crontab命令

# crontab命令引入

前一天学习了 at 命令是针对仅运行一次的任务，**循环运行的例行性计划任务，linux系统则是由 cron (crond) 这个系统服务来控制的**。Linux 系统上面原本就有非常多的计划性工作，因此这个系统服务是默认启动的。另外, 由于使用者自己也可以设置计划任务，所以， **Linux 系统也提供了使用者控制计划任务的命令 :crontab 命令**。

# crontab命令简介

crond是linux下用来周期性的执行某种任务或等待处理某些事件的一个守护进程，与windows下的计划任务类似，当安装完成操作系统后，默认会安装此服务工具，并且会自动启动crond进程，crond进程每分钟会定期检查是否有要执行的任务，如果有要执行的任务，则自动执行该任务。

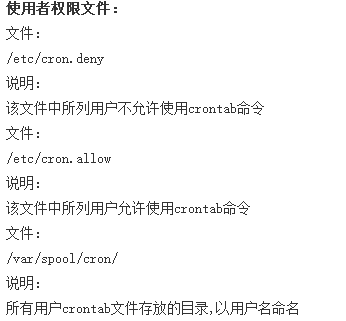
Linux下的任务调度分为两类，**系统任务调度和用户任务调度**。

**系统任务调度**：系统周期性所要执行的工作，比如写缓存数据到硬盘、日志清理等。在**/etc目录下有一个crontab文件**，这个就是**系统任务调度的配置文件**。



前四行是用来配置crond任务运行的环境变量，第一行SHELL变量指定了系统要使用哪个shell，这里是bash，第二行PATH变量指定了系统执行命令的路径，第三行MAILTO变量指定了crond的任务执行信息将通过电子邮件发送给root用户，如果MAILTO变量的值为空，则表示不发送任务执行信息给用户，第四行的HOME变量指定了在执行命令或者脚本时使用的主目录。第六至九行表示的含义将在下个小节详细讲述。这里不在多说。

**用户任务调度**：用户定期要执行的工作，比如用户数据备份、定时邮件提醒等。用户可以使用 crontab 工具来定制自己的计划任务。所有用户定义的crontab 文件都被保存在 **/var/spool/cron目录**中。其文件名与用户名一致。



# crontab文件的含义：

用户所建立的**crontab文件**中，每一行都代表一项任务，每行的每个字段代表一项设置，它的格式共分为**六个字段**，**前五段是时间设定段**，**第六段是要执行的命令段**，格式如下：

**minute hour day month week** ***command***

其中：

minute： 表示分钟，可以是从0到59之间的任何整数。

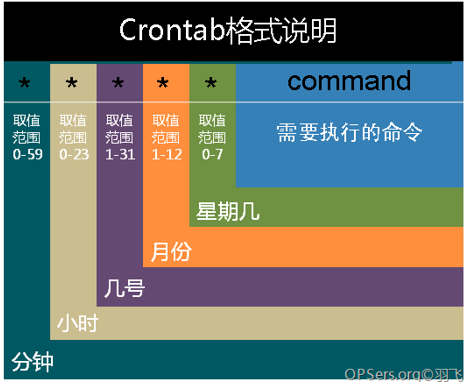
hour：表示小时，可以是从0到23之间的任何整数。

day：表示日期，可以是从1到31之间的任何整数。

month：表示月份，可以是从1到12之间的任何整数。

**week：表示星期几，可以是从0到7之间的任何整数，这里的0或7代表星期日**。

**command：要执行的命令，可以是系统命令，也可以是自己编写的脚本文件**



**在以上各个字段中，还可以使用以下特殊字符：**

**星号（\*）**：代表所有可能的值，例如month字段如果是星号，则表示在满足其它字段的制约条件后每月都执行该命令操作。

**逗号（,）**：可以用逗号隔开的值指定一个列表范围，例如，“1,2,5,7,8,9”

**中杠（-）**：可以用整数之间的中杠表示一个整数范围，例如“2-6”表示“2,3,4,5,6”

**正斜线（/）**：可以用正斜线指定时间的间隔频率，例如“0-23/2”表示每两小时执行一次。同时正斜线可以和星号一起使用，例如\*/10，如果用在minute字段，表示每十分钟执行一次。

# crond服务

安装crontab：

yum install crontabs

服务操作说明：

/sbin/service crond start //启动服务

/sbin/service crond stop //关闭服务

/sbin/service crond restart //重启服务

/sbin/service crond reload //重新载入配置

查看crontab服务状态：

service crond status

手动启动crontab服务：

service crond start

查看crontab服务是否已设置为开机启动，执行命令：

ntsysv

加入开机自动启动：

chkconfig –level 35 crond on

# crontab命令详解

1.命令格式：

crontab [-u user] file

crontab [-u user] [ -e | -l | -r ]

**2．命令功能：**

通过crontab 命令，我们可以在固定的间隔时间执行指定的系统指令或 shell script脚本。时间间隔的单位可以是分钟、小时、日、月、周及以上的任意组合。这个命令非常设合周期性的日志分析或数据备份等工作。

**3．命令参数：**

-u user：用来设定某个用户的crontab服务，例如，“-u ixdba”表示设定ixdba用户的crontab服务，此参数一般有root用户来运行。

file：file是命令文件的名字,表示将file做为crontab的任务列表文件并载入crontab。如果在命令行中没有指定这个文件，crontab命令将接受标准输入（键盘）上键入的命令，并将它们载入crontab。

-e：编辑某个用户的crontab文件内容。如果不指定用户，则表示编辑当前用户的crontab文件。

-l：显示某个用户的crontab文件内容，如果不指定用户，则表示显示当前用户的crontab文件内容。

-r：从/var/spool/cron目录中删除某个用户的crontab文件，如果不指定用户，则默认删除当前用户的crontab文件。

-i：在删除用户的crontab文件时给确认提示。

**4．常用方法：**

**1). 创建一个新的crontab文件**

在考虑向cron进程提交一个crontab文件之前，首先要做的一件事情就是设置环境变量EDITOR。cron进程根据它来确定使用哪个编辑器编辑crontab文件。9 9 %的UNIX和LINUX用户都使用vi，如果你也是这样，那么你就编辑$ HOME目录下的. profile文件，在其中加入这样一行：

EDITOR=vi; export EDITOR

然后保存并退出。不妨创建一个名为<user> cron的文件，其中<user>是用户名，例如， davecron。在该文件中加入如下的内容。

      # (put your own initials here)echo the date to the console every

      # 15minutes between 6pm and 6am

      0,15,30,45 18-06 \* \* \* /bin/echo 'date' > /dev/console

    保存并退出。确信前面5个域用空格分隔。

在上面的例子中，系统将每隔1 5分钟向控制台输出一次当前时间。如果系统崩溃或挂起，从最后所显示的时间就可以一眼看出系统是什么时间停止工作的。在有些系统中，用tty1来表示控制台，可以根据实际情况对上面的例子进行相应的修改。为了提交你刚刚创建的crontab文件，可以把这个新创建的文件作为cron命令的参数：

     $ crontab davecron

现在该文件已经提交给cron进程，它将每隔1 5分钟运行一次。

同时，新创建文件的一个副本已经被放在/var/spool/cron目录中，文件名就是用户名(即dave)。

**2). 列出crontab文件**

   为了列出crontab文件，可以用：

     $ crontab -l

     0,15,30,45,18-06 \* \* \* /bin/echo `date` > dev/tty1

你将会看到和上面类似的内容。可以使用这种方法在$ H O M E目录中对crontab文件做一备份：

     $ crontab -l > $HOME/mycron

    这样，一旦不小心误删了crontab文件，可以用上一节所讲述的方法迅速恢复。

**3). 编辑crontab文件**

   如果希望添加、删除或编辑crontab文件中的条目，而E D I TO R环境变量又设置为v i，那么就可以用v i来编辑crontab文件，相应的命令为：

     $ crontab -e

可以像使用v i编辑其他任何文件那样修改crontab文件并退出。如果修改了某些条目或添加了新的条目，那么在保存该文件时， c r o n会对其进行必要的完整性检查。如果其中的某个域出现了超出允许范围的值，它会提示你。

我们在编辑crontab文件时，没准会加入新的条目。例如，加入下面的一条：

    # DT:delete core files,at 3.30am on 1,7,14,21,26,26 days of each month

     30 3 1,7,14,21,26 \* \* /bin/find -name "core' -exec rm {} \;

现在保存并退出。最好在crontab文件的每一个条目之上加入一条注释，这样就可以知道它的功能、运行时间，更为重要的是，知道这是哪位用户的作业。

现在让我们使用前面讲过的crontab -l命令列出它的全部信息：

    $ crontab -l

    # (crondave installed on Tue May 4 13:07:43 1999)

    # DT:ech the date to the console every 30 minites

   0,15,30,45 18-06 \* \* \* /bin/echo `date` > /dev/tty1

    # DT:delete core files,at 3.30am on 1,7,14,21,26,26 days of each month

    30 3 1,7,14,21,26 \* \* /bin/find -name "core' -exec rm {} \;

**4). 删除crontab文件**

要删除crontab文件，可以用：

    $ crontab -r

**5). 恢复丢失的crontab文件**

如果不小心误删了crontab文件，假设你在自己的$ H O M E目录下还有一个备份，那么可以将其拷贝到/var/spool/cron/<username>，其中<username>是用户名。如果由于权限问题无法完成拷贝，可以用：

     $ crontab <filename>

    其中，<filename>是你在$ H O M E目录中副本的文件名。

我建议你在自己的$ H O M E目录中保存一个该文件的副本。我就有过类似的经历，有数次误删了crontab文件（因为r键紧挨在e键的右边）。这就是为什么有些系统文档建议不要直接编辑crontab文件，而是编辑该文件的一个副本，然后重新提交新的文件。

有些crontab的变体有些怪异，所以在使用crontab命令时要格外小心。如果遗漏了任何选项，crontab可能会打开一个空文件，或者看起来像是个空文件。这时敲delete键退出，不要按<Ctrl-D>，否则你将丢失crontab文件。

# 使用实例

**实例1：每1分钟执行一次command**

**命令：**\* \* \* \* \* command

**实例2：每小时的第3和第15分钟执行**

**命令：** 3,15 \* \* \* \* command

**实例3：在上午8点到11点的第3和第15分钟执行**

**命令：**3,15 8-11 \* \* \* command

**实例4：每隔两天的上午8点到11点的第3和第15分钟执行**

**命令：**3,15 8-11 \*/2 \* \* command

**实例5：每个星期一的上午8点到11点的第3和第15分钟执行**

**命令：**3,15 8-11 \* \* 1 command

**实例6：每晚的21:30重启smb**

**命令：**30 21 \* \* \* /etc/init.d/smb restart

**实例7：每月1、10、22日的4 : 45重启smb**

**命令：**45 4 1,10,22 \* \* /etc/init.d/smb restart

**实例8：每周六、周日的1 : 10重启smb**

**命令：**10 1 \* \* 6,0 /etc/init.d/smb restart

**实例9：每天18 : 00至23 : 00之间每隔30分钟重启smb**

**命令：**0,30 18-23 \* \* \* /etc/init.d/smb restart

**实例10：每星期六的晚上11 : 00 pm重启smb**

**命令：**0 23 \* \* 6 /etc/init.d/smb restart

**实例11：每一小时重启smb**

**命令：**\* \*/1 \* \* \* /etc/init.d/smb restart

**实例12：晚上11点到早上7点之间，每隔一小时重启smb**

**命令：**\* 23-7/1 \* \* \* /etc/init.d/smb restart

**实例13：每月的4号与每周一到周三的11点重启smb**

**命令：**0 11 4 \* mon-wed /etc/init.d/smb restart

**实例14：一月一号的4点重启smb**

**命令：**0 4 1 jan \* /etc/init.d/smb restart

**实例15：每小时执行/etc/cron.hourly目录内的脚本**

**命令：01 \* \* \* \*    root run-parts /etc/cron.hourly**

**说明：**

run-parts这个参数了，如果去掉这个参数的话，后面就可以写要运行的某个脚本名，而不是目录名了

# 使用注意事项

**1. 注意环境变量问题**

有时我们创建了一个crontab，但是这个任务却无法自动执行，而手动执行这个任务却没有问题，这种情况一般是由于在crontab文件中没有配置环境变量引起的。

在crontab文件中定义多个调度任务时，需要特别注意的一个问题就是环境变量的设置，因为我们手动执行某个任务时，是在当前shell环境下进行的，程序当然能找到环境变量，而系统自动执行任务调度时，是不会加载任何环境变量的，因此，就需要在crontab文件中指定任务运行所需的所有环境变量，这样，系统执行任务调度时就没有问题了。

不要假定cron知道所需要的特殊环境，它其实并不知道。所以你要保证在shelll脚本中提供所有必要的路径和环境变量，除了一些自动设置的全局变量。所以注意如下3点：

1）脚本中涉及文件路径时写全局路径；

2）脚本执行要用到java或其他环境变量时，通过source命令引入环境变量，如：

cat start\_cbp.sh

#!/bin/sh

source /etc/profile

export RUN\_CONF=/home/d139/conf/platform/cbp/cbp\_jboss.conf

/usr/local/jboss-4.0.5/bin/run.sh -c mev &

3）当手动执行脚本OK，但是crontab死活不执行时。这时必须大胆怀疑是环境变量惹的祸，并可以尝试在crontab中直接引入环境变量解决问题。如：

0 \* \* \* \* . /etc/profile;/bin/sh /var/www/java/audit\_no\_count/bin/restart\_audit.sh

**2. 注意清理系统用户的邮件日志**

每条任务调度执行完毕，系统都会将任务输出信息通过电子邮件的形式发送给当前系统用户，这样日积月累，日志信息会非常大，可能会影响系统的正常运行，因此，将每条任务进行重定向处理非常重要。

例如，可以在crontab文件中设置如下形式，忽略日志输出：

0 \*/3 \* \* \* /usr/local/apache2/apachectl restart >/dev/null 2>&1

“/dev/null 2>&1”表示先将标准输出重定向到/dev/null，然后将标准错误重定向到标准输出，由于标准输出已经重定向到了/dev/null，因此标准错误也会重定向到/dev/null，这样日志输出问题就解决了。

**3. 系统级任务调度与用户级任务调度**

系统级任务调度主要完成系统的一些维护操作，用户级任务调度主要完成用户自定义的一些任务，可以将用户级任务调度放到系统级任务调度来完成（不建议这么做），但是反过来却不行，root用户的任务调度操作可以通过“crontab –uroot –e”来设置，也可以将调度任务直接写入/etc/crontab文件，需要注意的是，如果要定义一个定时重启系统的任务，就必须将任务放到/etc/crontab文件，即使在root用户下创建一个定时重启系统的任务也是无效的。

**4.** **其他注意事项**

新创建的cron job，不会马上执行，至少要过2分钟才执行。如果重启cron则马上执行。

当crontab突然失效时，可以尝试/etc/init.d/crond restart解决问题。或者查看日志看某个job有没有执行/报错tail -f /var/log/cron。

千万别乱运行crontab -r。它从Crontab目录（/var/spool/cron）中删除用户的Crontab文件。删除了该用户的所有crontab都没了。

在crontab中%是有特殊含义的，表示换行的意思。如果要用的话必须进行转义\%，如经常用的date ‘+%Y%m%d’在crontab里是不会执行的，应该换成date ‘+\%Y\%m\%d’。