卸载软件包

--reinstall:与 install 子命令一起使用，重新安装软件包

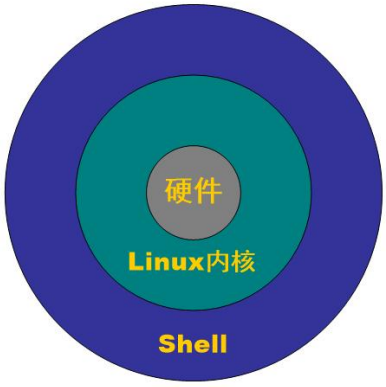
共享文件夹的创建

1. 在 windows 下准备一个合适的文件夹用于数据共享
2. 虚拟机-》设置-》选项-》共享文件夹
3. 选择总是启用
4. 添加共享文件夹

5. ls /mnt/hgfs 查看添加的共享文件夹是否存在
6. 如果/mnt/hgfs 下没有共享文件夹，参照下面步骤：

> 使用以下办法解决(root 权限)：
> 1. vmware-hgfsclient 命令查看当前有哪些共享的目录，这里我只使用了 shared 文件夹
>
> 2. 使用 mount -t vmhgfs .host:/shared /mnt/hgfs 命令挂载该共享文件夹(注意：带.号的哦)，其中.host:/Documents 是共享名，只需把 Documents 换成
> 使用 vmware-hgfsclient 命令得到的目录，/mnt/hgfs 是挂载点
>
> 3. 到此为止是可以使用该共享文件夹了，但每次都得重复 mount 一次，所以需要设置为随机启动后自动挂载
> 编辑 /etc/fstab，添加下面一行
> .host:/ /mnt/hgfs fuse.vmhgfs-fuse allow_other 0 0
> 补充：
> 如果显示
> Error: cannot mount filesystem: No such device
> ubuntu 则先执行 sudo apt-get install open-vm-dkms
> 然后再执行 3

shell 命令介绍



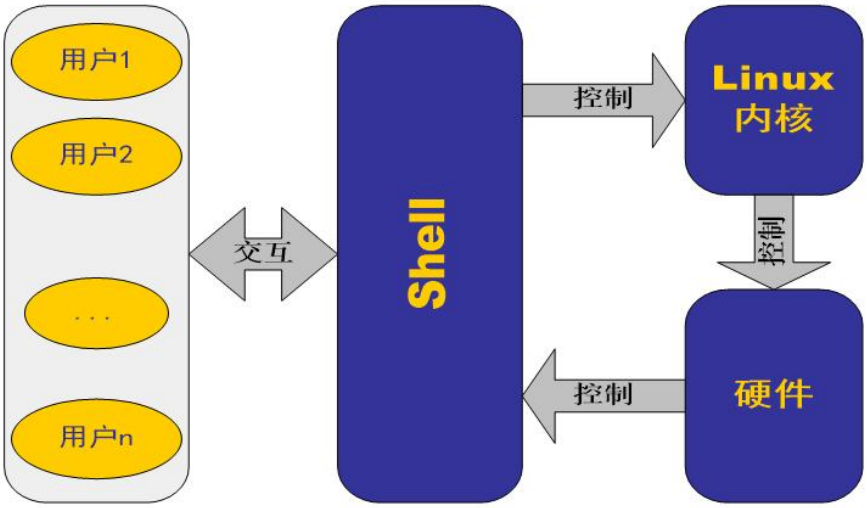
定义

命令是用户向系统内核发出控制请求，与之交互的文本流

shell

Shell 是一个命令行解释器，将用户命令解析为操作系统所能理解的指令，实现用户与操作系统的交互。

Shell 为操作系统提供了内核之上的功能，直接用来管理和运行系统。



shell 脚本

当需要重复执行若干命令，可以将这些命令集合起来，加入一定的控制语句，编辑成为 Shell 脚本文件，交给 Shell 批量执行。

shell 分类

1.Bourne Shell（简称 sh）：Bourne Shell 由 AT&T 贝尔实验室的 S.R.Bourne 开发，也因开发者的姓名而得名。它是 Unix 的第一个 Shell 程序，早已成为工业标准。目前几乎所有的 Linux 系统都支持它。不过 Bourne Shell 的作业控制功能薄弱，且不支持别名与历史记录等功能。目前大多操作系统是将其作为应急 Shell 使用。

2.C Shell（简称 csh）：C Shell 由加利福尼亚大学伯克利分校开发。最初开发的目的是改进 Bourne Shell 的一些缺点，并使 Shell 脚本的编程风格类似于 C 语言，因而受到广大 C 程序员的拥护。不过 C Shell 的健壮性不如 Bourne Shell。

3.Korn Shell（简称 ksh）：Korn Shell 由 David Korn 开发，解决了 Bourne Shell 的用户交互问题，并克服了 C Shell 的脚本编程怪癖的缺点。Korn Shell 的缺点是需要许可证，这导致它应用范围不如 Bourne Shell 广泛。

4.Bourne Again Shell（简称 bash）：Bourne Again Shell 由 AT&T 贝尔实验室开发，是 Bourne

Shell 的增强版。随着几年的不断完善，已经成为最流行的 Shell。它包括了早期的 Bourne Shell 和 Korn Shell 的原始功能，以及某些 C Shell 脚本语言的特性。此外，它还具有以下特点：能够提供环境变量以配置用户 Shell 环境，支持历史记录，内置算术功能，支持通配符表达式，将常用命令内置简化。

Shell 基本命令格式

1. 命令行提示符：

通常 Shell 命令提示符采用以下的格式：

> ubuntu@ubuntu:~$
> 用户名@计算机名：路径名
> username：用户名，显示当前登录用户的账户名；
> hostname：主机名，显示登录的主机名，例如若远程登录后，则显示登录的主机名；
> direction：目录名，显示当前所处的路径，当在根目录下显示为“/”，当在用户主目录下显示为“~”；

2. 命令格式：

通常一条命令包含三个要素：命令名称、选项、参数。命令名称是必须的，选项和参数都可能是可选项。命令格式如下所示：

```
$ Command [-Options] Argument1 Argument2 ...
  指令    选项    参数 1    参数 2...
```

$：Shell 提示符，如果当前用户为超级用户，提示符为“#”，其他用户的提示符均为“$”；

注意：切换到管理员用户：su　退出：exit

Command：命令名称，Shell 命令或程序，严格区分大小写，例如设置日期指令为 date 等；

Options：命令选项，用于改变命令执行动作的类型，由“-”引导，可以同时带有多个选项；

Argument：命令参数，指出命令作用的对象或目标，有的命令允许带多个参数。

注意事项：

1.一条命令的三要素之间用空格隔开；

2.若将多个命令在一行书写，用分号（;）将各命令隔开；

3.如果一条命令不能在一行写完，在行尾使用反斜杠（\）标明该条命令未结束。

bash 特色功能

1. 补齐命令与文件名

在使用 Shell 命令时，很多用户会经常遇到命令或文件名没有记全的情况。Bash Shell 的命令和文件名补齐功能会帮助用户。在输入命令或文件名的前几个字符后，按 TAB 键或 ESC 键自动补齐剩余没有输入的字符串。如果存在多个命令或文件有相同前缀，Shell 将列出所有相同前缀的命令或文件。Shell 给出的提示信息，帮助用户回忆和完成输入。之后等待用户输入足够的字符。

需要说明的是，连续按两下 TAB 键或 ESC 键，用于命令补齐；按下一次 TAB 键，用于文件名补齐。

2. 查询命令历史

用户在 Shell 下的操作是有很大连续性的，曾经输入的命令可能需要多次使用。当用户在操作中发现问题，需要查看曾经执行过的操作。Bash 将用户曾经键入的命令序列保存在一个命令历史表中。按上箭头键，便可逐条向上追溯曾经使用过的命令，并显示在命令提示符处；按下箭头键，便可向下查询命令历史。

除了使用上下键，可以翻阅历史命令外，Bash Shell 还提供了 history 命令。该命令将命令历

史表按列表形式，从记录号 1 开始，一次性全部显示出来。

使用格式：

```
history n：显示最近的 n 条命令
```

history 只能记录有限条的历史命令，默认保留 500 条命令。Bash Shell 将历史命令容量保存在环境变量 HISTSIZE 中。使用“echo $HISTSIZE”查看当前历史命令容量；通过直接赋值的方法，修改这个环境变量。

```
ubuntu@ubuntu:~$ echo $HISTSIZE  #在终端打印变量数值
1000
ubuntu@ubuntu:~$ HISTSIZE=1500#修改变量数值
ubuntu@ubuntu:~$ echo $HISTSIZE
1500
```

3. 给命令起别名

格式：

```
alias 别名='命令名' #起别名
unalias 别名：取消别名
ex:
wdl@UbuntuFisher:~$ alias
alias  dirlist='ls -l'
alias  ls='ls --color=auto'
wdl@UbuntuFisher:~$ unalias  dirlist
wdl@UbuntuFisher:~$ alias
alias  ls='ls --color=auto'
```

alias 命令在不带任何参数情况下，默认为列出当前已定义的别名。如果打算取消某个别名，可以使用 unalias 命令。

shell 脚本中的特殊字符

1. 通配符

当需要用命令处理一组文件，例如 file1.txt、file2.txt、file3.txt……，用户不必一一输入文件名，可以使用 Shell 通配符。

通配符	含义	实例
星号（*）	匹配任意长度的字符串	用file_*.txt，匹配file_wang.txt、file_Lee.txt、file3_Liu.txt
问号（?）	匹配一个长度的字符	用flie_?.txt，匹配file_1.txt、file1_2.txt、file_3.txt
方括号（[...]）	匹配其中指定的一个字符	用file_[otr].txt，匹配file_o.txt、file_r.txt和file_t.txt
方括号（[-]）	匹配指定的一个字符范围	用file_[a-z].txt，匹配file_a.txt、file_b.txt，直到file_z.txt
方括号（[^...]）	除了其中指定的字符，均可匹配	用file_[^otr].txt，除了file_o.txt、file_r.txt和file_t.txt的其他文件

2. 管道符

管道可以把一系列命令连接起来，意味着第一个命令的输出将作为第二个命令的输入，通过管道传递给第二个命令，第二个命令的输出又将作为第三个命令的输入，以此类推。就像通过使用“|”符连成了一个管道。

wc 参数 目标文件：统计文件中行号、单词个数以及字符个数

-l:统计行号

-c:字符个数

-w:单词个数

ls|wc -w

例子：

ls|wc -w:查看当前路径下文件的个数

3. 输入/输出重定向

输入/输出重定向是改变 Shell 命令或程序默认的标准输入/输出目标，重新定向到新的目标。

Linux 中默认的标准输入定义为键盘，标准输出定义为终端窗口。

用户可以为当前操作改变输入或输出，迫使某个特定命令的输入或输出来源为外部文件。

重定向符	含义	实例
> file	将file文件重定向为输出源，新建模式	ls /usr > Lsoutput.txt，将ls /usr的执行结果，写到Lsoutput.txt文件中，若有同名文件将被删除
>> file	将file文件重定向为输出源，追加模式	ls /usr >> Lsoutput.txt，将ls /usr的执行结果，追加到Lsoutput.txt文件已有内容后
< file	将file文件重定向为输入源	wc < file1，将file1中的内容作为输入传给wc命令
2> 或 &>	将由命令产生的错误信息输入到	ls noexistingfile.txt 2> err.log，使用ls命令，查看一个不存在的文件时，将系统错误提示保存在err.log文件中

基本系统维护命令

1. 定时关机：sudo shutdown -h +时间
2. 定时关机：sudo shutdown -r +时间
3. 修改用户密码：sudo passwd 用户名
4. 更改为管理员模式：su
5. 退出管理员模式：exit
6. 终端输出指定信息：echo 目标内容

   echo -n 目标内容 :输出内容后不换行

7. clear:清屏
8. date:显示日期
9. df:查看磁盘使用情况
    a. -a:查看实际物理内存使用情况

```
/dev/sda1       46260208 23266604 20880240   53% /
```

    b. -h :以兆为单位查看
    c. -k:以千字节为单位查看
    d. -T:查看磁盘使用情况时附带查看磁盘文件类型
10. mount:挂载设备命令

    umount:取消挂载

    以安装 nfs 服务为例子去感受挂载命令的使用:

nfs 服务安装：

nfs 服务：网络文件系统(Network File System)

作用：开发板通过网络的方式远程从 ubuntu 服务器端挂载跟文件系统

1. 安装 nfs 服务器端

sudo apt-get install nfs-kernel-server

2. 修改 nfs 服务的配置文件

打开 sudo vi /etc/exports,在配置文件的最后一行添加以下内容

/home/ubuntu/nfs/rootfs *(rw,sync,no_root_squash,no_subtree_check)

解释：

/home/ubuntu/nfs/rootfs：跟文件系统的路径，修改为自己的路径

* ---> 指所有的用户

rw ---> 对跟文件系统可读可写的权限

sync ---> 同步文件

no_root_squash ---> 如何客户端为 root 用户，那对文件系统有 root 的权限

no_subtree_check ---> 不检查子目录的权限

注：

1> *(: 之间不可以出现空格

2> rw,sync,no_root_squash,no_subtree_check

逗号后边不允许有空格

3> 前边不要加#，#是注释

3. 创建 nfs 文件夹

1> 创建 nfs 文件夹

cd ~

mkdir nfs

chmod 777 nfs

2> 拷贝跟文件系统的压缩包到 nfs 目录下

可以是共享文件夹或直接拖拽。

rootfs-ok.tarr.xz ----》跟文件系统的压缩包

3> 使用 tar 对跟文件系统进行解压缩

tar -vxf rootfs-ok.tar.xz

解压缩之后会得到一个 rootfs 的文件夹

注：不要在 windows 下进行解压缩，windows 不支持软连接文件

4. 重启 nfs 服务使其立即生效

sudo service nfs-kernel-server restart

注：只要修改配置文件，就需要重启服务

1. 完成 nfs 服务的安装和测试，并且把它当作作业发布在 CSDN
2. 预习 shell 脚本(18 条消息) shell 脚本的使用入门（超全）_IT_cdc 的博客-CSDN 博客_shell 脚本入门
3. 回忆 c 语言关键字的使用:

    static extern volitile const break continue