Topic-8

System Initialization and System Services

Introduction

Learning how Linux boots up is critical. When you have this information you can use it to alter the type of login screen you get as well as which programs start up.

The Linux Boot Sequence

You might remember when you installed Linux that the installation process prompted you for a list of partitions and the sizes of each in which your filesystems would be placed.

When allocating disk space for the partitions, the first sector, or data unit, for each partition is always reserved for programmable code used in booting. The very first sector of the hard disk is reserved for the same purpose and is called the master boot record (MBR).

When booting from a hard disk, the PC system BIOS loads and executes the boot loader code in the MBR. The MBR then needs to know which partitions on the disk have boot loader code specific to their operating systems in their boot sectors and then attempts to boot one of them.

Fedora Linux is supplied with the GRUB boot loader which is fairly sophisticated and therefore cannot entirely fit in the 512 bytes of the MBR. The GRUB MBR boot loader merely searches for a special boot partition and loads a second stage boot loader. This then reads the data in the `/boot/grub/grub.conf` configuration file, which lists all the available operating systems and their booting parameters. When this is complete, the second stage boot loader then displays the familiar Fedora branded splash screen that lists all the configured operating system kernels for your choice.

Note: In some operating systems, such as Debian / Ubuntu, the `/boot/grub/grub.conf` file may also be referred to by the name `/boot/grub/menu.lst`.

Sample grub.conf file

```
default=0
timeout=10
splashimage=(hd0,0)/grub/splash.xpm.gz
title Fedora Core (2.6.8-1.521)
        root (hd0,0)
        kernel /vmlinuz-2.6.8-1.521 ro root=LABEL=/
        initrd /initrd-2.6.8-1.521.img
title Windows 2000
      rootnoverify (hd0,1)
      chainloader +1
```

When Linux begins to boot with its kernel, it first runs the /sbin/init program, which does some system checks, such as verifying the integrity of the file systems, and starts vital programs needed for the operating system to function properly. It then inspects the /etc/inittab file to determine Linux's overall mode of operation or runlevel. A listing of valid runlevels can be seen in Table 7-1.

7-1 Linux Runlevels

Mode	Directory	Run Level Description
0	/etc/rc.d/rc0.d	Halt
1	/etc/rc.d/rc1.d	Single-user mode
2	/etc/rc.d/rc2.d	Not used (user-definable)
3	/etc/rc.d/rc3.d	Full multi-user mode (no GUI interface)
4	/etc/rc.d/rc4.d	Not used (user-definable)
5	/etc/rc.d/rc5.d	Full multiuser mode (with GUI interface)
6	/etc/rc.d/rc6.d	Reboot

Based on the selected runlevel, the init process then executes startup scripts located in subdirectories of the `/etc/rc.d` directory. Scripts used for runlevels 0 to 6 are located in subdirectories `/etc/rc.d/rc0.d` through `/etc/rc.d/rc6.d`, respectively.

Here is a directory listing of the scripts in the `/etc/rc.d/rc3.d` directory:

```
[root@bigboy tmp]# ls /etc/rc.d/rc3.d
...    ...    K75netfs      K96pcmcia    ...    ...
...    ...    K86nfslock    S05kudzu     ...    ...
...    ...    K87portmap    S09wlan      ...    ...
...    ...    K91isdn       S10network   ...    ...
...    ...    K92iptables   S12syslog    ...    ...
...    ...    K95firstboot  S17keytable  ...    ...
[root@bigboy tmp]#
```

As you can see, each filename in these directories either starts with an "S" which signifies the script should be run at startup, or a K, which means the script should be run when the system is shutting down. If a script isn't there, it won't be run.

Most Linux packages place their startup script in the `/etc/init.d` directory and place symbolic links (pointers) to this script in the appropriate subdirectory of `/etc/rc.d`. This makes file management a lot easier. The deletion of a link doesn't delete the file, which can then be used for another day.

The number that follows the K or S specifies the position in which the scripts should be run in ascending order. In our example, `kudzu` with a value 05 will be started before `wlan` with a value of 09. Fortunately you don't have to be a scripting/symbolic linking guru to make sure everything works right because Fedora comes with a nifty utility called `chkconfig` while Debian / Ubuntu uses the `update-rc.d` command to do it all for you. This is explained later.

Determining the Default Boot runlevel

The default boot runlevel is set in the file /etc/inittab with the initdefault variable. When set to 3, the

system boots up with the text interface on the VGA console; when set to 5, you get the GUI. Here is a snippet of the file (delete the initdefault line you don't need):

```
# Default runlevel. The runlevels used by RHS are:
# 0 - halt (Do NOT set initdefault to this)
# 1 - Single user mode
# 2 - Multiuser, without NFS (The same as 3, if you do not have networking)
# 3 - Full multiuser mode
# 4 - unused
# 5 - X11
# 6 - reboot (Do NOT set initdefault to this)
#
id:3:initdefault:                       # Console Text Mode
id:5:initdefault:                       # Console GUI Mode
```

Note the following:

- Most home users boot up with a Windows like GUI (runlevel 5)
- Most techies will tend to boot up with a plain text-based command-line-type interface (runlevel 3)
- Changing initdefault from 3 to 5, or vice-versa, has an effect upon your next reboot. See the following section on how to get a GUI login all the time until the next reboot.
- Of course, don't set the initdefault value to 6 or your system will constantly reboot. Setting it to 0 will never allow it to start!

Getting a GUI Console

Manual Method: You can start the X terminal GUI application each time you need it by running the startx command at the VGA console. Remember that when you log out you will get the regular text-based console again.

```
[root@bigboy tmp]# startx
```

Automatic Method: You can have Linux automatically start the X terminal GUI console for every login attempt until your next reboot by using the init command. You will need to edit your initdefault variable in your /etc/inittab file, as mentioned in the preceding section to keep this functionality even after you reboot.

```
[root@bigboy tmp]# init 5
```

When the CPU capacity or available memory on your server is low or you want to maximize all system resources, you might want to operate in text mode runlevel 3 most of the time, using the GUI only as necessary with the startx command.

Servers that double as personal workstations, or servers that might have to be operated for an extended period of time by relatively nontechnical staff, may need to be run at runlevel 5 all the time through the init 5 command. Remember you can make runlevel 5 permanent even after a reboot by editing the /etc/inittab file.

Get a Basic Text Terminal Without Exiting the GUI

There are a number of ways for you to get a command prompt when running a Linux GUI. This can be important if you need quick access to commands or you are not familiar with the GUI menu option layout.

Using a GUI Terminal Window

You can open a GUI-based window with a command prompt inside by doing the following:

- Click on the Fedora logo button in the bottom left hand corner of the screen.
- Click on Systems Tools and then Terminal

Using Virtual Consoles

By default, Linux runs six virtual console or TTY sessions running on the VGA console. These are defined by the mingetty statements in the /etc/inittab file. The X terminal GUI console creates its own virtual console using the first available TTY that is not controlled by mingetty. This makes the GUI run as number 7:

- You can step through each virtual console session by using the Ctl-Alt-F1 through F6 key sequence. You'll get a new login prompt for each attempt.
- You can get the GUI login with the sequence Ctl-Alt-F7 only in run level 5, or if the GUI is running after launching startx.

System Shutdown and Rebooting

It is usually not a good idea to immediately power off your system when you are finished using it. This can cause files that are being updated to become corrupted, or worse, you could corrupt the filesystem directory structure. Linux has a number of ways to gracefully shut down and reboot your system which will be outlined in this section.

Halt/Shut Down The System

The init command will allow you to change the current runlevel, and for a shutdown, that value is 0. Here is an example:

```
[root@bigboy tmp]# init 0
```

Fedora also has a shutdown command which can also be used to the same effect. It often prompts you as to whether you are sure you want to execute the command, which can be avoided with the -y switch. The -h switch forces the system to halt, and the first argument tells it how long to wait before starting the procedure, in this case 0 minutes. You can also specify shutting down at a specific time of the day; please refer to the man pages for details. Another advantage of the shutdown command is that it warns people that the shutdown is going to occur.

```
[root@bigboy tmp]# shutdown -hy 0
```

```
Broadcast message from root (pts/0) (Sat Nov  6 13:15:27 2004):

The system is going down for system halt NOW!
[root@bigboy tmp]#
```

Reboot The System

You can also use the init command to reboot the system immediately by entering runlevel 6.

```
[root@bigboy tmp]# init 6
```

The "reboot" command has the same effect, but it also sends a warning message to all users.

```
[root@bigboy tmp]# reboot

Broadcast message from root (pts/0) (Sat Nov  6 12:39:31 2004):

The system is going down for reboot NOW!
[root@bigboy tmp]#
```

More graceful reboots can be done with the shutdown command using the -r switch and specifying a delay, which in this case is 10 minutes.

```
[root@bigboy root]# shutdown -ry 10

Broadcast message from root (pts/0) (Sat Nov  6 13:26:39 2004):

The system is going DOWN for reboot in 10 minutes!

Broadcast message from root (pts/0) (Sat Nov  6 13:27:39 2004):

The system is going DOWN for reboot in 9 minutes!
...
...
...
Broadcast message from root (pts/0) (Sat Nov  6 13:36:39 2004):

The system is going down for reboot NOW!
```

Another reason is the recovery of your root password.

Switching to Single-user Mode

When the system is running normally, this can be done by using the init command to enter runlevel 1. It is best to do this from the console, because if you do it from a remote terminal session you'll be logged out.

```
[root@bigboy root]# init 1
...
...
bash-2.05b#
```

Unfortunately, this gives no prior warning to users, and the shutdown command doesn't have a single-

user mode option. This can be overcome by running the shutdown command with a delay in minutes as the only argument.

```
[root@bigboy tmp]# shutdown 1

Broadcast message from root (pts/0) (Sat Nov  6 13:44:59 2004):

The system is going DOWN to maintenance mode in 1 minute!

Broadcast message from root (pts/0) (Sat Nov  6 13:45:59 2004):

The system is going down to maintenance mode NOW!

...
...
bash-2.05b#
```

Entering Single-user Mode At The Grub Splash Screen

You can enter single user mode directly after turning on the power to your system. The steps to do this are listed below.

1. Power on your system. Wait for the "Grub loading" message to appear and, depending on your Linux distribution, get ready to hit either any key or the ESC key to enter the grub boot menu.

```
Grub loading, please wait ...
Press ESC to enter the menu
```

or

```
Grub loading, please wait ...
Press any key to enter the menu
```

2. You will then get grub's main menu which will display a list of available kernels. Use the arrow keys to scroll to your desired version of the kernel and then press e for "edit".

```
Fedora Core (2.6.18-1.2239.fc5smp)
Fedora Core (2.6.18-1.2200.fc5smp)
```

3. The kernel's boot menu will appear. Use the arrow keys to scroll to the "kernel" line and then press e for "edit".

```
root (hd0,0)
kernel /vmlinuz-2.6.18-1.2239.fc5smp ro root=LABEL=/
initrd /initrd-2.6.18-1.2239.fc5smp.img
```

4. A grub edit prompt will appear. Use the arrow keys to move to the end of the line and add the word "single" to the end, separated by a space. Change

```
grub edit> kernel /vmlinuz-2.6.18-1.2239.fc5smp ro root=LABEL=/
```

to

```
grub edit> kernel /vmlinuz-2.6.18-1.2239.fc5smp ro root=LABEL=/ single
```

5. Press enter to save your changes, and then b for "boot". 6. The system will continue to boot, but will go straight to the root # prompt without first asking for a username and password.

Reverting To Your Default runlevel From Single User Mode

The exit command forces the system to exit runlevel 1 and revert to the default runlevel for the system. You can also use the init command (for example init 3 and init 5) to alter this default behavior:

```
bash-2.05b# exit
INIT: Entering runlevel: 3
...
...
...
Fedora Core release 2 (Tettnang)
Kernel 2.6.8-1.521 on an i686
bigboy login:
```

Root Password Recovery

Sometimes you might forget the root password, or the previous systems administrator may move on to a new job without giving it to you. To do this, follow these steps:

1. Go to the VGA console and press Ctrl-Alt-Del. The system will then shut down in an orderly fashion.
2. Reboot the system and enter single-user mode.
3. Once at the command prompt, change your password. Single user mode assumes the person at the console is the systems administrator root, so you don't have to specify a root username.
4. Return to your default runlevel by using the exit command.

Starting and Stopping Daemons

A simple definition of a daemon is a programs that runs unattended even when nobody is logged into your system. Common examples of daemons are the `syslog` daemon which receives system error messages and writes them to a log file; the `apache` or `httpd` daemon that serves web pages to Internet web browsers and the `sendmail` daemon that places email it receives into your inbox.

The startup scripts I have been mentioning in the `/etc/init.d` directory govern the activation of daemons that were installed with some of your Linux packages. The commands to start and stop them are universal.

Starting a Daemon

If a startup script exists in the `/etc/init.d` directory, then its daemon can be started by specifying its filename followed by the keyword "start" as seen here:

```
root@u-bigboy:~# /etc/init.d/apache start
 * Starting apache 1.3 web server...
   ...done.
root@u-bigboy:~#
```

Stopping a Daemon

Daemons can be stopped by specifying its script filename followed by the keyword "stop":

```
root@u-bigboy:~# /etc/init.d/apache stop
```

```
 * Stopping apache 1.3 web server...
   ...done.
root@u-bigboy:~#
```

Restarting a Daemon

Daemons usually only read their configuration files when they are started, therefore if you edit the file, you have to restart the daemon for the new settings to become active. This can be done with the keyword "restart":

```
root@u-bigboy:~# /etc/init.d/apache restart
 * Restarting apache 1.3 web server...
   ...done.
root@u-bigboy:~#
```

Don’t worry about configuring your daemons. Later we'll be covering some commonly used daemons and will discuss them with ample examples.

The service command

Some operating systems such as Fedora and Redhat also come with the shortcut `service` command which allows you to control daemons with the "start", "stop" and "restart" keywords, but with less typing. Here are some quick, intuitive examples of doing this:

```
[root@bigboy ~]# service httpd start
[root@bigboy ~]# service httpd stop
[root@bigboy ~]# service httpd restart
```

The service command also has the "status" keyword which will provide a brief report on what the daemon is doing.

```
[root@bigboy ~]# service httpd status
httpd (pid 6135 6133 6132 6131 6130 6129 6128 6127 1561) is running...
[root@bigboy ~]#
```

Using chkconfig to Start Daemons at Each runlevel

As stated earlier, the chkconfig command can be used to adjust which applications start at each runlevel. You can use this command with the --list switch to get a full listing of packages listed in /etc/init.d and the runlevels at which they will be on or off:

```
[root@bigboy tmp]# chkconfig --list
keytable 0:off 1:on  2:on  3:on 4:on  5:on 6:off
atd      0:off 1:off 2:off 3:on 4:on  5:on 6:off
syslog   0:off 1:off 2:on  3:on 4:on  5:on 6:off
gpm      0:off 1:off 2:on  3:on 4:on  5:on 6:off
kudzu    0:off 1:off 2:off 3:on 4:on  5:on 6:off
wlan     0:off 1:off 2:on  3:on 4:on  5:on 6:off
sendmail 0:off 1:off 2:off 3:on 4:off 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
random   0:off 1:off 2:on  3:on 4:on  5:on 6:off
...
...
```

chkconfig Examples

You can use chkconfig to change runlevels for particular packages. Here we see sendmail will start with a regular startup at runlevel 3 or 5. Let's change it so that sendmail doesn't startup at boot.

Use Chkconfig to Get a Listing of sendmail's Current Startup Options

The chkconfig command can be used with grep to determine the run levels in which sendmail will run. Here we see it will run at levels 3 and 5.

```
[root@bigboy tmp]# chkconfig --list | grep mail
sendmail 0:off 1:off 2:off 3:on 4:off 5:on 6:off
[root@bigboy tmp]#
```

Switch Off sendmail Starting Up in Levels 3 and 5

The chkconfig command with the --level switch indicates that some action needs to be done at the runlevels entered as its values. The first argument in the command is the package you want to affect and the second defines whether you want it on or off. In this case we want sendmail not to be started when entering runlevels 3 and 5:

```
[root@bigboy tmp]# chkconfig --level 35 sendmail off
[root@bigboy tmp]#
```

By not specifying the runlevels with the --level switch, chckconfig will make the changes for runlevels 3 and 5 automatically:

```
[root@bigboy tmp]# chkconfig sendmail off
```

Because the intention is to permanently shutdown sendmail permanently, we might also have to stop it from running now.

```
[root@bigboy tmp]# service sendmail stop
Shutting down sendmail: [  OK  ]
Shutting down sm-client: [  OK  ]
[root@bigboy tmp]#
```

Double-check that sendmail Will Not Start Up

We can then use chkconfig to double-check our work.

```
[root@bigboy tmp]# chkconfig --list | grep mail
sendmail 0:off 1:off 2:off 3:off 4:off 5:off 6:off
[root@bigboy tmp]#
```

Turn On sendmail Again

To reactivate sendmail, we can use chkconfig once more, but with the on argument. Start sendmail again to get it running immediately, not just after the next reboot.

```
[root@bigboy tmp]# chkconfig sendmail on
[root@bigboy tmp]# chkconfig --list | grep mail
sendmail 0:off 1:off 2:off 3:on 4:off 5:on 6:off
[root@bigboy tmp]# service sendmail start
Starting sendmail: [  OK  ]
Starting sm-client: [  OK  ]
[root@bigboy tmp]#
```

Using chkconfig to Improve Security

A default Fedora installation automatically starts a number of daemons that you may not necessarily need for a Web server. This usually results in your system listening on a variety of unexpected TCP/IP ports that could be used as doors into your system by hackers.

The screen output of the netstat -an command below shows a typical case. Some ports are relatively easy to recognize. TCP ports 25 and 22 are for mail and SSH, respectively, but some others are less obvious. Should you use the chkconfig command and the scripts in the /etc/init.d directory to shut these down permanently?

```
[root@bigboy tmp]# netstat -an
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address
State
tcp    0    0 0.0.0.0:32768    0.0.0.0:*     LISTEN
tcp    0    0 127.0.0.1:32769  0.0.0.0:*     LISTEN
tcp    0    0 0.0.0.0:111      0.0.0.0:*     LISTEN
tcp    0    0 127.0.0.1:631    0.0.0.0:*     LISTEN
tcp    0    0 127.0.0.1:25     0.0.0.0:*     LISTEN
tcp    0    0 :::22            :::*          LISTEN
udp    0    0 0.0.0.0:32768    0.0.0.0:*
udp    0    0 0.0.0.0:930      0.0.0.0:*
udp    0    0 0.0.0.0:68       0.0.0.0:*
udp    0    0 0.0.0.0:111      0.0.0.0:*
udp    0    0 0.0.0.0:631      0.0.0.0:*
...
...
[root@bigboy tmp]#
```

For example, how do you know which startup script is responsible for TCP port 111? The answer is to use the lsof command which lists all open, or actively used, files and can be given additional options to extend its scope to include the TCP/IP protocol stack.

```
[root@ bigboy tmp]# lsof -i tcp:111
COMMAND  PID USER   FD   TYPE DEVICE SIZE NODE NAME
portmap 1165  rpc    4u  IPv4   2979       TCP *:sunrpc (LISTEN)
[root@ bigboy tmp #

[root@bigboy tmp]# lsof -i tcp:32769
COMMAND  PID USER   FD   TYPE DEVICE SIZE NODE NAME
xinetd  1522 root    5u  IPv4   2764       TCP probe-001:32769 (LISTEN)
```

```
[root@bigboy tmp]#

[root@bigboy root]# lsof -i udp:123
COMMAND  PID USER   FD   TYPE DEVICE SIZE NODE NAME
ntpd    1321  ntp    4u  IPv4   3390       UDP *:ntp
...
...
[root@bigboy root]#
```

In some cases it's tricky to determine the application based on the results of the lsof command. In the example below, we've discovered that TCP port 32768 is being used by rpc.statd, but there is no rpc.statd file in the /etc/init.d directory. The simple solution is to use the grep command to search all the files for the string rpc.statd to determine which one is responsible for its operation. We soon discover that the nfslock daemon uses it. If you don't need nfslock, then shut it down permanently.

```
[root@bigboy tmp]# lsof -i tcp:32768
COMMAND    PID    USER   FD   TYPE DEVICE SIZE NODE NAME
rpc.statd 1178 rpcuser    6u  IPv4   2400       TCP *:32768 (LISTEN)
[root@bigboy tmp]# ls /etc/init.d/rpc.statd
ls: /etc/init.d/rpc.statd: No such file or directory
[root@bigboy tmp]# grep -i statd /etc/init.d/*
/etc/init.d/nfslock:[ -x /sbin/rpc.statd ] || exit 0
...
...
[root@bigboy tmp]#
```

As a rule of thumb, applications listening only on the loopback interface (IP address 127.0.0.1) are usually the least susceptible to network attack and probably don't need to be stopped for network security reasons. Those listening on all interfaces, depicted as IP address 0.0.0.0, are naturally more vulnerable and their continued operation should be dependent on your server's needs. I usually shutdown nfs, nfslock, netfs, portmap, and cups printing as standard practice on Internet servers.

Remember to thoroughly research your options thoroughly before choosing to shut down an application. Use the Linux man pages, reference books and the Internet for information. Unpredictable results are always undesirable.

Shutting down applications is only a part of server security. Firewalls, physical access restrictions, password policies, and patch updates need to be considered. Full coverage of server and network security is beyond the scope of this book, but you should always have a security reference guide on hand to guide your final decisions.

Final Tips on chkconfig

Remember the following:

- In most cases, you want to modify runlevels 3 and 5 simultaneously and with the same values.
- Don't add/remove anything to other runlevels unless you absolutely know what you are doing. Don't experiment, unless in a test environment.
- chkconfig doesn't start the programs in the /etc/init.d directory, it just configures them to be started or ignored when the system boots up. The commands for starting and stopping the programs covered in this book are covered in each respective chapter.

Using sysv-rc-conf to Start Daemons at Each runlevel

With Debian / Ubuntu Linux, the `update-rc.d` command replaces `chkconfig` as the default package for modifying `/etc/init.d` script links. Unfortunately, the utility was written to facilitate link modification when packages are installed or removed, but is less friendly when you need to alter links for existing packages you want to keep.

Fortunately there is hope for the harried systems administrator in the form of the `sysv-rc-conf` package which uses an almost identical syntax to `chkconfig`, and has a GUI mode if you run it from the command line without any arguments. This section will show you some important tips in using `sysv-rc-conf`.

Installing sysv-rc-conf

The `sysv-rc-conf` package can be installed easily using `apt-get`. Here is an example.

```
root@u-bigboy:~# apt-get install sysv-rc-conf
```

Listing the runlevels for Daemons

This can be done with the `--list` option. In this example below we get a listing for only the apache daemon.

```
root@u-bigboy:~#  sysv-rc-conf  --list apache
apache       0:off      1:off   2:on    3:on    4:on    5:on    6:off
root@u-bigboy:~#
```

Here we get a listing for everything.

```
root@u-bigboy:~#  sysv-rc-conf  --list
acpi-support 0:off      1:off   2:on    3:on    4:on    5:on    6:off
acpid        0:off      1:off   2:on    3:on    4:on    5:on    6:off
alsa-utils   0:off      6:off
…
…
…
vbesave      2:on       3:on    4:on    5:on
x11-common   S:on
root@u-bigboy:~#
```

Setting the runlevels for Daemons

The `sysv-rc-conf` program has further similarities to the `chkconfig` syntax. Here we set the `apache` daemon to start automatically at levels 2 through 5.

```
root@u-bigboy:~# sysv-rc-conf  apache on
```

Similarly, we can set the `apache` daemon not to start at levels 2 through 5 with this command:

```
root@u-bigboy:~# sysv-rc-conf  apache off
```

Finally we can use set the `apache` daemon to start only at levels 3 and 5.

```
root@u-bigboy:~# sysv-rc-conf  --level 35 apache on
```