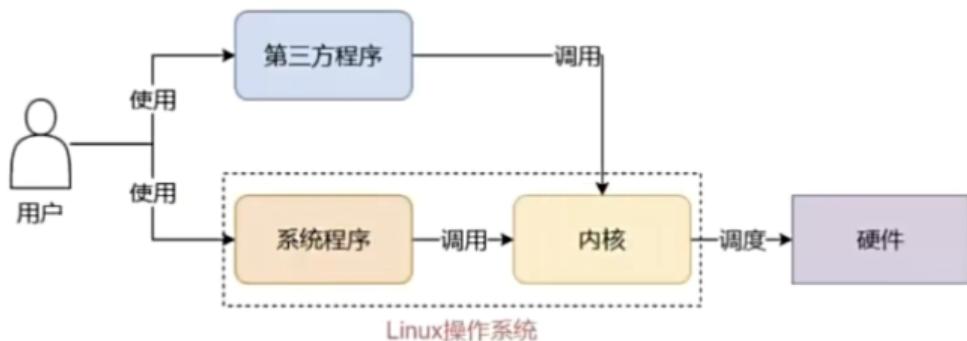


初识linux

Linux内核

linux系统的组成如下：

- Linux系统内核
 - 系统级应用程序
-
- 内核提供系统最核心的功能，如：调度CPU、调度内存、调度文件系统、调度网络通讯、调度IO等。
 - 系统级应用程序，可以理解为出厂自带程序，可供用户快速上手操作系统，如：文件管理器、任务管理器、图片查看、音乐播放



虚拟机

1. 什么是虚拟机？

通过虚拟化技术，在电脑内，虚拟出计算机硬件，并给虚拟的硬件安装操作系统，即可得到一台虚拟的电脑，称之为虚拟机。

2. 为什么要使用虚拟机？

学习Linux系统，需要有Linux系统环境。

我们不能给自己电脑重装系统为Linux，所以通过虚拟机的形式，得到可以用的Linux系统环境，供后续学习使用。

虚拟机快照

通过快照可以实现虚拟机状态回滚。

虚拟机的克隆

右键虚拟机--管理--克隆

可以在本机克隆，也可以克隆到另一台计算机上

虚拟机的迁移删除

虚拟系统安装好了，它的本质就是文件，因此迁移和删除很方便，直接对文件夹进行操作就行了。

Linux基础命令

Linux的目录结构

Linux的目录结构是一个属性结构。Linux没有盘符的概念，只有一个根目录/，所有文件都在它下面。

在windows系统中，路径之间的层级关系用\表示，而在Linux中，路径之间的层级关系用/来表示

linux下一切皆为文件，就算是硬件（如cpu）也被当作文件进行存储

Linux命令入门

Linux命令基础

无论是什么命令，用于什么用途，在Linux中，命令有其通用的格式：

command [-options] [parameter]

- command: 命令本身
- -options: [可选，非必填]命令的一些选项，可以通过选项控制命令的行为细节
- parameter: [可选，非必填]命令的参数，多数用于命令的指向目标等

语法中的[]，表示可选的意思

示例：

- ls -l /home/iTheima, ls是命令本身，-l是选项，/home/iTheima是参数
 - 意思是以列表的形式，显示/home/iTheima目录内的内容

ls命令入门

直接输入ls命令，表示列出当前工作目录(默认HOME目录)下的内容。

HOME目录：操作系统允许有多用户。每个Linux操作用户在Linux系统的个人账户目录，路径在`/home/用户名`

`ls`命令格式：`ls [-a -l -h] [Linux路径]`，中括号表示可选

- `-a`：表示列出所有文件（包括隐藏的文件），而在Linux中，以`.`开头的是隐藏文件
- `-l`：表示以列表（竖向）的形式展示内容，并展现更多信息
- `-a/-al/-la`：这三种都表示以列表形式展示所有文件
- `-h`：必须和`-l`一起使用，用来更细致的展现文件的大小

命令是可以组合使用的，比如`ls -lah`，等同于`ls -a -l -h`

`ls -R`：递归的查看

`tree` 目录：树状显示目录

目录切换相关命令

通过`cd`更改当前工作目录，不写路径表示回到`home`目录

`pwd` 命令查看当前所在工作目录的绝对路径。（print work directory）

路径相关

绝对路径：以**根目录为起点**，描述路径的一种写法，路径描述以`/`开头

相对路径：以**当前目录为起点**，路径描述不需要`/`开头

特殊表示：

- `.`：表示当前目录
- `..`：表示上一级目录，`cd/..` 可返回上两级目录
- `~`：`HOME`目录

创建目录

`mkdir [-p] Linux路径`

`mkdir`：make directory

- 参数必填
- `-p`可选，表示自动创建不存在的父目录，适用于创建连续多层级的目录

文件操作命令

`touch` Linux路径 创建文件

`cat [-n]` Linux路径 查看文件内容，全部显示，不能修改。-n表示显示行号

`cat ... | more`：分页浏览；enter来显示下一行，空格显示下一页

`more` Linux路径 查看文件内容，支持翻页。空格翻页，q退出查看

less：

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

操作	功能说明
空白键	向下翻动一页；
[pagedown]	向下翻动一页
[pageup]	向上翻动一页；
/字串	向下搜寻『字串』的功能；n：向下查找；N：向上查找；
?字串	向上搜寻『字串』的功能；n：向上查找；N：向下查找；
q	离开 less 这个程序；

✓ 基本语法
less 要查看的文件
✓ 操作说明

✓ 应用实例
案例：采用less查看一个大文件文件 /opt/杂文.txt

`cp [-r]` 参数1 参数2：

- -r：用于复制文件夹使用，表示递归
- 参数一：被复制
- 参数二：复制目的地

\cp：用法同上，如果文件名相同覆盖目标文件夹中的文件

`mv` 参数1 参数2：移动文件/文件夹。在将文件（夹）移动至文件（夹）时，如果目标文件不存在，自身进行改名后移动

`rmdir` 目录名：只能删除空目录

`rm [-r -f]` 参数1 ... 参数n：

- -r用于删除文件夹
- -f表示force，强制删除（不会提示确认信息）

普通用户删除不会弹出提示，root会提示，所以一般用户用不到-f

- 补充：用su -root切换root，用exit退出
- 参数用空格隔开
- rm 命令支持通配符：
 - test*表示任何以test开头的内容
 - *test以test结尾
 - *test* 包含test

查找命令

`which` 要查找的命令 命令其实是一个个可执行的程序，相当于exe。which只能查找命令的位置

`find` 起始路径 `-name "被查找的文件名"` 搜索指定文件。find也可以使用通配符

`find` 起始路径 `-size [+ -]n[单位]` 按文件大小查找（+ -表示大于小于）。n表示大小数字。单位有k、M、G

`locate` 文件：快速定位。它是把所有目录结构创建到数据库中进行查找，所以速度很快。并且因此再执行locate命令前，先要执行 `updatedb` 指令来创建locate数据库

grep、wc、管道符

`grep [-n -i] 关键字 文件路径` :从文件中通过关键字过滤文件行

- `-n`表示在结果中显示匹配的行的行号
- `-i`表示忽略大小写
- 如果关键字带有空格或其他符号，用""将关键字包围
- 文件路径也可以作为内容输入端口

`wc [-c -m -l -w] 文件路径` :

- `-c` 统计bytes数量
- `-m`统计字符数量
- `-l`统计行数
- `-w`统计单词数量
- 参数：被统计的文件，可以作为内容输入端口

管道符 |：将左边命令的结果作为右边命令的输入。于是可以将左边结果当作上面两个可以作为输入端口的参数。

比如 `ls -l /user/bin | grep gtf`, 输出 `-rwxr-xr-x. 1 root root 19592 2月 27 00:37 gtf`

- 管道符可以嵌套

echo、tail和重定向符

`echo` 输出的内容 :在命令行输出指定的内容。比如 `echo $HOSTNAME` 输出环境变量信息

在使用echo pwd时我们本意是要输出当前工作路径，但是pwd被当作文本输出

用``将pwd包围表示将pwd当作命令执行。

重定向符：

- > 表示将左侧命令的结果，覆盖写入符号右侧指定的文件中。如果文件不存在则创建
- >> 表示将左侧命令的结果，追加写入右侧指定文件

`tail [-f -num] 路径`

- 参数：被查看的文件路径

- -f：表示持续跟踪。在命令不停止的情况下，对文件进行更改，tail输出的内容会同步更新。通过 `ctrl+c` 退出追踪
- -num：查看尾部多少行，不填默认十行。填写数字

`head -n 数字 文件`：查看文件前n行

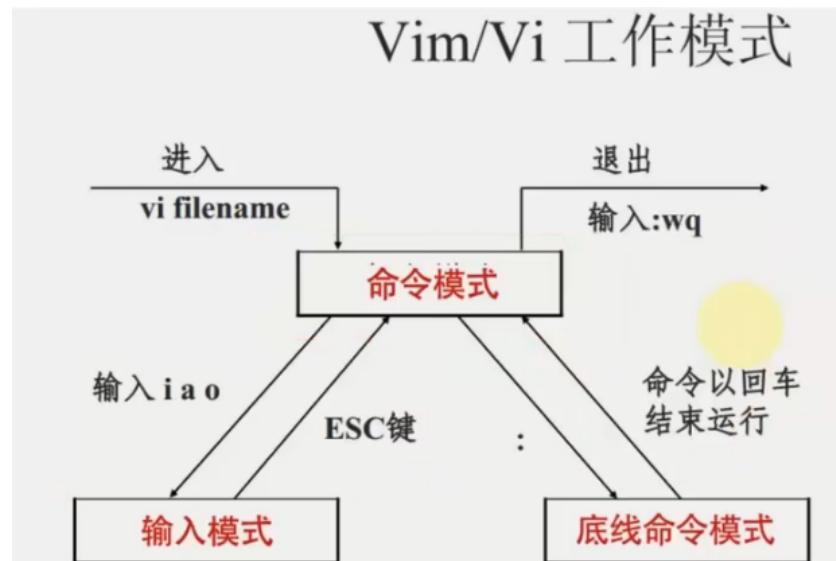
vi编辑器

visual interface 是linux中最经典的文本编辑器

vim是vi的加强版本，不仅能编辑文本，还能编辑shell程序、以不同颜色的字体来辨别语法的正确性

三种工作模式

- 命令模式：按键都被理解为命令。此模式下不能自由进行文本编辑
- 输入模式：就是编辑模式、插入模式。此模式下可以对文件内容进行自由编辑
- 底线命令模式：以`:`开头，通常用于文件的保存、退出



指正：上图的iao为i或a或o。并且用:wq保存退出。`:w`只保存不退出，`:q`只退出

使用

`vi` 文件路径

`vim` 文件路径

使用如上命令能够使用vi/vim编辑器编辑文本。后续全部使用vim

- 如果文件表示的路径不存在，此命令会用于编辑新文件

命令模式快捷键

命令模式	<code>pageup (PgUp)</code>	向上翻页
命令模式	<code>pangdown (PgDn)</code>	向下翻页
命令模式	/	进入搜索模式
命令模式	<code>n</code>	向下继续搜索
命令模式	<code>N</code>	向上继续搜索

命令模式	<code>dd</code>	删除光标所在行的内容
命令模式	<code>n dd</code>	<code>n</code> 是数字，表示删除当前光标向下 <code>n</code> 行
命令模式	<code>yy</code>	复制当前行
命令模式	<code>nyy</code>	<code>n</code> 是数字，复制当前行和下面的 <code>n</code> 行
命令模式	<code>p</code>	粘贴复制的内容
命令模式	<code>u</code>	撤销修改
命令模式	<code>ctrl + r</code>	反向撤销修改
命令模式	<code>gg</code>	跳到首行
命令模式	<code>G</code>	跳到行尾
命令模式	<code>dG</code> 	从当前行开始，向下全部删除
命令模式	<code>dgg</code>	从当前行开始，向上全部删除
命令模式	<code>d\$</code>	从当前光标开始，删除到本行的结尾
命令模式	<code>d0</code>	从当前光标开始，删除到本行的开头

注：G是跳到最后一行行首

数字+ `shift+g` 跳转到指定行

在底线命令模式下：

`:set nu` 显示行号

`:set nonu` 不显示行号

关机重启、注销

`shutdown -h now` : 立刻关机

`shutdown -h 1` : 一分钟后关机

`shutdown -r now` : 现在重启

`halt` : 立刻关机

`reboot` : 立刻重启

`sync`：把内存的数据同步到磁盘

注：虽然目前的shutdown/reboot/halt等命令均已经在关机前进行了sync，但是还是应该在重启或关闭系统之前，先运行sync

`logout`：注销用户；这个命令在图形运行级别无效，在运行级别3下有效（以后讲）

运行级别

● 基本介绍

运行级别说明：

0：关机

1：单用户【找回丢失密码】

2：多用户状态没有网络服务

3：多用户状态有网络服务

4：系统未使用保留给用户

5：图形界面

6：系统重启

常用运行级别是3和5，也可以指定默认运行级别，后面演示

● 应用实例

命令：`init [0123456]` 应用案例：通过init 来切换不同的运行级别，比如动 5-3，然后关机。

`systemctl get-default`：查看默认级别

`systemctl set-default TARGET.target`：设置默认级别。`multi-user.target` 为等级

`3.graphical.target` 为等级5。

用户和权限

root用户

在linux中，权限最大的账户为root（超级管理员）

`su [-] [用户名]`：

- `-` 是可选的，表示是否在切换用户后加载环境变量，建议带上
- 省略用户名表示切换到root
- `exit` 或 `ctrl+d` 退回上一个用户

`sudo` 其他命令

- 在其它命令之前，带上`sudo`，即可为这一条命令临时赋予root授权
- 经过`sudo`认证的普通用户才能使用`sudo`

配置`sudo`权限：

- 切换到root用户，执行`visudo`命令
- 在文件最后添加 `用户名 ALL=(ALL) NOPASSWD:ALL`
- 通过`:wq`保存

用户和用户组

Linux系统中可以：

- 配置多个用户
- 配置多个用户组
- 用户可以加入多个用户组

Linux中关于权限的管控级别有两个，分别是：

- 针对用户的权限控制
- 针对用户组的权限控制

比如，针对某文件，可以管控用户的权限，也可以控制用户组的权限

以下命令需root执行

- 创建用户组：`groupadd 组名`
- 删除：`groupdel 组名`
- `groupmod`：修改已有用户组的信息
- `useradd [-g -d] 用户名`
 - `-g`指定用户的组，不指定会创建用户的同名组并自动加入。
 - `-d`指定用户的HOME路径，不指定，默认在`/home/用户名`
- `userdel [-r] 名`：`-r`表示删除HOME目录。不加`-r`在删除用户时HOME目录保存
- `usermod -aG 用户组 用户名`：将指定用户加入指定用户组
- `usermod -d 目录名 用户名`：改变用户登录进入的初始目录
- `getent passwd`：查看当前系统中有哪些用户
- `getent group`：查看组
- `passwd 用户名`：修改密码。修改自己密码可以，修改别人密码需要root权限
- `passwd -l`：锁定账户

- `passwd -u`: 解锁账户
- `/etc/passwd`文件: 用户的配置文件, 记录用户的各种信息
- `/etc/shadow`文件: 口令的配置文件
- `/etc/group`文件: 组的配置文件, 记录Linux包含的组的信息
- 查看这些文件需要root权限

查看用户信息

- `id [用户名]`: 查看用户信息, 不写用户名查看自己
- `whoami`: 查看登陆时使用的用户, 使用了su命令后显示不变

查看权限控制信息

通过`ls -l`可以以列表形式查看内容, 并显示权限细节

```
[itheima@localhost ~]$ ls -l
总用量 0
drwxr-xr-x. 3 itheima itheima 37 9月 23 03:17 Desktop
drwxr-xr-x. 2 itheima itheima 6 9月 22 23:57 Documents
drwxr-xr-x. 2 itheima itheima 6 9月 22 23:57 Downloads
-rw-rw-r--. 1 itheima itheima 0 9月 26 00:16 hello.txt
drwxr-xr-x. 2 itheima itheima 6 9月 22 23:57 Music
drwxr-xr-x. 2 itheima itheima 6 9月 22 23:57 Pictures
drwxr-xr-x. 2 itheima itheima 6 9月 22 23:57 Public
drwxr-xr-x. 2 itheima itheima 6 9月 22 23:57 Templates
drwxr-xr-x. 2 itheima itheima 6 9月 22 23:57 Videos
```

- 序号1, 表示文件、文件夹的权限控制信息
- 序号2, 表示文件、文件夹所属用户
- 序号3, 表示文件、文件夹所属用户组

序号一的具体含义:

- 第一位: d表示文件夹, l表示软链接, -表示文件, c表示字符设备文件 (如鼠标键盘), b是块设备 (如硬盘)
- 2-4位: 表示本用户对此文件的权限 (r读、w写、x执行)
- 5-7位: 表示本用户组其他用户对本文件的权限
- 8-10位: 表示其他用户的权限

- rwx作用到文件
 - 1. [r]代表可读(read): 可以读取, 查看
 - 2. [w]代表可写(write): 可以修改, 但是不代表可以删除该文件. 删除一个文件的前提条件是对该文件所在的目录有写权限, 才能删除该文件.
 - 3. [x]代表可执行(execute): 可以被执行
- rwx作用到目录
 - 1. [r]代表可读(read): 可以读取, ls查看目录内容
 - 2. [w]代表可写(write): 可以修改, 对目录内创建+删除+重命名目录
 - 3. [x]代表可执行(execute): 可以进入该目录

所以, 如果要对目录内的文件进行操作, 必须要对该目录有一定的权限

修改权限控制

chmod

chmod命令修改文件、文件夹的权限信息

只有文件所属用户及root可以修改

chmod [-R] 权限 文件或文件夹： -R表示对文件夹内所有文件执行操作

示例: chmod u=rwx,g=rx,o=x hello.txt

简洁写法: chmod 751 文件名 。r为4, w为2, x为1

chmod o+w 文件/目录名： 给其他组加上w权限

chmod a-x 文件/目录名： 给所有去掉x权限

chown

可以修改文件、文件夹所属用户和用户组。 (只有root)

chown [-R] [用户] [:] [用户组] 文件/文件夹

- -R: 同chmod规则

示例:

- chown root hello.txt, 将hello.txt所属用户修改为root
- chown :root hello.txt, 将hello.txt所属用户组修改为root
- chown root:itheima hello.txt, 将hello.txt所属用户修改为root, 用户组修改为itheima
- chown -R root test, 将文件夹test的所属用户修改为root并对文件夹内全部内容应用同样规则

实用操作

小技巧

ctrl+c： 强制停止/输入错误直接终止输入跳到下一行

`ctrl+d` : 退出当前账户的登录/退出某些特定程序的专属页面 (不能用于vi/vim)

`history` : 查看输入的历史命令

`!命令前缀` : 执行上一次匹配前缀的命令

`! 数字` : 执行history展示出的指定行数的命令

`ctrl+r`: 输入内容去匹配历史命令

`ctrl+a` 跳到命令开头

`ctrl+e` 跳到命令结尾

`ctrl+l` 或 `clear` : 清空

安装

`yum`: RPM包软件管理器

`yum [-y] [install remove search] 软件名`

- `-y`: 自动确认

`yum`命令需要联网、root权限

软链接

软链接：将文件、文件夹链接到其他位置。 (类似于win中的快捷方式，打开的是本体)

`ln -s 参数一 参数二`

- 参数1: 原文件或目录
- 参数二: 软链接名

日期

`date [+格式化字符串]`

• 格式化字符串: 通过特定的字符串标记, 来控制显示的日期格式

- %Y 年
- %y 年份后两位数字 (00..99)
- %M 月份 (01..12)
- %d 日 (01..31)
- %H 小时 (00..23)
- %M 分钟 (00..59)
- %S 秒 (00..60)
- %s 自 1970-01-01 00:00:00 UTC 到现在的秒数

```
date "+%Y-%m-%d %H:%M:%S"
```

`date -s 字符串时间`: 将字符串时间变成默认格式输出

`cal [选项]`: 不加选项, 显示本月日历

环境变量

`$PATH` : 取出PATH环境变量的值

`echo $PATH` : 输出

- 临时设置: `export 变量名=变量值`
- 永久生效:
 - 针对当前用户: 配置在~/bashrc文件中
 - 针对所有用户: 配置在/etc/profile中,
 - 并通过 `source 配置文件` 立刻生效

下载和上传

`rz` : 从win上传文件

`sz 文件` : 下载文件到win

压缩解压

● tar 指令

tar 指令是打包指令，最后打包后的文件是 .tar.gz 的文件。

✓ 基本语法

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

选项说明

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

✓ 应用实例

案例1: 压缩多个文件，将 /home/pig.txt 和 /home/cat.txt 压缩成 pc.tar.gz

tar -zcvf pc.tar.gz /home/pig.txt /home/cat.txt

案例2: 将/home 的文件夹 压缩成 myhome.tar.gz

tar -zcvf myhome.tar.gz /home/

案例3: 将 pc.tar.gz 解压到当前目录

tar -zxvf pc.tar.gz

案例4: 将myhome.tar.gz 解压到 /opt/tmp2目录下 (1) mkdir /opt/tmp2 (2) tar -zxvf /home/myhome.tar.gz -C /opt/tmp2

`zip [-r] 目标.zip 要压缩的文件或目录 . . . :`

-r: 被压缩的包包含文件夹

`unzip [-d 目的地] .zip`

-d: 指定要解压去的位置。不写默认解压到当前文件夹

任务调度

任务调度：是指系统在某个时间执行特定的命令或程序

任务调度分类：

1. 系统工作：有些重要任务必须周而复始的执行，比如病毒扫描
2. 个别用户工作：个别用户可能希望定时执行某些程序，比如对mysql数据库的备份

crond

基本语法：

`crontab [选项]`

常用选项：

- -e: 编辑crontab定时任务
- -l: 查询crontab任务
- -r: 删除当前用户的所有crontab任务

`service crond restart`：重启任务调度

举例：

我们要每一分钟将/etc/目录下的信息写入/tmp/to.text

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

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

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

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

步骤:

- (1) vim /home/my.sh 写入内容 date >> /home/mycal 和 cal >> /home/mycal
- (2) 给 my.sh增加执行权限，chmod u+x /home/my.sh
- (2) crontab -e 增加 */1 * * * * /home/my.sh

at定时任务

```
at [选项] [时间]
```

Ctrl+D结束at输入

atq查看有没有未执行任务

atrm+编号：删除已经设置的任务

1. at命令是一次性定时计划任务，at的守护进程atd会以后台模式运行，检查作业队列来运行。
2. 默认情况下，atd守护进程每60秒检查作业队列，有作业时，会检查作业运行时间，如果时间与当前时间匹配，则运行此作业。
3. at命令是一次性定时计划任务，执行完一个任务后不再执行此任务了
4. 在使用at命令的时候，一定要保证atd进程的启动，可以使用相关指令来查看
`ps -ef | grep atd //可以检测atd是否在运行`

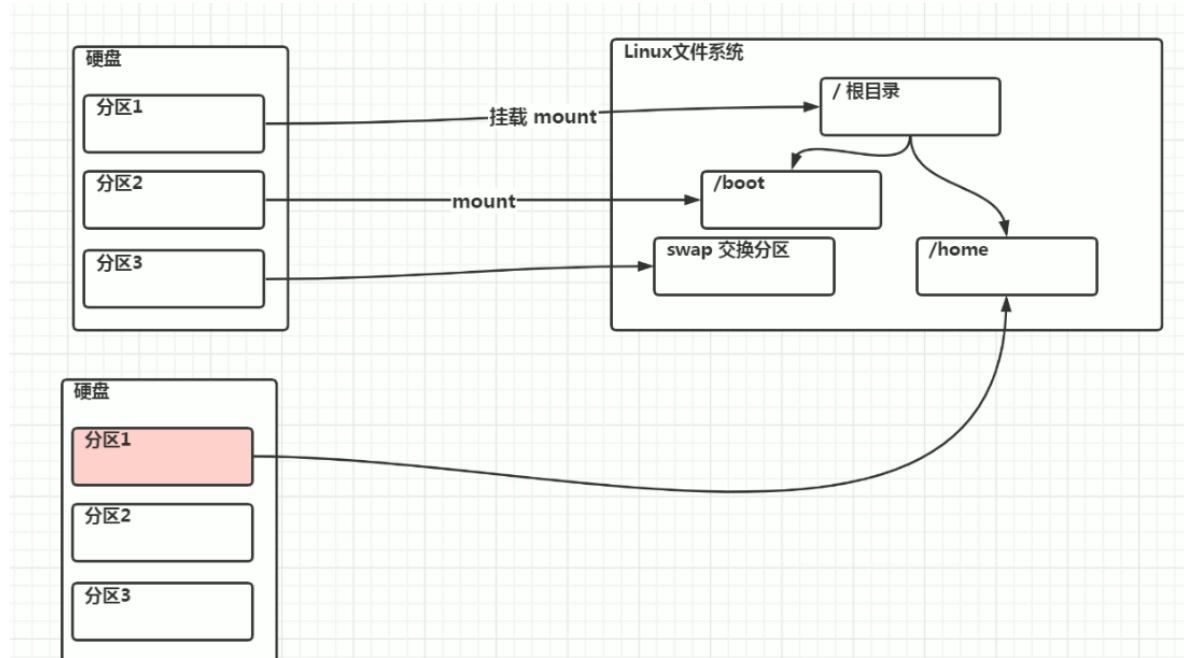
选项	含义
-m	当指定的任务被完成后，将给用户发送邮件，即使没有标准输出
-I	atq的别名
-d	atrm的别名
-v	显示任务将被执行的时间
-c	打印任务的内容到标准输出
-V	显示版本信息
-q <队列>	使用指定的队列
-f <文件>	从指定文件读入任务而不是从标准输入读入
-t <时间参数>	以时间参数的形式提交要运行的任务

at指定时间的方法：

1. 接受在当天的hh:mm (小时:分钟) 式的时间指定。假如该时间已过去，那么就放在第二天执行。例如：04:00
2. 使用midnight (深夜) , noon (中午) , teatime (饮茶时间，一般是下午4点) 等比较模糊的词语来指定时间。
3. 采用12小时计时制，即在时间后面加上AM (上午) 或PM (下午) 来说明是上午还是下午。例如：12pm
4. 指定命令执行的具体日期，指定格式为month day (月 日) 或mm/dd/yy (月/日/年) 或dd.mm.yy (日.月.年)，指定的日期必须跟在指定时间的后面。例如：04:00 2021-03-1
5. 使用相对计时法。指定格式为：now + count time-units，now就是当前时间，time-units是时间单位，这里能够是minutes (分钟) 、hours (小时) 、days (天) 、weeks (星期) 。count是时间的数量，几天，几小时。例如：now + 5 minutes
6. 直接使用today (今天) 、tomorrow (明天) 来指定完成命令的时间。

磁盘分区、挂载

磁盘分区机制



`lsblk` : 查看所有设备挂载情况

增加磁盘应用示例

- 说明：

下面我们以增加一块硬盘为例来熟悉下磁盘的相关指令和深入理解磁盘分区、挂载、卸载的概念。

- 如何增加一块硬盘

- 1. 虚拟机添加硬盘
- 2. 分区
- 3. 格式化
- 4. 挂载
- 5. 设置可以自动挂载

- 虚拟机增加硬盘步骤2

分区命令 `fdisk /dev/sdb`

开始对/sdb分区

`m` 显示命令列表

`p` 显示磁盘分区 同 `fdisk -l`

`n` 新增分区

`d` 删除分区

`w` 写入并退出

说明：开始分区后输入n，新增分区，然后选择p，分区类型为主分区。两次回车默认剩余全部空间。最后输入w写入分区并退出，若不保存退出输入q。

- 虚拟机增加硬盘步骤3

格式化磁盘

分区命令: `mkfs -t ext4 /dev/sdb1`

其中ext4是分区类型

- 虚拟机增加硬盘步骤4

挂载: 将一个分区与一个目录联系起来，

`mount 设备名称 挂载目录`

例如： `mount /dev/sdb1 /newdisk`

`umount 设备名称 或者 挂载目录`

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

注意: 用命令行挂载重启后会失效

- 虚拟机增加硬盘步骤5

● 永久挂载: 通过修改/etc/fstab实现挂载

添加完成后 执行`mount -a`即刻生效

磁盘情况查询

`df [-h]` :硬盘使用情况

-h :单位

`du -h / 目录`：查询指定目录的磁盘占用情况，默认为当前目录

-s：指定目录占用大小汇总

-a：含文件

--max-depth=1：子目录深度

-c：列出明细的同时，增加汇总值

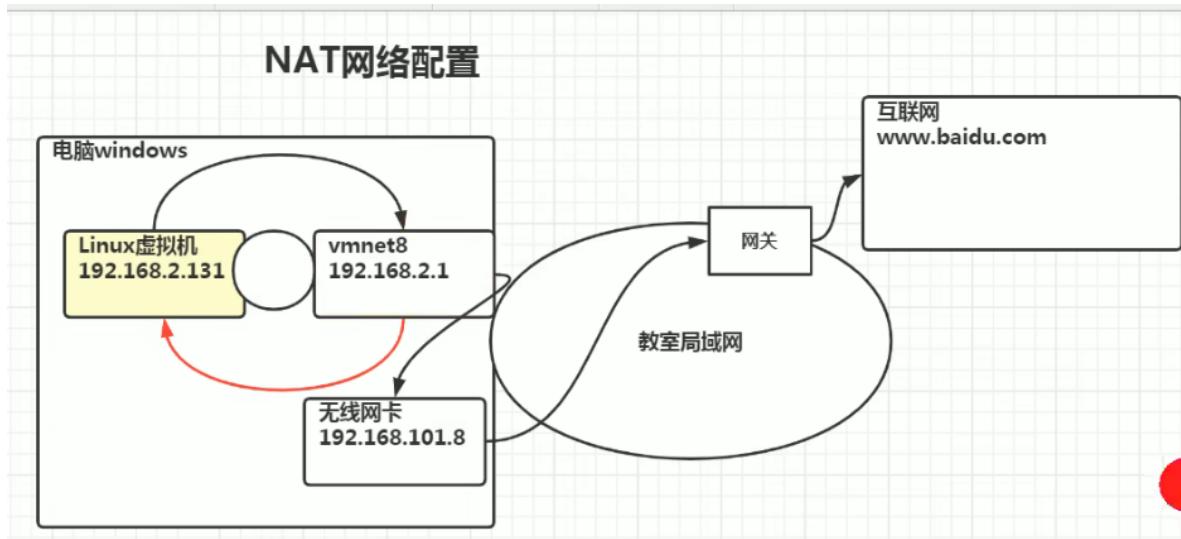
`iostat [-x] num1 num2`

- -x:显示更多信息
- num1: 刷新间隔
- num2: 刷新几次

网络配置

NAT网络配置原理

linux能访问百度的原理图：



网络配置指令

windows环境的vmnet8的配置：

cmd输入 `ipconfig`

查看linux网络配置:

```
ifconfig
```

网络请求和下载

`ping [-c num] ip或主机名`: 检查指定的网络服务器是否可联通

- -c :检查的次数, 不使用-c将无限次检查

`wget [-b] url` :在命令行内下载网络文件

- -b 后台下载

`curl [-O] url` : 发送http网络请求, 可用于下载文件、获取信息

- -O:用于下载文件, 当url是下载链接时, 可以使用此选项保存文件

网络环境配置

将linux的ip地址设置为静态的

具体查看教程 (需要了解网络相关知识)

主机名和hosts映射

主机名:

`hostnamectl set-hostname 名` :设置主机名

`hostname` : 查看主机名

DNS (Domain Name System)

DNS: 域名系统, 是互联网上作为域名和IP地址互相映射的一个分布式数据库

1. 浏览器先检查浏览器缓存中有没有该域名解析IP地址，有就先调用这个IP完成解析；如果没有，就检查DNS解析器缓存，如果有直接返回IP完成解析。这两个缓存，可以理解为本地解析器缓存
2. 一般来说，当电脑第一次成功访问某一网站后，在一定时间内，浏览器或操作系统会缓存他的IP地址（DNS解析记录）。如在cmd窗口中输入
`ipconfig /displaydns //DNS域名解析缓存`
`ipconfig /flushdns //手动清理dns缓存`
3. 如果本地解析器缓存没有找到对应映射，检查系统中hosts文件中有没有配置对应的域名IP映射，如果有，则完成解析并返回。
4. 如果本地DNS解析器缓存和hosts文件中均没有找到对应的IP，则到域名服务DNS进行解析域

即 浏览器缓存-->本地DNS缓存-->hosts文件-->域名服务DNS

如果都没有，返回域名不存在

hosts映射：

hosts：一个文本文件，用来记录IP和主机名（hostname）的映射关系

通过主机名找到某个linux系统

✓ windows
在 C:\Windows\System32\drivers\etc\hosts 文件指定即可
案例: 192.168.200.130 hspedu100

✓ linux
在 /etc/hosts 文件 指定
案例: 192.168.200.1 ThinkPad-PC

*进程管理

进程

- LINUX中，每个执行的程序都称为一个进程。每一个进程都分配一个ID号（pid，进程号）
- 每个进程都可能以两种方式存在：前台和后台。前台进程就是用户目前的屏幕上可以进行操作的；后台进程则是实际在操作，但屏幕上无法看到的进程，通常使用后台方式执行。
- 一般系统的服务都是以后台进程的方式存在，而且都会常驻在系统中，直到关机才结束。

`ps []`：查看进程信息

- -e :显示全部信息
- -f :以完全格式化的形式展示全部信息
- -a: 显示当前终端所有进程信息
- -u: 以用户的格式显示进程信息
- -x: 显示后台进程运行的参数

一般结合 |、more和grep使用

● ps详解

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

● 应用实例

要求 : 以全格式显示当前所有的进程 , 查看进程的父进程。查看 sshd 的父进程信息

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

`kill [-9] 进程ID`

- -9 :强制关闭
- 不会杀死子进程

`killall 进程名` : 通过进程名杀死进程

- 会杀死子进程
- 支持通配符
- 在系统因负载过大而变得很慢时很有用

案例1：踢掉某个非法登录用户

`kill 进程号`, 比如 `kill 11421`

案例2: 终止远程登录服务sshd, 在适当时候再次重启sshd服务

`kill sshd对应的进程号; /bin/systemctl start sshd.service`

案例3: 终止多个gedit, 演示 `killall gedit`

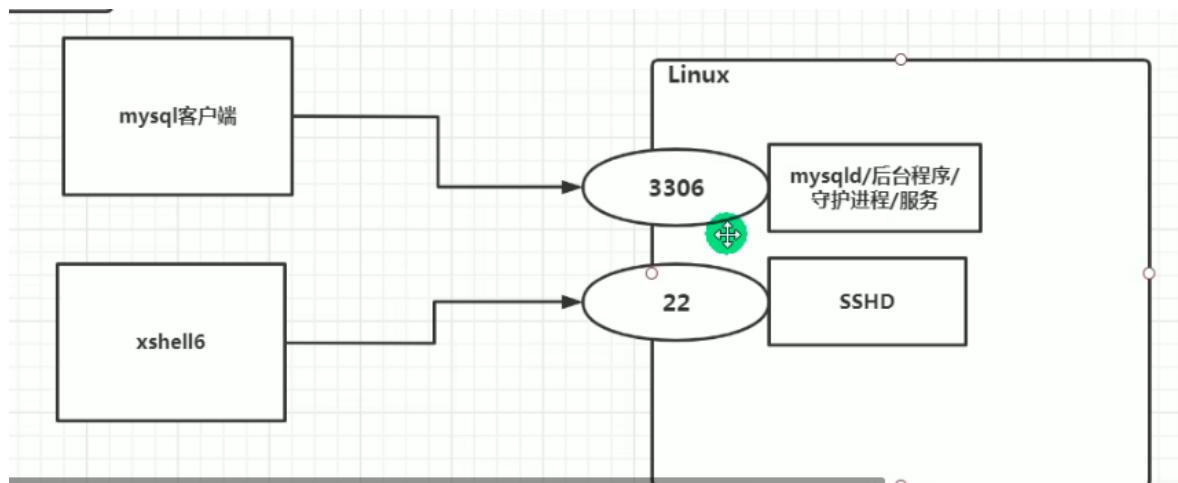
案例4 : 强制杀掉一个终端, 指令 `kill -9 bash 对应的进程号`

`pstree []`: 更直观的查看进程信息

- -p: 显示进程PID
- -u: 显示进程所属用户 (不是所有进程都有所属用户)

服务管理

服务 (service) 本质就是进程，但是是运行在后台的，通常都会监听某个端口，等待其他程序的请求。比如(mysql, sshd, 防火墙等)。因此我们又称之为守护进程。



service管理指令

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

Centos7后很多服务不再使用service，而是systemctl

`systemctl [start stop status restart] 服务名`

systemctl管理的服务在/usr/lib/systemd/system中查看

被service管理的服务在/etc/init.d查看

`setup`: 查看全部服务；前面带有`*`号的是随着linux启动而自动启动的，可以通过光标移动到`*`位置，再空格来取消`*`号

服务运行级别

✓ Linux系统有7种运行级别(runlevel)：常用的是级别3和5
运行级别0：系统停机状态，系统默认运行级别不能设为0，否则不能正常启动
运行级别1：单用户工作状态，root权限，用于系统维护，禁止远程登陆
运行级别2：多用户状态(没有NFS)，不支持网络
运行级别3：完全的多用户状态(有NFS)，登陆后进入控制台命令行模式
运行级别4：系统未使用，保留
运行级别5：X11控制台，登陆后进入图形GUI模式
运行级别6：系统正常关闭并重启，默认运行级别不能设为6，否则不能正常启动

linux开机流程：



运行级别的切换在上面的 [运行级别](#) 中讲过

systemctl设置服务自启动状态

因为3级别和5级别是常用的，所以以下指令在设置的时候默认是改变这两个级别的是否自启状态

```
systemctl list-unit-files [|grep 服务名]：查看服务是否开机自启
`systemctl enable 服务名：设置开机自启
`systemctl disable 服务名：关闭
`systemctl is-enabled 服务名：查看某个服务是否自启
```

firewall

生产环境中需要将防火墙打开，但是如果打开，外部请求数据包就不能跟服务器监听端口通讯。这时，需要打开指定的端口。比如80、22等。

测试连接：

```
telnet IP 端口
```

查看协议：

```
netstat -anp | more
```

```
firewall-cmd --permanent --add-port=端口号/协议：打开端口
```

```
firewall-cmd --permanent --remove-port=端口号/协议：关闭端口
```

```
firewall-cmd --reload：重新载入才能生效
```

```
firewall-cmd --query-port=端口号/协议：查询端口是否开放
```

2. 开放111端口

```
firewall-cmd --permanent --add-port=111/tcp；需要firewall-cmd --reload
```

3. 再次关闭111端口

```
firewall-cmd --permanent --add-port=111/tcp；需要firewall-cmd --reload
```

动态监控管理

top与ps命令很相似，都用来显示正在执行的进程。top与ps最大的不同之处在于top可以动态更新。

```
top []：
```

- -d 秒数：指定每几秒更新，默认3秒
- -i：使top不显示任何闲置或僵死进程
- -p：通过指定进程id来仅仅监控某个进程的状态。

显示出的每一列表示的信息：

- PR：优先级，数值越小，优先级越高。
- NI：进程的 nice 值。nice 值用于调整进程的优先级。正值表示低优先级，负值表示高优先级。
- VIRT：进程使用的虚拟内存总量。通常以 KB 或 MB 显示。
- RES：进程使用的常驻内存（Resident Memory）。
- SHR：进程使用的共享内存大小。表示进程与其他进程共享的内存部分，例如共享库。
- S：进程状态

使用top后输入：

- P：以CPU使用率排序，默认就是此项
- M：以内存使用率排序
- N：以PID从大到小排序
- q：退出top
- u：再输入用户名，回车，可以得到用户的进程
- k：再输入要结束的进程ID。如果要输入信号量，输入9强制删除

*监控网络状态

```
netstat []：
```

- -an：按照一定的顺序输出
- -p：显示哪个进程在调用

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	0	0.0.0.0:111	0.0.0.0:*	LISTEN
tcp	0	0	192.168.122.1:53	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:22	0.0.0.0:*	LISTEN
tcp	0	0	127.0.0.1:631	0.0.0.0:*	LISTEN
tcp	0	0	127.0.0.1:25	0.0.0.0:*	LISTEN
tcp	0	0	127.0.0.1:6011	0.0.0.0:*	LISTEN
tcp	0	36	192.168.200.130:22	192.168.200.1:14498	ESTABLISHED
tcp6	0	0	:::111	:::*	LISTEN
tcp6	0	0	:::22	:::*	LISTEN

proto表示网络协议

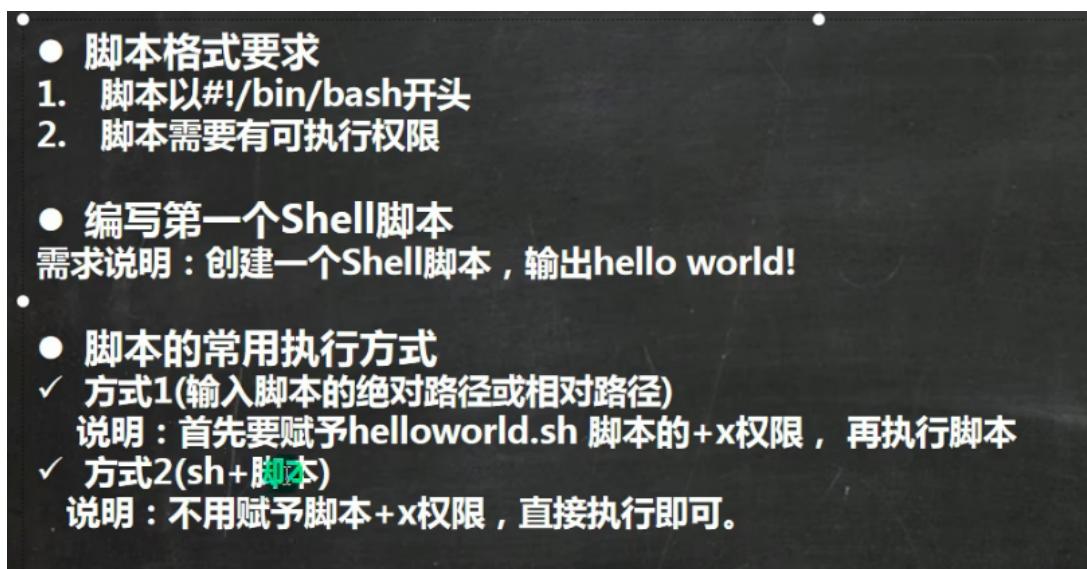
而local address 和 foreign address就比如说sshd和xshell的关系。其中前者IP地址为linux的IP地址；后者的IP地址为windows上vmnet8的IP地址。

state代表状态， LISTEN代表监听， ESTABLISHED表示已建立连接

Shell编程

快速入门

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



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

./程序 进行执行

Shell变量

分类

Linux Shell中的变量分为系统变量和用户自定义变量。

系统变量：\$HOME、\$PWD、\$SHELL、\$USER等等。

显示当前shell中所有变量: set

定义

shell变量的定义:

- 定义变量: 变量名=值
- 撤销变量: unset 变量
- 声明静态变量: readonly 变量, 注意: 不能unset

举例

```
#!/bin/bash
A=100
echo A=$A

unset A
echo A=$A

readonly B=100
echo B=$B
```

输出:

```
A=100
A=
B=100
```

定义变量的规则

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

✓ 将命令的返回值赋给变量
1. A=`date` 反引号，运行里面的命令，并把结果返回给变量A
2. A=\$(date) 等价于反引号
•

设置环境变量

export 变量名=变量值 : 设置环境变量

source 配置文件 : 让修改后的配置信息生效

echo \$变量名 : 查询环境变量的值

多行注释

```
:<<!    多行内容    !
```

位置参数变量

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

- 基本语法

- \$n (功能描述：n为数字，\$0代表命令本身，\$1-\$9代表第一到第九个参数，十以上的参数需要用大括号包含，如\${10})
- \$* (功能描述：这个变量代表命令行中所有的参数，\$*把所有的参数看成一个整体)
- \$@ (功能描述：这个变量也代表命令行中所有的参数，不过\$@把每个参数区分对待)
- \$# (功能描述：这个变量代表命令行中所有参数的个数)

其实就是传参

预定义变量

就是shell设计者事先定义好的变量，可以直接在shell中使用

\$\$：当前进程的进程号

\$\$!：后台运行的最后一个进程的进程号

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

运算符

- 基本语法

- “\$((运算式))” 或 “[运算式]” 或者 expr m + n //expression表达式
- 注意expr运算符间要有空格，如果希望将expr的结果赋给某个变量，使用``
- expr m - n
- expr *, /, % 乘，除，取余

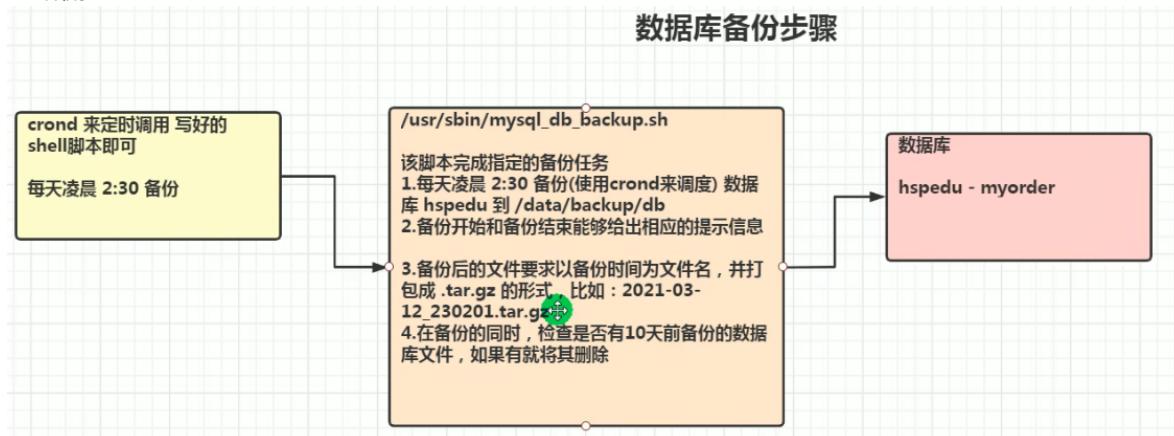
条件判断

定时备份数据库

● 需求分析

1. 每天凌晨 2:30 备份数据库 hspedu 到 /data/backup/db
2. 备份开始和备份结束能够给出相应的提示信息
3. 备份后的文件要求以备份时间为文件名，并打包成 .tar.gz 的形式，比如：2021-03-12_230201.tar.gz
4. 在备份的同时，检查是否有10天前备份的数据库文件，如果有就将其删除。

思路梳理：



shell脚本：

```
#!/bin/bash
#备份目录
BACKUP=/data/backup/db
#当前时间
DATETIME=$(date +%Y-%m-%d_%H%M%S)
echo $DATETIME

#数据库的地址
HOST=localhost
#数据库的用户名
DB_USER=root
#数据库密码
DB_PW=abc123
#备份的数据库名
DATABASE=muite

#创建备份目录,如果不存在的话
[ ! -d "${BACKUP}/${DATETIME}" ] && mkdir -p "${BACKUP}/${DATETIME}"

#备份数据库
mysqldump -u${DB_USER} -p${DB_PW} --host=${HOST} -q -R --databases ${DATABASE} | gzip > ${BACKUP}/${DATETIME}/$DATETIME.sql.gz

#将文件处理成tar.gz
cd ${BACKUP}
tar -zcvf ${DATETIME}.tar.gz ${DATETIME}
#删除对应的备份目录
rm -rf ${BACKUP}/${DATETIME}

# 删除10天前的备份文件
```

```
find ${BACKUP} -atime +10 -name "*.tar.gz" -exec rm -rf {} \;
echo "备份数据库${DATABASE}成功"
```

crond:

```
30 2 * * * /usr/sbin/mysql_db_backup.sh
```

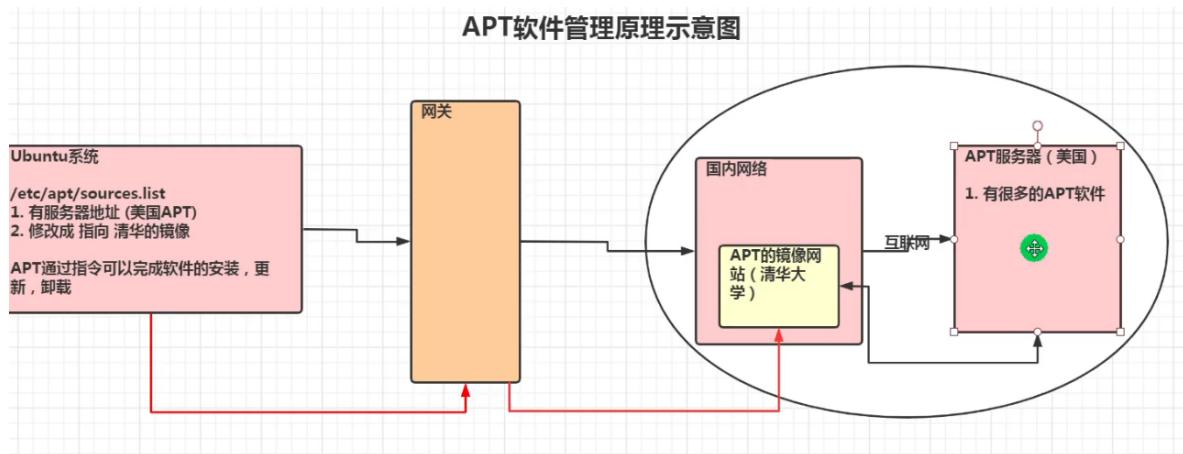
APT

APT原理

advanced packaging tools

安装包管理工具，在ubuntu下，我们可以使用apt命令进行软件包的安装、删除、清理等，类似于windows中的软件管理工具。

原理示意图：



相关命令

`sudo apt-get update`：更新源

`sudo apt-get install package`：下载

`sudo apt-get remove package`：删除

`sudo apt-cache show package`：查看软件相关信息

`sudo apt-get source package`：下载该包的源代码

备份文件：

```
sudo cp 文件 目标文件
```

远程登录

ssh：

ssh是Secure Shell 的缩写，是建立在应用层和传输层基础上的安全协议

常用于远程登陆。如果A机器想被B机器远程控制，那么A机器需要安装SSH服务器，B机器需要安装SSH客户端。

从一台Linux远程登录到另一台Linux：（在创建服务器集群时，会使用到该技术）

基本语法：`ssh 用户名@IP`

`exit或logout`：登出

*日志管理

/var/log是系统日志文件的保存位置

常用日志：

`/var/log/boot.log`：系统启动日志

`/var/log/cron`：记录与系统定时任务相关的信息

`/var/log/lastlog`：记录系统中所有用户最后一次登录时间。这个文件是二进制文件，要用lastlog查看

`/var/log/maillog`：记录邮件信息

`/var/log/message`：记录系统重要信息。如果系统出现问题，**首先要检查的就是这个日志**

`/var/log/secure`：记录验证和授权方面的信息，只要涉及**账户和密码**的程序都会记录。比如系统的登录、ssh的登陆、su切换用户、sudo授权，甚至添加用户和修改用户密码。

`/var/log/utmp`：记录当前已经登录的用户的信息。这个文件会随着用户的登录和注销不断变化，**只记录当前登录用户的信息**。这个文件**不能用vi查看**，而要用w、who、users等命令查看。

日志管理服务

有一个rsyslogd后台程序（服务）帮我们记录日志信息。

✓ 由日志服务 rsyslog 记录的日志文件，日志文件的格式包含以下 4 列：

1. 事件产生的时间
 2. 产生事件的服务器的主机名
 3. 产生事件的服务名或程序名
 4. 事件的具体信息
- ✓ 日志如何查看实例

查看一下 /var/log/secure 日志，这个日志中记录的是用户验证和授权方面的信息 来分析如何查看

日志服务配置文件

`ps aux |grep "rsyslog" |grep -v "grep"`：查询Linux中的 rsyslogd 服务有没有启动。其中 -v 表示查找不含"grep"的服务（用于过滤本条命令）

`systemctl list-unit-files | grep rsyslog`：查询 rsyslogd 服务的自启状态

可以在 /etc/rsyslog.d/50-default.conf 中查看日志记录默认规则

编辑文件时的格式为： *.* 存放日志文件

其中第一个*代表日志类型，第二个*代表日志级别

1. 日志类型分为：

auth	##pam产生的日志
authpriv	##ssh、ftp等登录信息的验证信息
corn	##时间任务相关
kern	##内核
lpr	##打印
mail	##邮件
mark(syslog)-rsyslog	##服务内部的信息，时间标识
news	##新闻组
user	##用户程序产生的相关信息
uucp	##unix to unix copy主机之间相关的通信
local 1-7	##自定义的日志设备
•	•

2. 日志级别分为：

debug	##有调试信息的，日志通信最多
info	##一般信息日志，最常用
notice	##最具有重要性的普通条件的信息
warning	##警告级别
err	##错误级别，阻止某个功能或者模块不能正常工作的信息
crit	##严重级别，阻止整个系统或者整个软件不能正常工作的信息
alert	##需要立刻修改的信息
emerg	##内核崩溃等重要信息
none	##什么都不记录

注意：从上到下，级别从低到高，记录信息越来越少

自定义日志服务

可以在 /etc/log/rsyslog.d/ 目录下创建新的.conf文件自定义日志服务。然后使用 `sudo systemctl restart rsyslog` 重新加载日志服务以刷新。

原理是在 /etc/rsyslog.conf 文件下使用了 `$IncludeConfig /etc/rsyslog.d/*.conf` 导入了日志记录规则

日志轮替

● 基本介绍

日志轮替就是把旧的日志文件移动并改名，同时建立新的空日志文件，当旧日志文件超出保存的范围之后，就会进行删除

● 日志轮替文件命名

1. centos7使用logrotate进行日志轮替管理，要想改变日志轮替文件名字，通过 /etc/logrotate.conf 配置文件中 “dateext” 参数：
2. 如果配置文件中有 “dateext” 参数，那么日志会用日期来作为日志文件的后缀，例如 “secure-20201010”。这样日志文件名不会重叠，也就不需要日志文件的改名，只需要指定保存日志个数，删除多余的日志文件即可。
3. 如果配置文件中没有 “dateext” 参数，日志文件就需要进行改名了。当第一次进行日志轮替时，**当前的 “secure” 日志会自动改名为 “secure.1”，然后新建 “secure” 日志，用来保存新的日志。**当第二次进行日志轮替时，“secure.1” 会自动改名为 “secure.2”，当前的 “secure” 日志会自动改名为 “secure.1”，然后也会新建 “secure” 日志，用来保存新的日志，以此类推。

● logrotate 配置文件

✓ /etc/logrotate.conf 为 logrotate 的全局配置文件

```
# rotate log files weekly. 每周对日志文件进行一次轮替
weekly
# keep 4 weeks worth of backlogs. 共保存4份日志文件，当建立新的日志文件时，旧的将会被删除
rotate 4
# create new (empty) log files after rotating old ones. 创建新的空的日志文件，在日志轮替后
create
# use date as a suffix of the rotated file. 使用日期作为日志轮替文件的后缀
dateext
# uncomment this if you want your log files compressed. 日志文件是否压缩。如果取消注释，则日志会在转储的同时进行压缩
#compress
#RPM packages drop log rotation information into this directory
include /etc/logrotate.d
# 包含 /etc/logrotate.d/ 目录中所有的子配置文件。也就是说会把这个目录中所有子配置文件读取进来，
#下面是单独设置，优先级更高。
# no packages own wtmp and btmp -- we'll rotate them here
/var/log/wtmp {
    monthly # 每月对日志文件进行一次轮替
    create 0664 root utmp # 建立的新日志文件，权限是 0664，所有者是 root，所属组是 utmp 组
    minsize 1M # 日志文件最小轮替大小是 1MB。也就是日志一定要超过 1MB 才会轮替，否则就算时间达到一个月，也不进行日志转储
    rotate 1 # 仅保留一个日志备份。也就是只有 wtmp 和 wtmp.1 日志保留而已
}
/var/log/btmp {
    missingok # 如果日志不存在，则忽略该日志的警告信息
    monthly
    create 0600 root utmp
    rotate 1
}
```

● logrotate 配置文件	
✓ 参数说明	
参数	参数说明
daily	日志的轮替周期是每天
weekly	日志的轮替周期是每周
monthly	日志的轮替周期是每月
rotate 数字	保留的日志文件的个数。0 指没有备份
compress	日志轮替时，旧的日志进行压缩
create mode owner group	建立新日志，同时指定新日志的权限与所有者和所属组。
• mail address	当日志轮替时，输出内容通过邮件发送到指定的邮件地址。
missingok	如果日志不存在，则忽略该日志的警告信息
notifempty	如果日志为空文件，则不进行日志轮替
minsize 大小	日志轮替的最小值。也就是日志一定要达到这个最小值才会轮替，否则就算时间达到也不轮替
size 大小	日志只有大于指定大小才进行日志轮替，而不是按照时间轮替。
dateext	使用日期作为日志轮替文件的后缀。
sharedscripts	在此关键字之后的脚本只执行一次。
prerotate/endscript	在日志轮替之前执行脚本命令。
postrotate/endscript	在日志轮替之后执行脚本命令。

日志轮替机制：

● 日志轮替机制原理

日志轮替之所以可以在指定的时间备份日志，是依赖系统定时任务。在 /etc/cron.daily/ 目录，就会发现这个目录中是有 logrotate 文件(可执行)，logrotate 通过这个文件依赖定时任务执行的。

```
[root@hspedu100 logrotate.d]# ls -l /etc/cron.daily/
总用量 12
-rwx----- 1 root root 219 10月 31 2018 logrotate
-rwxr-xr-x 1 root root 618 10月 30 2018 man-db.cron
-rwx----- 1 root root 208 4月 11 2018 mlocate
[root@hspedu100 logrotate.d]#
```

内存日志

有些内存在系统运行时是写在内存中，没有写到文件中，比如内核相关的日志。重启会清空

● journalctl 可以查看内存日志，这里我们看看常用的指令

```
journalctl ##查看全部
journalctl -n 3 ##查看最新3条
journalctl --since 19:00 --until 19:10:10 #查看起始时间到结束时间的日志可加日期
journalctl -p err ##报错日志
journalctl -o verbose ##日志详细内容
journalctl _PID=1245 _COMM=sshd ##查看包含这些参数的日志（在详细日志查看）
•或者 journalctl | grep sshd
```

定制自己的Linux

Linux启动流程：

- 1、首先Linux要通过自检，检查硬件设备有没有故障
- 2、如果有多块启动盘的话，需要在BIOS中选择启动磁盘
- 3、启动MBR中的bootloader引导程序
- 4、加载内核文件
- 5、执行所有进程的父进程、老祖宗systemd
- 6、欢迎界面

在Linux的启动流程中，加载内核文件时关键文件：

- 1) kernel文件: vmlinuz-3.10.0-957.el7.x86_64
- 2) initrd文件: initramfs-3.10.0-957.el7.x86_64.img

原理见：

【【小白入门 通俗易懂】韩顺平一周学会Linux】https://www.bilibili.com/video/BV1Sv411r7vd?p=126&vd_source=c054be8430afebb3d00e5f2d0b77f9fc

后面画了一张原理图

实操见：

【【小白入门 通俗易懂】韩顺平一周学会Linux】https://www.bilibili.com/video/BV1Sv411r7vd?p=127&vd_source=c054be8430afebb3d00e5f2d0b77f9fc

升级内核

`apt list --upgradable | grep linux-image`：查看可升级的版本，无输出则说明没有可用升级版本

`sudo apt install linux-image-6.8.0-57-generic`：升级指定版本

`dpkg --list | grep linux-image`：查看所有已安装的内核

备份与恢复

方式：

- 把需要的文件（或分区）用TAR打包，下次需要恢复的时候，再解压覆盖
- 使用dump和restore命令

参考：https://blog.csdn.net/Zheng_Huang/article/details/108325467

备份

示例操作

1. 创建一个完整备份

假设你想对 `/home` 目录进行完整备份，并将备份文件保存为 `/backup/home.dump`：

```
sudo dump -0uf /backup/home.dump /home
```

- `-0u`：指定备份级别为 0（完全备份），并且更新 `/etc/dumpdates` 文件。
- `-f`：指定输出文件或设备。

2. 创建一个增量备份

如果你已经创建了一个完整的备份，现在想创建一个增量备份（例如级别 1），可以运行：

```
sudo dump -1uf /backup/home_level1.dump /home
```

这会备份自上次级别 0 备份以来更改过的所有文件。

3. 使用压缩选项

为了节省空间，你可以在备份时使用压缩：

```
sudo dump -0uj /backup/home.dump.gz /home
```

这里的 `-j` 选项告诉 `dump` 使用 bzip2 压缩备份数据。

4. 注意

只有分区支持增量备份，目录或文件不支持

上面对 home 进行增量备份也只有当 home 是分区时才能够成功执行。

5. 查看

`dump -w`：显示备份的文件及其最后一次备份的层级、时间、日期

`cat /etc/dumpdates`：查看各次备份的时间

恢复

● 基本介绍

`restore` 命令用来恢复已备份的文件，可以从 `dump` 生成的备份文件中恢复原文件

● `restore` 基本语法

`restore [模式选项] [选项]`

说明下面四个模式，不能混用，在一次命令中，只能指定一种。

`-C`：使用对比模式，将备份的文件与已存在的文件相互对比。

`-i`：使用交互模式，在进行还原操作时，`restors` 指令将依序询问用户

`-r`：进行还原模式

`-t`：查看模式，看备份文件有哪些文件

选项

`-f <备份设备>`：从指定的文件中读取备份数据，进行还原操作

在使用 `restore` 恢复数据之前，运行 `restore -cf` 可以帮助你确认备份文件是否完好无损，避免在恢复过程中遇到问题。

举例：

```
restore -C -f /root/boot.dump.1
```