Linux 关机命令分析

任桥伟
MSN：ilttv@hotmail.com
Email：ilttv@163.com

linux下有关关机重启的命令主要有：shutdown、reboot、halt、poweroff、telinit、init。它们都可以达到关机或重启的目的，但是每个命令的工作流程并不一样。它们都由软件包sysvinit产生，你可以从网上下载它的源码来分析各个命令的流程，同时，sysvinit也包含了与登陆等有关的命令。下面分析sysvinit-2.86软件包的源码，同时参考它们的手册来对它们深入了解。

1、从Makefile可以知道，上面的命令并不都是互相独立的，poweroff、reboot是halt的链接，telinit是init的链接。

```
ln -sf halt $(ROOT)/sbin/reboot
ln -sf halt $(ROOT)/sbin/poweroff
ln -sf init $(ROOT)/sbin/telinit
```

在终端输入

```
# ls -l /sbin/poweroff
# ls -l /sbin/reboot
# ls -l /sbin/telinit
```

也可以看到上面的结果。

2、halt。

参数说明：

[-n] 防止sync系统调用，它用在用fsck修补根分区之后，以阻止内核用老版本的超级块〔superblock〕覆盖修补过的超级块。

[-w] 并不是真正的重启或关机，只是写wtmp〔/var/log/wtmp〕纪录。

[-d] 不写wtmp纪录〔已包含在选项[-n]中〕。

[-f] 没有调用shutdown而强制关机或重启（halt/reboot）。

[-h] 使硬盘处于standby模式。

[-i] 关掉所有的网络接口。

[-p] 该选项为缺省选项。就是关机时调用poweroff。

前面已经知道，poweroff、reboot是halt的链接，halt会首先判断用户执行的是否是poweroff和reboot中的一个。如果执行的是poweroff，则等效于-p参数，执行reboot的情况将在下面说明。

解析命令行参数后，会调用geteuid系统调用判断是否为root用户，如果为普通用户，halt退出。

接下来即是针对不同的参数执行不同操作的过程。首先是不带任何参数的情况（或者参数中不含-w或-f），halt会通过INIT_VERSION和RUNLEVEL环境变量，或者读取/var/run/utmp文件（通过linux的用户组函数）获得系统所在的运行级。如果系统不在0和6运行级，会判断执行的是否reboot，如果用户执行reboot，则调用"shutdown -r"；否则执行"shutdown -h"。

其它情况下，都是通过调用reboot系统调用来达到关机或重启的目的，有关reboot系统调用，可以使用man 2 reboot命令查看它的手册。如果开启了CTRL-ALT-DEL，同时按下CTRL-ALT-DEL时，会调用reboot(RB_ENABLE_CAD)重启；如果用户执行的是poweroff命令，会调用reboot(RB_POWER_OFF)关机。

按照默认设置，/etc/inittab文件指定你的系统可在控制台使用CTRL-ALT-DEL键组合来关闭并重启

系统。如果你想完全禁止这个功能，需要将/etc/inittab文件中下面一行注释掉：

ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -r now

上面命令里的-a选项通知shutdown命令寻找/etc/shutdown.allow文件，并把CTRL-ALT-DEL的功能限定给某些用户。详见shutdown命令部分。

3、运行级。

所谓运行级别是一种系统软件环境配置，在此特定的环境中只允许某一组选定的进程存在。 init 给不同的运行级别派生的进程在 /etc/inittab 文件中定义。 Init 可以启动到8个不同的运行级别上：0-6 和 S 或 s。运行级别可以由超级用户通过 telinit 命令来转换，此命令可以将转换信号传递给 init，告诉它切换到哪个运行级别。

运行级别0，1，和 6为系统保留的专用运行级别。运行级别 0 用来关机，运行级别 6 用来重启，运行级别 1 用来使计算机进入单用户模式。运行级别 S 不是给我们直接使用的，更多是为进入运行级别 1 时运行某些可执行脚本时被调用。

运行级 7-9 虽然在文档中说明，但也可以使用。不使用它们的原因只是因为"传统" Unix 变种不使用这些运行级别。另外，运行级别 S 和运行级别 s 实际上是相同的，它们只是系统内同一运行级别的两个不同的别名而已。

可以使用sysvinit包的runlevel命令获得系统当前的运行级，不用加任何参数，直接在终端执行runlevel即可（也是通过用户组相关的函数读取/var/run/utmp文件获得）。下面是几个运行级的简单介绍：

0 - 关机（千万不要把initdefault 设置为0 ）
1 - 单用户模式
2 - 多用户，但是没有 NFS
3 - 完全多用户模式
4 - 没有用到
5 - X11
6 - 重启（千万不要把initdefault 设置为6 ）

4、shutdown。

shutdown 命令可以安全地关闭或重启Linux系统，它在系统关闭之前给系统上的所有登录用户提示一条警告信息。该命令还允许用户指定一个时间参数，可以是一个精确的时间，也可以是从现在开始的一个时间段。所有进程都将接收到 SIGTERM 信号。这可以使 vi 等程序有时间将处于编辑状态的文件进行存储，邮件和新闻程序进程则可以将所有缓冲池内的数据进行适当的清除等等。

shutdown 通过通知 init 进程，要求它改换运行级别来实现。运行级别 0 用来关闭系统，运行级别 6 用来重启系统，运行级别 1 用来使系统进入执行系统管理任务状态，如果没有给出 -h 或 -r 参数时，这是 shutdown 命令的默认工作状态。系统执行该命令后，会自动进行数据同步的工作。

shutdown [-akrhHPfnc] [-t secs] time [warning message]

参数说明:

[-a] 使用/etc/showdown.allow文件。

[-t sec] 通知init在转换到其它运行级别前，发送警告(warning)信号后延时(sec)秒数后再发送关闭(kill)信号。

[-r] 重启。

[-k] 并不真正关机，只是送警告信号给每位登录者〔login〕。

[-h] 关机后关闭电源〔halt〕。

[-n] 不用init，而是自己来关机。不鼓励使用这个选项，而且该选项所产生的后果往往不总是你所预期得到的。

[-c] 取消目前正在执行的shutdown。所以这个选项当然没有时间参数，但是可以输入一个用来解释的讯息，而这信息将会送到每位使用者。

[-f] 重启〔reboot〕时忽略fsck。

[-F] 重启〔reboot〕时强迫fsck。

time 关机时间。整个参数是必须的。格式可以有很多种。首先，可以是 hh:mm 格式的绝对时间，其中 hh 指的是小时（一到二位数），mm 指的是分钟（二位数）。第二种是 +m 格式，其中 m 指的是等待的分钟数。 now 是 +0 的别名。

warning-message 发送给所有用户的消息。

与halt一样，shutdown调用getuid系统调用判断是否为root用户，如果为普通用户，调用exit(1)退出。

接下来会解析命令行参数。如果带-a参数，会检测是否存在/etc/shutdown.allow文件。接下来它比较文件中的登录名与虚拟终端的登录用户列表 (在/var/run/utmp)。只有当授权的用户之一或者 root 登录了，它才会继续。否则，它会在终端输出 shutdown: no authorized users logged in
/etc/shutdown.allow 的格式是每行一个用户名。允许出现空行和注释行。如果 /etc/shutdown.allow 不存在，-a 参数将被忽略。

执行shutdown时，会产生/var/run/shutdown.pid文件，里面放的是运行shutdown的进程pid，带有-c参数时，会检测这个pid，如果小于0则在终端输出错误信息：

shutdown: cannot find pid of running shutdown

否则杀死shutdown进程。

带有-f参数时，会创建/fastboot文件，这个文件在系统重启时会被检测到。启动脚本rc会检测是否存在这个文件，如果有，表示系统正常关闭，就不再执行fsck，之后，启动进程删除/fastboot。

带有-F参数时，会创建/forcefsck文件，这个文件再系统重启时会被检测到。启动脚本rc会检测是否存在这个文件，如果有，就执行fsck force，这时，即使正常卸载的文件系统也会被检测。之后，启动进程删除/forcefask。

如果有使用time参数，shutdown会创建/etc/nologin文件，禁止新用户登陆，除非shutdown向init发信号前意外终止。在调用init改变运行级前会删除这个文件。

5、init。
Init 是所有进程的父进程。它的首要任务是从一个存储在文件 /etc/inittab 里面的脚本里创建进程。

init 通过检查自己的进程号来判断自己是 init 还是 telinit ；真的 init 的进程号永远都是 1。
如果init发现执行的是telinit，它会调用函数

```
int telinit(char *progname, int argc, char **argv);
```

执行telinit的操作。如果执行的是真正的init，它会调用函数

```
int init_main();
```

进入主循环。

如果执行init，用法是：

```
init [-a] [-s] [-b] [-z xxx] [0123456s]
```

如果执行的是telinit，用法是：

```
telinit [-t sec] [0123456sSQqabcUu]
```

但执行telinit时，telinit函数仍然通过向init fifo（/dev/inidctl）写入命令的方式通知init执行相应的操作。

参数说明：

0,1,2,3,4,5,6 将运行级别切换到指定的运行级别。

a,b,c 只运行那些 /etc/inittab 文件中运行级别是 a，b 或 c 的记录。

Q,q 通知 init 重新检测 /etc/inittab 文件。

S,s 将运行级别切换到单用户模式下。

U,u 自动重启（保留状态），此操作不会对文件/etc/inittab进行重新检测。执行此操作时，运行级别必须处在 Ss12345 之一，否则，该请求将被忽略。

-t sec 告诉 init 两次发送 SIGTERM 和 SIGKILL 信号的时间间隔。默认值是 5 秒。

init会对终端进行一些默认的设置，这里有一些快捷键可以使用，比如：

ctrl+d 退出登陆，等效于logout命令

ctrl+c 杀死应用程序

ctrl+q

ctrl+s 暂停应用程序运行，可用ctrl+q回复运行

ctrl+z 挂起应用程序，此时ps显示的该进程状态变为T

ctrl+x

接下来，init将查找/etc/inittab文件，看看是否有类型为 initdefault 的记录. initdefault 记录决定系统初始运行级别。如果没有这条记录（或者根本就没有 /etc/inittab ），那么，必须在系统控制台输入想要进入的运行级别。然后，init会解析/etc/inittab 文件中的各个条目并执行相应操作。

运行级别 S 或 s 把系统带入单用户模式，此模式不需要 /etc/initttab 文件。单用户模式中，/sbin/sulogin 会在 /dev/console 这个设备上打开。

当第一次进入多用户模式时，init 会执行boot 和 bootwait 记录以便在用户可以登录之前挂载文件系统。然后再执行相应指定的各进程。

当启动一个新的进程时， init 会检查是否存在 /etc/initscript 文件。如果存在该文件，则使用该脚本来启动该进程。

如果系统中存在文件 /var/run/utmp 和 /var/log/wtmp，那么当每个子进程终止时，init 会将终止信息和原因记录进这两个文件中。

当 init 启动了所有指定的子进程后，它会不断地侦测系统进程情况，如：它的某个子进程被终止、电源失效、或由 telinit 发出的改变运行级别的信号。当它接受到以上的这些信号之一时，它会自动重新扫描 /etc/inittab 文件，并执行相应操作。所以，新的记录可以随时加入到此文件中。并且，init 仍然等待系统发出了上述信号。在更新了各种系统文件后，如果你希望得到即时的更新，你可以使用telinit Q 或 q 命令来唤醒 init 让它即刻重新检测/etc/inittab 文件。

当 init 得到更新运行级别的请求， init会向所有没有在新运行级别中定义的进程发送一个警告信号 SIGTERM 。在等待 5 秒钟之后，它会发出强制中断所有进程的运行的信号 SIGKILL 。注意，init 假设所有的这些进程（包括它们的后代）都仍然在 init 最初创建它们的同一进程组里。如果有任何进程改变了它们的进程组，那么它就收不到这些信号。这样的进程，你需要分别进行手工的终止。

6、inittab。

/etc/inittab定义了系统缺省运行级别，系统进入新运行级别需要做什么。inittab文件的格式：

id:runlevel:action:process

id，用来唯一标志表项。它是一个字符串，对于getty或mingetty等其他login程序项，要求id与tty的编号相同，否则getty程序将不能正常工作。

runlevel，是init所处于的运行级别的标识，一般使用0-6以及S或s，也可以是空，空则代表运行级别0～6。当请求init改变运行级别时，那些runlevel字段中不包括新运行级别的进程将收到SIGTERM警告信号，并且最后被杀死；只有a、b、c启动的命令外（a、b、c不是真正的运行级别）。

action，告诉init执行的动作，即如何处理process字段指定的进程。action字段允许的值及对应的动作分别为：

1）respawn：如果process字段指定的进程没有运行，则启动该进程，init不等待处理结束，而是继续扫描inittab文件中的后续进程，当这样的进程终止时，init会重新启动它，如果这样的进程已经运行，则什么也不做。

2）wait：启动process字段指定的进程，并等到处理结束才去处理inittab中的下一记录项。

3）once：启动process字段指定的进程，不等待处理结束就去处理下一记录项。当这样的进程终止时，也不再重新启动它，在进入新的运行级别时，如果这样的进程仍在运行，init也不重新启动它。

4）boot：只有在系统启动时，init才处理这样的记录项，启动相应进程，并不等待处理结束就去处理下一个记录项。当这样的进程终止时，系统也不重启它。

5）bootwait：系统启动后，当第一次从单用户模式进入多用户模式时处理这样的记录项，init启动这样的进程，并且等待它的处理结束，然后再进行下一个记录项的处理，当这样的进程终止时，系统也不重启它。

6）powerfail：当init接到断电的信号（SIGPWR）时，处理指定的进程。

7）powerwait：当init接到断电的信号（SIGPWR）时，处理指定的进程，并且等到处理结束才去检查其他的记录项。

8）off：如果指定的进程正在运行，init就给它发SIGTERM警告信号，在向它发出信号SIGKILL强制其结束之前等待5秒，如果这样的进程不存在，则忽略这一项。

9）ondemand：功能通respawn，不同的是，与具体的运行级别无关，只用于runlevel字段是a、b、c的那些记录项。

10）sysinit：指定的进程在访问控制台之前执行，这样的记录项仅用于对某些设备的初始化，目的是为了使init在这样的设备上向用户提问有关运行级别的问题，init需要等待进程运行结束后才继续。

11）initdefault：指定一个默认的运行级别，只有当init一开始被调用时才扫描这一项，如果rstate字段指定了多个运行级别，其中最大的数字是默认的运行级别，如果runlevel字段是空的，init认为字段是0123456，于是进入级别6，这样便陷入了一个循环，如果inittab文件中没有包含initdefault的记录项，则在系统启动时请求用户为它指定一个初始运行级别。

process，该字段中进程可以是任意的守候进程、可执行脚本或程序，后面可以带参数。

7、sysvinit包的其它工具。

pidof，是killall5的链接，获得一个正在运行的进程的id。运行时后面跟进程的名称作为参数。

last，显示最近登录的用户列表。在指定了用户名和终端名的情况下，last只显示符合这些参数的记录。终端的名字可以简写，因此 last 0 等同于 last tty0。每次系统重新启动时，虚用户 reboot 都会被记录到日志中。所以last reboot 会列出自日志文件创建以来的所有重新启动的日志记录。

mountpoint，检测一个目录是否是一个挂载点。

mesg，控制其他人对终端的访问。

参考：http://fanqiang.chinaunix.net/system/linux/2002-01-30/1023.shtml