Boot Process Tips

Donnie Barnes, djb@redhat.com

V1.0, January 25, 1996

1 Preface

"Red Hat Tips" are documentation meant to help Red Hat users with specific tasks. Some of these documents are for new users, some are for advanced users. Hopefully each document will also be of help for both new and advanced users. If you have contributions to make, please send them to djb@redhat.com. If you have changes that need to be made to individual Tips, send them to the author of that document.

2 Introduction

This Document is a brief description of the internals of the boot up process. It basically covers in detail how the machine boots using SysV Init and the differences between the original Init used in older Slackware releases.

Init is the program that gets run by the kernel at boot time. It is in charge of starting all the normal processes that need to run at boot time. These include the gettys that allow you to log in, NFS daemons, FTP daemons, and anything else you want to run when your machine boots.

3 SysV Init

System V init is fast becoming the standard in the Linux world to control the startup of software at boot time. This is because it is easier to use and more powerful and flexible than the traditional BSD init.

The init binary is located in `/sbin` and not `/etc`. This is important as one might try and upgrade a machine to System V init without re-installing and reformatting. The linux kernel looks in `/etc` for its init first, so you must make sure and delete your old init from there if any.

SYSV init also differs from BSD init in that the config files are in a subdirectory of `/etc` instead of residing directly in `/etc`. This directory is called `rc.d`. In there you will find `rc.sysinit` and the following directories:

- init.d
- rc0.d
- rc1.d
- rc2.d
- rc3.d

- rc4.d
- rc5.d
- rc6.d

`init.d` contains a bunch of scripts. Basically, you need one script for each service you may need to start at boot time or when entering another runlevel. Services include things like networking, nfs, sendmail, httpd, etc. Services do not include things like setserial that must only be run once and then exited. Things like that should go in `rc.local` or `rc.serial`.

`rc.local` should be in `/etc/rc.d` if you want one. Most systems include one even though it doesn't do much. You can also include an rc.serial in `/etc/rc.d` if you need to do serial port specific things at boot time.

The chain of events is as follows:

- the kernel looks in several places for `init` and runs the first one it finds
- `init` runs `/etc/rc.d/rc.sysinit`
- `rc.sysinit` does a bunch of necessary things and then runs `rc.serial` (if it exists)
- `init` runs all the scripts for the default runlevel.
- `init` runs `rc.local`

The default runlevel is decided in `/etc/inittab`. You should have a line close to the top like:

```
id:3:initdefault:
```

From this, you'd look in the second column and see that the default runlevel is 3, as should be the case for most systems. If you want to change it, you can edit `/etc/inittab` by hand and change the 3. Be very careful when you are messing with the inittab. If you do mess up, you can get in to fix it by rebooting and doing:

```
LILO boot:  linux single
```

This *should* allow you to boot into single user mode so you can fix it.

Now, how does it run all the right scripts? If you do an `ls -l` on `rc3.d`, you might see something like:

```
lrwxrwxrwx 1 root root 17  13:11 S10network -> ../init.d/network
lrwxrwxrwx 1 root root 16  13:11 S30syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 14  13:32 S40cron -> ../init.d/cron
lrwxrwxrwx 1 root root 14  13:11 S50inet -> ../init.d/inet
lrwxrwxrwx 1 root root 13  13:11 S60nfs -> ../init.d/nfs
lrwxrwxrwx 1 root root 15  13:11 S70nfsfs -> ../init.d/nfsfs
lrwxrwxrwx 1 root root 18  13:11 S75keytable -> ../init.d/keytable
lrwxrwxrwx 1 root root 23  13:11 S80sendmail -> ../init.d/sendmail.init
lrwxrwxrwx 1 root root 18  13:11 S90lpd -> ../init.d/lpd.init
lrwxrwxrwx 1 root root 11  13:11 S99local -> ../rc.local
```

What you'll notice is that there are no real "files" in the directory. Everything there is a link to one of the scripts in the init.d directory. The links also have an "S" and a number at the beginning. The "S" means to start this particular script and a "K" would mean to stop it. The number is just there for ordering purposes. Init will start all the services based on the order they appear. You can duplicate numbers, but it will only confuse you somewhat. You just need to use a two digit number only, along with an upper case "S" or "K" to start or stop the services you need to.

How does it start and stop services? Simple. Each of the scripts is written to accept an argument which can be "start" and "stop". You can execute those scripts by hand in fact with a command like:

```
/etc/rc.d/init.d/httpd.init stop
```

to stop the httpd server. Init just reads the name and if it has a "K", it calls the script with the "stop" argument. If it has an "S" it calls the script with a "start" argument.

Why all these runlevels?

Some people want an easy way to setup machines to be multi-purpose. I could have a "server" runlevel that just runs httpd, sendmail, networking, etc. Then I could have a "user" runlevel that runs xdm, networking, etc.

4 Copyright Notice