**Crontab简介：**

CRONTAB，相当于Linux、Unix下的计划任务，用来定时重复的执行某项任务。Crontab是一个用于设置 ***周期性*** 被执行的任务的工具。

被周期性执行的任务我们称为CronJob。

周期性执行的任务列表我们称为Crontable。

**Putty工具：**

免费的SSH连接工具

[http://www.puttu.org/](http://www.putty.org/)

**安装并检查 Crontab 服务：**

安装：

- yum install vixie-cron

- yum install crontabs

检查：

- crontab -l #检查Crontab工具是否安装

- service crond status #检查crond服务是否启动

- service crond #查询crond服务可以使用的命令



案例演习1：

每分钟都打印当前时间到一个日志文件中

\*\*\*\*\*\*\*\*\*date >> /tmp/log.txt

- crontab -e #打开计划任务

#写入【\*/1 \* \* \* \* date >> /tmp/log.txt】,保存退出

- crontab -l #打印计划任务

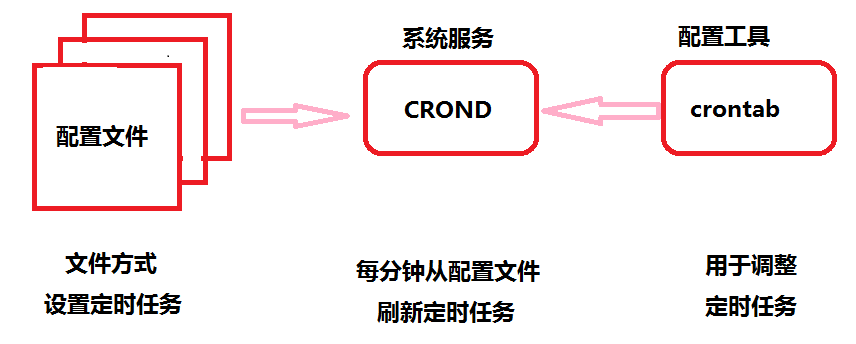
- tail -f /tmp/log.txt #动态刷新log.txt文件的最后一行

1. **Crontab基本组成：**

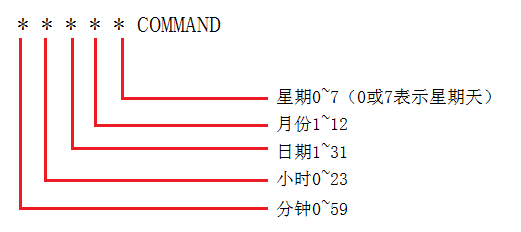
CROND 系统服务：每分钟都会从配置文件刷新定时任务，并执行计划任务

CROND以文件的形式设定定时任务

Crontab配置工具：用于调整定时任务



Crontab的配置文件格式



例如：

* 每天10:30重启apache服务

30 10 \* \* \* service httpd restart

* 每月1、10、22日4:45重启apache

45 4 1,10,22 \* \* service httpd restart

* 每月1到10日4:45重启apache

45 4 1-10 \* \* service httpd restart

* 每隔两分钟重启apache服务

\*/2 \* \* \* \* service httpd restart

* 每奇数分钟重启apache服务

1-59/2 \* \* \* \* service httpd restart

* 晚上11点到早上7点之间，每隔一小时重启apache

0 23-7/1 \* \* \* service httpd restart

* 每天18点到23点之间每隔30分钟重启apache

0-59/30 18-23 \* \* \* service httpd restart

0,30 18-23 \* \* \* service httpd restart

小结：

- \*表示任何时候都能匹配

- 用逗号隔开表示，表示在这几个点处执行命令

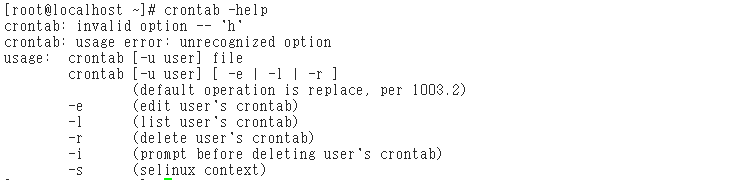
- -表示两点之间时执行命令

- \*/x 表示每x间隔执行一次命令

* 使用Crontab命令时，最好使用绝对路径

**二、Crontab工具的使用：**

- crontab -help



-u：指定计划任务分配的用户，默认是当前登录用户

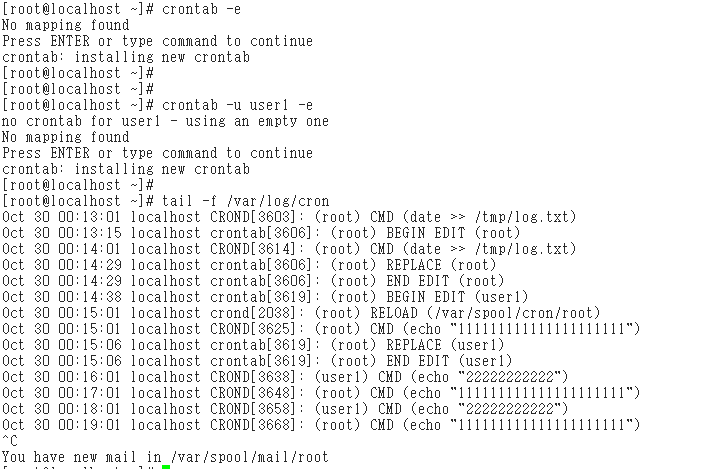
-e：编辑计划任务

-l：列出用户的计划任务

-r：删除用户的计划任务

案例演习2：

多用户同时设定定时任务



【root】Crontab -e：

1-59/2 \* \* \* \* echo “11111111111111111111”

【root】Crontab -u user1 -e：

0-58/2 \* \* \* \* echo “22222222222222222”

**三、Crontab 配置文件：**

**访问控制**

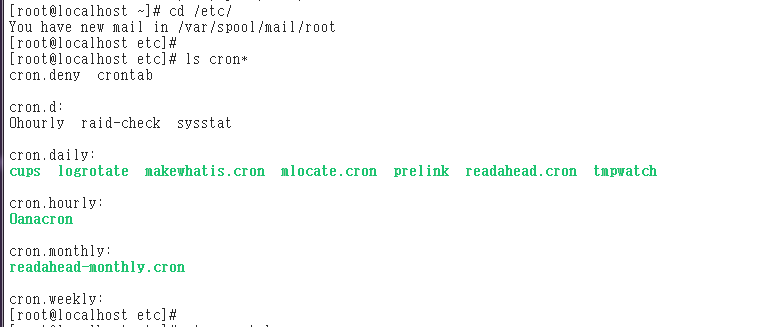
和一次性定时任务类似，crontab也有访问控制文件。

当系统中有 /etc/cron.allow文件时，只有写入此文件的用户可以使用crontab 命令，没有写入的用户不能使用crontab 命令。当系统中只有文件 /etc/cron.deny 文件时，写入该文件中的用户不可以使用crontab命令，root用户不受此限制。

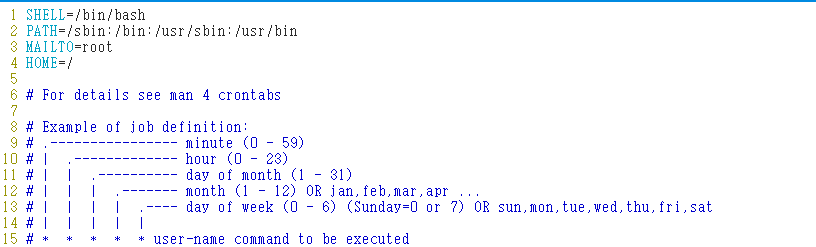
Cron.allow的优先级高于cron.deny，如果两个文件都存在，则cron.deny将被忽略

全局配置文件

- /etc/crontab



-1>打开文件 crontab



在该全局的配置文件中也可以添加新的系统级别的计划任务。（需要指定用户）

- \*/1 \* \* \* \* root echo “root->cron/crontab”

MAILTO表示的是计划任务改变之后生成日志的发送对象。

-2>进入配置文件cron.d

- cd cron.d

- ls

- cat sysstat

Crontab 在载入配置文件的过程中，会把sysstat文件中的配置信息载入成root用户的计划任务。

Crontab -e 修改内容，不会改变sysstat配置文件中的信息。

直接修改sysstat配置文件中的信息，不会直接载入到crontab中，可以使用命令

crontab sysstat的方式将配置文件中的内容导入crontab中。这会覆盖原来crontab -e中的全部内容。

-3>

/etc/cron.d/sysstat 文件中有注明的用户名，在目录 /etc/cron.d 下的所有文件，都可以认为是系统配置文件的补充。

也可以在该目录下自己创建文件，写相应的计划任务，同样需要注明用户。不需要载入，就可以执行该计划，而且该计划并没有放到 crontab -l -u user中。因此说明，crontab工具不可以编辑系统级别的计划任务。

-4>补充

Crontab -e写入的定时任务，以及crontab -l列出的定时任务，保存在文件 /var/spool/mail/”user”文件中.不同用户对应不同的子文件夹

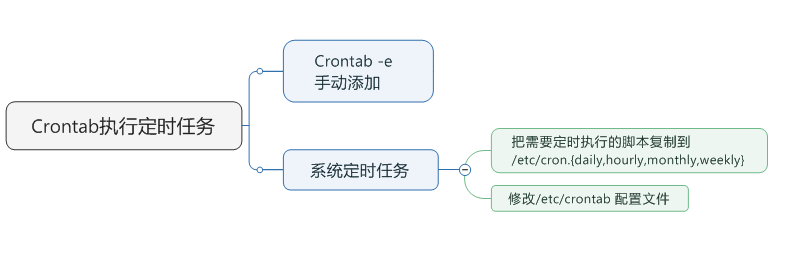
cat /var/spool/mail/”user”

**四、Crontab 日志**

/var/log/cron 文件保存着Crontab的任务执行记录

/var/spool/mail邮件发送地址

cd /var/log/cron # 任务执行记录



1. **Crontab的常见错误**
2. 环境变量

cd ~

ls -a

vim .bash\_profile

#添加新的变量

<

APP=\etc

export APP

>

source \_bash\_profile

echo $变量

**这里可以直接得到刚刚我们添加好的变量值，接下来，当我们把输出该变量的命令放到crontab中时，可以通过命令 tail -f /var/log/cron 查看到该计划命令确实执行，但echo该变量重定向到的文件中确是空行。表明，虽然我们预置了新的变量，但该变量没有载入到环境变量中。**

.bash\_profile 文件是隐藏文件，里面保存的是用户的环境变量

|  |
| --- |
| **source命令用法：** source FileName 作用:在当前bash环境下读取并执行FileName中的命令。 注：该命令通常用命令“.”来替代。 如：source .bash\_rc 与 . .bash\_rc 是等效的。 注意：source命令与shell scripts的区别是， source在当前bash环境下执行命令，而scripts是启动一个子shell来执行命令。这样如果把设置环境变量（或alias等等）的命令写进scripts中，就只会影响子shell,无法改变当前的BASH,所以通过文件（命令列）设置环境变量时，要用source 命令。 |

2、命令行双引号中使用 % 时，未加反斜线\

3、第三个和第五个域之间执行的是 “或”操作，其他的域之间时 “与”操作。

计划任务：四月的第一个星期天早晨1时59分运行a.sh

错误命令：59 1 1-7 4 0 /root/a.sh

执行结果：四月的1-7号和四月的每个周天的1:59都执行该命令

修 正：59 1 1-7 4 \* test `date + \%w` -eq 0 && /root/a.sh

1. 分钟设置误用

两个小时运行一次

补充：

Crontab**最小**只能设置到**每分钟**执行一次命令，如果想半分钟执行某个命令怎么做的？

计划任务：每半分钟执行一次 将date打印到/tmp/date.log

执行命令：

\*/1 \* \* \* \* date >> /tmp/date.log

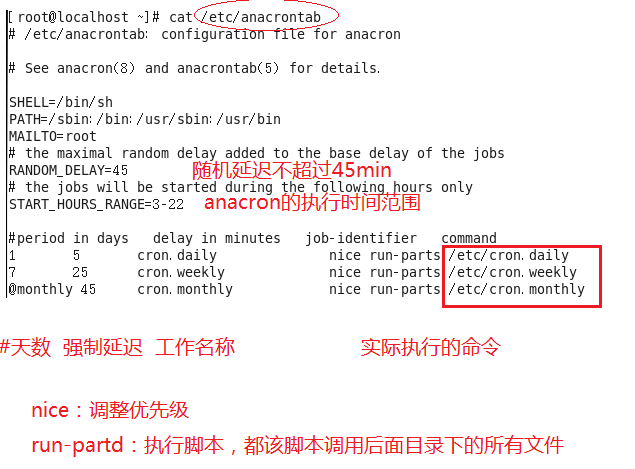
\*/1 \* \* \* \* sleep 30s;date >> /tmp/date.log

1. **anacron配置**

当服务器在该执行定时任务的时候，因为关机或者故障错过了某些定时任务，在服务器重新开机或这修复好之后，anacron会自动检测这段时间有无误漏掉个别定时任务。如果发生个别定时任务没有执行，它会再次执行。

它只会执行*/etc/cron.{}*和*/etc/crontab*配置文件中的定时任务。

* Anacron会使用一天，七天，一个月作为检测周期
* 在系统的 */var/spool/anacron/* 目录中存在*cron.{daily,monthly,weekly*}文件，用于记录上次执行cron的时间
* 和当前时间做比较，若两个时间的插值超过了anacron的指定时间差值，证明有cron任务被错过。



**Cron.daily工作来说明anacron执行过程**

1、首先读取 /var/spool/anacron/cron.daily 中的上一次anacron执行的时间

2、和当前时间比较，如果两个时间差超过1天，执行cron.daily工作

3、执行这个工作只能在3:00~22:00之间

4、执行工作时强制延迟时间为5分钟，再随机延迟0~45分钟时间

5、使用nice命令指定默认优先级使用run-parts脚本执行/etc/cron.daily 目录中的所有可执行文件