[精华] 用 chkconfig 管理 Init 脚本 [转贴]

http://www.chinaunix.net 作者 :sky-walker 发表于 : 2006-04-12 16:17:43

【发表评论】 【查看原文】 【Linux讨论区】【关闭】

用红帽子的 chkconfig 管理 Init 脚本
Jimmy Ball 翻译 : Feng Dahui

转载 LinuxAid

你的管理工具中多了个简单但非常有用的东东。

我喜欢发现新的UNIX 命令，尤其是那些关于系统管理的。当我得知红帽子发布 chkconfig 这个工具，我想起了在 IRIX--一个从 Silicon Graphics Inc.而来的UNIX变种--下的 chkconfig。 IRIX的 chkconfig被用来激活/禁止系统初始化的时候的服务，无需编辑，重命名或是移动 /etc中的 init脚本。

类似， Red Hat设计 chkconfig的目的就是用来管理系统初始化的时候启动的服务。不过，在我仔细阅读手册并作了些测试后，我很快发现 Red Hat扩展了 chkconfig，通过管理 init脚本的符号连接得以最终控制启动关闭时的系统任务，真是节省时间！

关于启动的基础知识

当你的 linux启动时，它显示的第一个进程是 init。如果你以前没看到过显示 init进程，输入：

```
# ps -ef | grep init
```

就会看到 init的PID。 简而言之， init 运行 /etc/inittab中描述的任务。

/etc/inittab中说明的任务在 init之后就会启动， 不过其它的任务启动很简单。例如，默认情况下 Red Hat的 /etc/inittab 对 Ctrl-Alt-Delete键序设置了一个陷阱（trap），当这些键在控制台模式下（不是 xdm）同时按下，就会运行 shutdown 命令。 在启动的时候， init基于 /etc/inittab 的设置选项设定这个特性，不过在这个键序发生的时候才会执行。

inittab 的格式允许以 "#"开始注释行，正常的条目用 ":" 界定。遵从如下的格式：

```
id:runlevel:action:process
```

id 代表用户定义的唯一的标志，
runlevel 可以使 0-6的组合或者为空，
action 来自一个关键词 keyword 描述 init如何对待 process，
process 是要执行的命令。

描述 action字段的各种关键字可以在 inittab的手册中找到。常用的关键字，不是全部，UNIX平台包括这些：

initdefault--定义一个系统启动后进入的运行级
wait --会被执行一次的进程 (当进入运行级的时候)。 init 进程将等待这个进程被终止。
boot --定义一个启动的时候执行的进程 .
bootwait --与 boot 类似 ，不过 init在继续运行前等待进程的终止
sysinit --定义一个进程在 boot的时候执行，在任何 boot或者 bootwait inittab 条目的前面执行。

runlevel 字段指明系统状态。例如，运行级 0代表系统关机，运行级 6代表系统重启 。不幸的事，不是所有的 Linux 发布都遵循同样的运行级定义 。在 Red Hat中，默认情况下支持下面这些

0. 系统挂起
1. 但用户 Single-user mode

2. 多用户，没有 NFS
3. 完整的多用户 Complete multiuser mode
4. 用户自定义
5. X11（XDM 登陆）
6. 重新启动

每一个运行级在 /etc/rc.d 下都有个相应的目录。如运行级 5，目录就是 /etc/rc.d/rc5.d。包含启动这个运行级的时候运行的相关任务的相关文件。在 Red Hat 中，这些文件一般都是 shell 脚本的符号连接，可以在 /etc/rc.d/init.d 中找到。

让我们用一个简单的例子看一下这些东西， 下面这两个例子行来自我们的 inittab 文件：

```
id:3:initdefault:
l3:3:wait:/etc/rc.d/rc 3
```

在 Red Hat 系统中这很典型。一旦 init 被启动，读取 /etc/inittab。从第一行，我们知道 init 将在系统启动后从运行运行级 3。一旦我们到了那个运行级，第二行告诉 init 去运行脚本 /etc/rc.d/rc 3 并且在执行前等待终止。

在 /etc/rc.d 目录的 rc 脚本收到 3 作为一个参数。这个 3 相当于运行级 3。结果 rc 脚本执行 /etc/rc.d/rc3.d 目录中的所有脚本。它首先用参数 "stop" 执行所有 K（代表 "kill" 杀掉进程或者服务）打头的脚本，接下来，它运行所有以 字母 S 打头的脚本，带有参数 "start" 启动进程或者服务。最后要指明，K 和 S 脚本的执行顺序是基于排序的；名为 S90mysql 的脚本将在 S95httpd 之前执行。

/etc/rc.d/rc3.d 中的脚本实际是对 /etc/rc.d/init.d 中文件的符号连接。UNIX 管理员可以在 rc3.d 中放制文件，实际情况下 Red Hat 的 init.d 目录是所有脚本的第一位置，然后声称逻辑连接到 rc*.d 目录。手工作这些文件的管理很烦人、琐碎。chkconfig 现在接手这件事情！Red Hat 的这个 chkconfig 工具就是专为管理 /etc/rc.d/rc[0-6].d 中的符号连接而设计。

查看 chkconfig 的项（Entries）

chkconfig 的二进制软件在 /sbin 下，默认权限允许任何用户执行。不过没有 root 权限的用户只能察看当前的 chkconfig 配置。输入

```
[root]# chkconfig --list | grep on
```

输出的部分内容大致如下：

```
amd 0:off 1:off 2:off 3:off 4:on 5:on 6:off
apmd 0:off 1:off 2:on 3:off 4:on 5:off 6:off
arpwatch 0:off 1:off 2:off 3:off 4:off 5:off 6:off
atd 0:off 1:off 2:off 3:on 4:on 5:on 6:off
autofs 0:off 1:off 2:off 3:off 4:off 5:off 6:off
named 0:off 1:off 2:off 3:off 4:off 5:off 6:off
bootparamd 0:off 1:off 2:off 3:off 4:off 5:off 6:off
keytable 0:off 1:off 2:on 3:on 4:on 5:on 6:off
crond 0:off 1:off 2:on 3:on 4:on 5:on 6:off
syslog 0:off 1:off 2:on 3:on 4:on 5:on 6:off
netfs 0:off 1:off 2:off 3:on 4:on 5:on 6:off
network 0:off 1:off 2:on 3:on 4:on 5:on 6:off
```

在输出的每一行，最开始的段代表在 /etc/rc.d/init.d 中的 init 脚本名。其余的区段表示脚本进入各个运行级时的各运行级 0-6 的状态。例如，crond 应当在进入运行级 2，3，4，5 的时候启动，当进入 0，1 and 6 的时候停止。我们可以通过 find 命令查找在 /etc/rc.d 中所有 crond 结尾的文件确信我们设置的正确性：

```
[root]# find /etc/rc.d -name '*crond' -print
/etc/rc.d/init.d/crond
/etc/rc.d/rc0.d/K60crond
/etc/rc.d/rc1.d/K60crond
/etc/rc.d/rc2.d/S40crond
/etc/rc.d/rc3.d/S40crond
/etc/rc.d/rc4.d/S40crond
/etc/rc.d/rc5.d/S40crond
/etc/rc.d/rc6.d/K60crond
```

注意 chkconfig 报告的每个 "off" 节 (0，1，6)，一个 kill 脚本存在 script is in place 每一个 "on" 节 (2，3，4，5)，有一个 start 脚本。接下来，执行一个不同的 find 命令以确信每个发现的文件

的类型：

```
[root]# find /etc/rc.d -name '*crond' -exec file {} ;
/etc/rc.d/init.d/crond: Bourne shell script text
/etc/rc.d/rc0.d/K60crond: symbolic link to
../init.d/crond
/etc/rc.d/rc1.d/K60crond: symbolic link to
../init.d/crond
/etc/rc.d/rc2.d/S40crond: symbolic link to
../init.d/crond
/etc/rc.d/rc3.d/S40crond: symbolic link to
../init.d/crond
/etc/rc.d/rc4.d/S40crond: symbolic link to
../init.d/crond
/etc/rc.d/rc5.d/S40crond: symbolic link to
../init.d/crond
/etc/rc.d/rc6.d/K60crond: symbolic link to
../init.d/crond
```

这表明在 init.d中找到的 crond是一个 shell脚本，找到的所有其他的文件都是对 crond 脚本的符号连接。

调整 chkconfig项

调整 chkconfig的 项几乎和列出现在的设置一样容易。格式：

```
chkconfig [--level <运行级>;] <名字>;
```

例如，如果我们决定在运行级 2禁止 crond,

```
# chkconfig --level 2 crond off
```

(root执行) 会在运行级 2关掉 crond. 运行 chkconfig --list会确信 crond的配置已经被调整。 更进一步，下面的 find 命令 command 显示一个 kill脚本已经在目录 rc2.d中代替了 start脚本 ：

```
[root]# find /etc/rc.d -name '*crond' -print
/etc/rc.d/init.d/crond
/etc/rc.d/rc0.d/K60crond
/etc/rc.d/rc1.d/K60crond
/etc/rc.d/rc2.d/K60crond
/etc/rc.d/rc3.d/S40crond
/etc/rc.d/rc4.d/S40crond
/etc/rc.d/rc5.d/S40crond
/etc/rc.d/rc6.d/K60crond
```

紧记 chkconfig不是立即自动禁止或激活一个服务的 它只是简单的改变了符号连接，超级用户可以用这个命令 /etc/rc.d/init.d/crond stop立刻禁止 crond服务。最后，你可以用一个命令行激活 /禁止多个运行级的某个命令 。例如输入：

```
chkconfig --levels 2345 crond on
```

会设定 crond 在运行级 2, 3, 4 和 5启动。

删掉一项

有的时候，删掉一个服务也很恰当。 例如，针对 sendmail, 在客户机上导入本地账号的邮件没有必要。运行 sendmail最为守护进程就不是必要的了。这种情况，我发现禁止 sendmail服务很有必要，减少了潜在的安全问题，从 chkconfig中删掉 sendmail, 输入：

```
chkconfig --del sendmail
```

在下面，我们的 find命令显示该处没有符号连接了，不过 sendmail的 init脚本仍然有：

```
[root]# find /etc/rc.d -name '*sendmail' -print /etc/rc.d/init.d/sendmail
```

在我看来这很完美。脚本保留了，万一 sendmail 需要作为一个服务实现呢？不过所有的符号连接去掉了。我们能在每一个运行级禁止 sendmail 服务，这将在每一个 rc*.d子目录中放置一个 kill脚本，虽然 sendmail从不

在初始化阶段启动，是个不必要的任务，可是，我曾看到一些系统管理员需要在特定的场合手工启动服务 。把 kil脚本留在那里确保可以干净的杀掉服务。

添加一个 chkconfig 项

到目前为止，一切顺利，我们已经知道使用 chkconfig如何查看、调整、删掉服务。现在添加一个新的服务。看下面的脚本。

```
--------------------------------------------------
Listing 1. Oracle Script
--------------------------------------------------
#!/bin/sh<\n>; 红帽子的

#chkconfig: 2345 80 05
#description: Oracle 8 Server

ORA_HOME=/usr/home/oracle/product/8.0.5
ORA_OWNER=oracle

if [ ! -f $ORA_HOME/bin/dbstart ]
then
echo "Oracle startup: cannot start"
exit
fi

case "$1" in
"start")
su-$ORA_OWNER -c $ORA_HOME/bin/dbstart
su-$ORA_OWNER -c "$ORA_HOME/bin/lsnrctl start"
;;
"stop")
su-$ORA_OWNER -c $ORA_HOME/bin/dbshut
su-$ORA_OWNER -c "$ORA_HOME/bin/lsnrctl stop"
;;
esac
--------------------------------------------------
```

使用这个脚本， Oracle 8可以以参数 "start"启动，以 "stop" 参数停止。它符合 init脚本的最小要求可以和 /etc/rc.d/rc脚本联合使用。

把脚本放到 /etc/rc.d/init.d中并运行 (以 root) ：

```
chmod +x /etc/rc.d/init.d/oracle
```

使你的脚本可执行 。如果你担心普通用户察看这个脚本，你可以设定更严格的文件权限 。只要这个脚本可以被 root作为单独的脚本运行就可以。

注意脚本中的两行注释 ：

```
#chkconfig: 2345 80 05
#description: Oracle 8 Server
```

chkconfig需要这些行来决定如何实现初始运行级添加服务，如何设定启动和停止顺序的优先级 。这些行指明脚本将为运行级 2， 3， 4， 5启动 Oracle 8 服务。另外，启动优先权将被设定为 80而停止优先权设定为 05。

现在脚本在合适的位置，并且有合适的执行权限，以及恰当的 chkconfig注释，我们可以添加 init脚本，作为 root，

```
# chkconfig --add oracle.
```

用 chkconfig的查询，我们能核实我们所作的添加 ：

```
[root]# chkconfig --list | grep oracle
oracle 0:off 1:off 2:on 3:on 4:on 5:on 6:off
```

而且，我们可以用标准的 find 命令察看 chkconfig如何设定符号连接：

```
[root]# find /etc/rc.d -name '*oracle' -print

/etc/rc.d/init.d/oracle
/etc/rc.d/rc0.d/K05oracle
/etc/rc.d/rc1.d/K05oracle
/etc/rc.d/rc2.d/S80oracle
/etc/rc.d/rc3.d/S80oracle
/etc/rc.d/rc4.d/S80oracle
/etc/rc.d/rc5.d/S80oracle
/etc/rc.d/rc6.d/K05oracle
```

正如需要的那样， kill连接的名字包含优先权 05而 start连接包含 80。如果你需要调整优先权， (e.g.我们停止的优先权需要设为 03)，简单的调整 oracle init脚本的 chkconfig 注释行并运行 reset命令 command，如下所示。符号连接会被改名：

```
[root]# chkconfig oracle reset
[root]# find /etc/rc.d -name '*oracle' -print
/etc/rc.d/init.d/oracle
/etc/rc.d/rc0.d/K03oracle
/etc/rc.d/rc1.d/K03oracle
/etc/rc.d/rc2.d/S80oracle
/etc/rc.d/rc3.d/S80oracle
/etc/rc.d/rc4.d/S80oracle
/etc/rc.d/rc5.d/S80oracle
/etc/rc.d/rc6.d/K03oracle
```

Red Hat 7中的改进

大家可能都知道了， inetd在 Red Hat 7种已经被 xinetd 所取代。而且， chkconfig 的功能已经被扩展，可以管理一些 xinetd的 Internet服务。例子如下：

```
[root]# chkconfig --list
...
xinetd based services:
finger: on
linuxconf-web: off
rexec: off
rlogin: off
rsh: off
ntalk: off
talk: off
telnet: on
tftp: off
wu-ftpd: on
```

禁掉一个 xinetd 服务，可能是 finger，你应该输入：

```
[root]# chkconfig finger off.
```

很简洁啊，呵呵。可是，这里有个问题。当配置已经改变，命令 /etc/init.d/xinetd reload指明 xinetd 自动重载入新的配置，被 chkconfig执行。这个脚本运行一个带有 SIGUSR2信号的 kill 指示 xinetd 进行一个"硬"重配置。

那意味着什么？哦，当我测试的时候，通过 xinetd 提供的活动服务 (i.e.，Telnet，FTP，etc.)立刻被中止。

如果你能计划在最合适的时间启动 /禁止你的系统上的服务，可能不是个问题。作为一种替代方式，你可以调整你的 /etc/init.d/xinetd脚本，这样 reload 选项发送一个 SIGUSR1 信号。 这是个"软"重配置。这将重启动你的服务而不中断你现存的连接。 chkconfig 管理下，添加 xinetd服务只要简单的添加 xinetd服务文件到 /etc/xinetd.d目录中。 chkconfig会自动的"捡起"它并使其可用，通过 chkconfig 工具进行管理。简洁阿！

结论

现在你已经应该认识到红帽子的 chkconfig工具管理 init脚本的好处了，虽然它的功能似乎简单了些，但是它节

省时间，这使其成为一个系统管理员适用的命令，值得记牢。

---

jeffyan 回复于：2003-11-16 12:52:32

thx !!!

双眼皮的猪 回复于：2003-11-16 13:45:57

前几天刚看过。不过是在 linuxeden

Fenng 回复于：2004-02-21 20:40:39

hoho ，

偶的这么久的翻译档了

割鹿刀 回复于：2006-04-12 15:04:29

好贴！！！看过，谢了

---

kaka_sun 回复于：2006-04-12 16:17:43

好啊，不错

原文链接：http://linux.chinaunix.net/bbs/viewthread.php?tid=203105
转载请注明作者名及原文出处
