**Linux定时任务Crond服务应用**

**作者：杨进杰 归档：学习笔记 日期： 2016/08/11**

**第1章 定时任务Crond介绍**

**1.1 Crond是什么？**

Crond是Linux 系统中用来定期执行命令或指定程序任务的一种服务或软件。一般情况下，我们安装完Centos6 Linux操作系统之后，默认便会启动Crond任务调度服务，在我们前面的系统安装及开机启动优化的设置中，我们也设置保留了Crond开机自启动。Crond服务会定期（默认每分钟检查一次）检查系统中是否有要执行的任务工作。如果有，便会根据其预先设定的定时任务规则自动执行该定时任务工作。这个Crond定时任务服务就相当于我们平时早起使用的闹钟一样。

特殊需求：crond服务搞不定了，一般工作中写脚本守护程序执行。

程序文件：程序代码组成，但是没有在计算机内执行。当前没有执行；

进程：所谓进程就是计算机中正在执行的程序；

守护进程就是一直运行的程序。

**1.2 为什么要使用Crond定时任务？**

为什么要使用定时任务呢？

例如：我们的数据库或者代码程序需要每天晚上0点做一次全备，这种情况人为操作不现实。所以我们就需要办法来进行周期性的执行任务。 这就是Linux系统的定时任务Crond。

**1.3 不同系统的定时任务与种类**

**1.3.1 Windows7系统的定时任务**

**1.3.2 Linux系统crond的定时任务**

Linux系统中定时任务调度的工作可以分为以下两种情况：

Linux系统自身定期执行的任务工作：系统周期性自行执行的任务工作，如轮循系统日志、备份系统数据、清理系统缓存等，这些任务无需我们认为干预。

[root@www html]# ls -l /var/log/message\*

-rw------- 1 root root 0 Aug 7 03:14 /var/log/messages

-rw-------. 1 root root 1300268 Jul 19 18:14 /var/log/messages-20160719

-rw-------. 1 root root 377590 Jul 24 03:14 /var/log/messages-20160724

-rw-------. 1 root root 193554 Jul 31 03:23 /var/log/messages-20160731

-rw------- 1 root root 611748 Aug 7 03:14 /var/log/messages-20160807

[root@www html]# ls -l /var/log/secure\*

-rw------- 1 root root 468 Aug 7 03:34 /var/log/secure

-rw-------. 1 root root 11240 Jul 19 18:15 /var/log/secure-20160719

-rw-------. 1 root root 9377 Jul 24 02:29 /var/log/secure-20160724

-rw-------. 1 root root 15730 Jul 30 20:00 /var/log/secure-20160731

-rw------- 1 root root 47576 Aug 7 00:52 /var/log/secure-20160807

**CentOS6 日志轮循结尾就是按日期。 像这样的工作就是由系统自身来完成的，不需要系统管理员来设置了，因此这里就不讲了。 系统自动轮循任务的设置配置路径：**

[root@www ~]# ls -l /etc/ |grep cron

-rw-------. 1 root root541 Nov 10 2015 anacrontab

drwxr-xr-x. 2 root root 4096 Jul 15 01:04 cron.d

drwxr-xr-x. 2 root root 4096 Jul 15 01:09 cron.daily

-rw-------. 1 root root 0 Nov 10 2015 cron.deny

drwxr-xr-x. 2 root root 4096 Jul 15 01:02 cron.hourly

drwxr-xr-x. 2 root root 4096 Jul 14 16:56 cron.monthly

-rw-r--r--. 1 root root457 Sep 27 2011 crontab

drwxr-xr-x. 2 root root 4096 Sep 27 2011 cron.weekly

用户执行的任务工作：某个用户或系统管理员定期要做的任务工作，例如每隔五分钟和互联网上时间服务器进行时间同步。每天凌晨1点备份数据。一般这些工作需要每个用户自行设置。

**1.3.3 Linux系统下定时任务软件种类**

Linux系统下的定时任务软件有很多，例如：at，crontab，anacron。

θ at：适合仅执行一次就结束的调度任务命令，属于突发性的工作任务。要执行at命令，还需要启动一个名为atd的服务才行。基本没有需求。

θ crontab：正如前面所说这个命令可以周期性的执行任务工作，例如：每五分钟和服务器同步时间。要执行crontab这个命令，也需要启动一个服务crond才行。重点

θ anacron：这个命令主要用于非7\*24小时开机的服务器准备的。anacron不能指定具体时间执行任务工作，而是以天为周期或者在系统每次开机后执行的任务工作。它会检测服务器停机期间应该执行，把没有进行的任务工作，并执行一遍。

注意：

crond是服务，是运行的程序，crontab是命令用来设置定时规则。

crond服务是企业生产工作中常用的重要服务at和anacron很少使用，可以忽略。

几乎每个服务器都会用到crond服务。

上千个服务器可以开发分布式定时任务项目方案。

**第2章 定时任务crond使用说明**

[root@www ~]# crontab --help

crontab: invalid option -- '-'

crontab: usage error: unrecognized option

usage: crontab [-u user] file

crontab [-u user] [ -e | -l | -r ] ###crontab -u boy -l 指定使用的用户执行任务

(default operation is replace, per 1003.2)###-l，-e参数实际上就是操作/var/spool/cron/当前用户的配置文件。

-e (edit user's crontab)###编辑crontab文件内容

-l (list user's crontab)###字母l，查看crontab文件内容

-r (delete user's crontab) ###删除crontab文件内容，用得少

-i (prompt before deleting user's crontab)###删除crontab文件内容，删除前提示确认

-s (selinux context)

[root@www ~]# crontab -l

no crontab for root

[root@www ~]# ls -l `which crontab`

-rwsr-xr-x. 1 root root 51784 Nov 10 2015 /usr/bin/crontab

**2.1.1 指令说明**

通过crontab我们可以在固定的间隔时间执行指令的系统指令或script脚本。时间间隔的单位可以是：分、时、日、月、周及以上的组合（日和周不要组合）。crond服务通过crontab命令可以很容易实现周期性的日志分析或数据备份等企业运维场景工作。

**2.1.2 使用者权限及定时任务文件**

* /etc/cron.deny 该文件里面的用户不允许使用crontab命令
* /etc/cron.allow 该文件里面的用户允许使用crontab，优先于/etc/cron.deny
* /var/spool/cron 所有用户crontab配置文件默认都存放在此目录，文件名以用户名命名

[root@www ~]# cat /var/spool/cron/root

#time sync by yjj at 2016-08-13

\*/5 \* \* \* \* /bin/sh /server/scripts/date.sh >/dev/null 2>&1

**2.1.3 定时任务指令的使用格式**

默认情况下，当用户建立定时任务规则后，该规则记录对应的配置文件会存在于/var/spool/cron中，其crontab配置文件对应的文件名与登录的用户名一致，root用户的定时任务配置文件为：/var/spool/cron/root。 crontab定时任务的书写格式很简单，用户的定时任务规则分6段，每段通过空格分隔。

基本格式：

\* \* \* \* \* cmd

cmd为要执行的命令或脚本， 如：/bin/sh /server/scripts/old.sh

分时日月周

第一个星代表分minute：00-59

第二个星代表时hour：00-23

第三个星代表日day of month：01-31

第四个星代表月month：01-12

第五个星代表周day of week：0-6（周日，可用0或者7代表）

[root@www ~]# cat /etc/crontab

SHELL=/bin/bash

PATH=/sbin:/bin:/usr/sbin:/usr/bin

MAILTO=root

HOME=/

# For details see man 4 crontabs

# Example of job definition:

# .---------------- minute (0 - 59)

# | .------------- hour (0 - 23)

# | | .---------- day of month (1 - 31)

# | | | .------- month (1 - 12) OR jan,feb,mar,apr ...

# | | | | .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat

# | | | | |

# \* \* \* \* \* user-name command to be executed

01 \* \* \* \* cmd每分钟执行一次命令

02 4 \* \* \* cmd每天凌晨04:02执行一次

22 4 \* \* 0 cmd每周日凌晨04:22执行一次

42 4 1 \* \* cmd每月1日凌晨04:42执行一次

**2.1.4 crontab语法格式中特殊符号含义**

特殊符号 含义

\* \*号表示“每”的意思

30 15 \* \* \* cmd 表示每月每周每日15:30都执行任务。

- 减号，表示分隔符，表示一个时间范围，区间段，比如01-12点，每天1点到12点的意思

每天凌晨1点到中午12点半点执行一次：

30 01-12 \* \* \* cmd

, 逗号，表示分隔时间段的意思。比如，

30 01,05,12 \* \* \* /bin/sh /scripts/old.sh 表示每天1点，5点，12点的半点执行一次/scripts/old.sh脚本

/n n代表数字，即“每隔n单位时间”，例如：每十分钟执行一次任务：\*/10 \* \* \* \* cmd，也可以写成00-59/10 \* \* \* \* cmd

**2.2 定时任务crontab实例**

crontab -e /var/spool/cron/root 会检查语法，直接使用echo，vi编辑配置不会

visudo /etc/sudoers 会检查语法，直接使用echo，vi编辑配置不会

实例2-1

[root@www ~]# crontab -e

##每两个小时整点，提醒休息

00 \*/2 \* \* \* echo "Have a break now." >> /tmp/test.txt

实例2-2

###每分钟执行一次/scripts/date.sh

\*/1 \* \* \* \* /bin/sh /scripts/date.sh###相当于

\* \* \* \* \* /bin/sh /scripts/date.sh

实例2-3

θ 30 3,12 \* \* \* /bin/sh /scripts/data.sh

每天凌晨3：30和中午12:30的时候执行一次脚本

θ 30 \*/6 \* \* \* /bin/sh /scripts/data.sh

每隔6个小时的半点执行一次脚本

θ 30 8-16/2 \* \* \* /bin/sh /scripts/data.sh

8点到16点之间每隔2个小时的半点执行一次脚本

θ 30 21 \* \* \* /bin/sh /scripts/data.sh

每天晚上21:30执行一次脚本

θ 45 4 1,10,22 \* \* /bin/sh /scripts/data.sh

每月1、10、22号凌晨4:45执行一次脚本

θ 10 1 \* \* 6,0 /bin/sh /scripts/data.sh

每周六、周日凌晨1:10分执行一次脚本

θ 00,30 18-23 \* \* \* /bin/sh /scripts/data.sh

每天18点到23点之间每隔30分钟执行一次脚本

θ 00 \*/1 \* \* \* /bin/sh /scripts/data.sh

每个小时的整点执行一次脚本

θ \* 23,00-07/1 \* \* \* /bin/sh /scripts/ data.sh ##不规范

每天23点和0点到7点之间的每一分钟执行一次脚本

θ 00 11 \* 4 1-3 /bin/sh /scripts/data.sh###周和日不要同时使用，不然可能达不到效果

4月的每周一到周三11:00执行一次脚本

实例2-4

θ 5月5日上午9:00上课

00 09 05 05 \* 上课

θ 每周日上午9:30上课

30 09 \* \* 0 上课

**第3章 生产环境crontab规范**

**3.1 规范定时任务两例**

例1：每分钟打印一次自己的名字拼音到“/server/log/自己的名字命令的文件”中。

#print my name to log by yangjinjie at 20160813

\* \* \* \* \* echo yangjinjie >> /server/log/yangjinjie.log##注意文件夹需要存在

\*/1 \* \* \* \* echo yangjinjie >> /server/log/yangjinjie.log ##这样写也可以

小结：

λ 定时任务要加注释

λ /server/log目录必须存在，没有的话需要事先创建

λ 定时任务中的路径一定要使用绝对路径

λ crond服务必须开启运行

λ 查看定时任务日志 tail /var/log/cron

--

例2：每周六、日上午9:00和14:00上课（执行/server/scripts/data.sh），要求/server/scripts/data.sh脚本的功能是打印当天的日期：格式为2016-03-23

技巧：

1. 定时任务一定尽量用脚本实现；

2. 命令行操作成功，然后放入脚本执行成功，最后再配置任务；

3. 复制操作成功的命令行脚本，在定时任务里配置；

4. /bin/sh /server/scripts/data.sh

**创建目录，编写脚本**

[root@www ~]# mkdir -p /server/scripts/

[root@www ~]# echo 'date +%F' >/server/scripts/data.sh

[root@www ~]# cat /server/scripts/data.sh

date +%F

[root@www ~]# /server/scripts/data.sh

-bash: /server/scripts/data.sh: Permission denied

[root@www ~]# sh /server/scripts/data.sh

2016-08-13

书写定时任务

[root@www ~]# echo "#study">>/var/spool/cron/root

[root@www ~]# echo "00 09,14 \* \* 6,7 /bin/sh /server/scripts/data.sh" >>/var/spool/cron/root

[root@www ~]# cat /var/spool/cron/root

#study

00 09,14 \* \* 6,7 /bin/sh /server/scripts/data.sh

**第4章 书写crond定时任务要领**

**4.1 命令行测试**

[root@www ~]# /usr/sbin/ntpdate ntp1.aliyun.com

13 Aug 12:13:09 ntpdate[85955]: adjust time server 182.92.12.11 offset 0.015303 sec

**4.2 为定时任务添加必要注释**

有注释，就知道定时任务的作用等等，这是个好习惯和规范。

[root@www ~]# crontab -l

#time sync by yjj at 2016-08-13

\*/5 \* \* \* \* /usr/sbin/ntpdate ntp1.aliyun.com >/dev/null 2>&1

**4.3 定时任务使用脚本**

[root@www ~]# crontab -l

#time sync by yjj at 2016-08-13

\*/5 \* \* \* \* /bin/sh /server/scripts/date.sh >/dev/null 2>&1

**4.4 定时任务执行的脚本要规范路径，并使用/bin/sh执行，如：/server/scripts**

###脚本路径使用绝对路径

###使用/bin/sh执行脚本，否则有可能因为忘了为脚本设定执行权限，从而无法完成任务。

[root@www ~]# crontab -l

#time sync by yjj at 2016-08-13

\*/5 \* \* \* \* /bin/sh /server/scripts/date.sh >/dev/null 2>&1

**4.5 定时任务结尾加 >/dev/null 2>&1**

[root@www ~]# crontab -l

#time sync by yjj at 2016-08-13

\*/5 \* \* \* \* /bin/sh /server/scripts/date.sh >/dev/null 2>&1

###

#/dev/null为特殊的字符设备文件，表示黑洞设备或空设备。

如果定时任务结尾不加 >/dev/null 2>&1,很容易导致硬盘inode空间被占满，从而系统服务不正常（var/spool/clientmqueue邮件临时队列目录，垃圾文件存放于此，如果是centos 6.4系统，默认不装sendmail服务，所以不会有这个目录。）

**4.6 在指定用户下执行相关定时任务**

这里要特别注意不同用户的环境变量问题，如果是调用了系统环境变量/etc/profile，最好在程序脚本中将用到的环境变量重新export下。

**4.7 生产任务程序不要随意打印输出信息**

在调试好脚本程序后，应尽量把DEBUG及命令输出的内容信息屏蔽掉，如果确实需要输出日志，可定向到日志文件里，避免产生系统垃圾。

**4.8 小结**

1. 先在命令行搞定

2. 定时任务添加注释

3. 定时任务最好使用脚本

4. 取消输出

5. 使用/bin/sh执行脚本

6. 脚本路径使用绝对路径

7. 定时任务加定向到空

配置定时任务规范操作过程

①首先要在命令行操作成功，然后复制成功的命令到脚本里，在各个细小环境减少出错的机会。

②然后测试脚本，测试成功后，复制脚本的规范路径到定时任务配置里，不要手敲。

③先在测试环境下测试，然后正式环境规范部署。

**第5章 工作中调试定时任务的方法**

规范的公司开发和运维人员操作流程：

1. 个人的开发配置环境；

2. 办公室的测试环境；

3. IDC机房的测试环境；

4. IDC机房的正式环境（分组、灰度发布）。

**5.1 增加执行频率调试任务**

在调试时，把任务执行频率调快一点，看能不能正常执行，如果正常，那就没问题了，再改成需要的任务的执行时间。

注意：有些任务时不允许频繁执行的，例如：定时往数据库里插入数据，这样的任务要在测试机上测试好，然后正式线上出问题的机会就少了。

**5.2 调整系统时间调试任务**

不能用于生产环境

用正确的执行任务的时间，设置完成后，可以修改下系统当前时间，改成任务执行时间的前几分钟来测试（或者重启定时任务服务）

**5.3 通过脚本日志输出调试定时任务**

在脚本中加入日志输出，然后把输出打到指定的日志中，然后观察日志内容的结果，看是否正确执行。

**5.4 注意一些任务命令带来的问题**

注意：

\* \* \* \* \* echo "==">>/tmp/lee.log >/dev/null 2>&1

这里隐藏的无法正确执行的任务配置，原因是前面多了>>,或者去掉结尾的 >/dev/null 2>&1。

\* \* \* \* \* tar zcvf /tmp/oldboy\_$(date +%F).tar.gz /etc/hosts ###这里的百分号需要转义，而使用脚本的话就无需转义

**5.5 注意环境变量导致的定时任务故障（java）**

例如：在调试java程序任务的时候，注意环境变量，把环境变量的定义加到脚本里。

**5.6 通过定时任务日志调试定时任务（/var/log/cron）**

**5.7 其他稀奇古怪的问题调试方法**

**5.8 使用sh -x 调试脚本**

[root@www ~]# sh -x /server/scripts/date.sh

+ /usr/sbin/ntpdate ntp1.aliyun.com

13 Aug 13:44:54 ntpdate[86651]: step time server 182.92.12.11 offset 0.783190 sec