Troubleshooting the GRUB boot loader:

I am interested in the boot loader from the perspective of what to do if

the boot loader fails or what ways you might want to interrupt the boot loader to change

the behavior of the boot process.

**Could not locate active partition**—When a boot loader is installed on a storage

medium, the partition is usually marked as bootable. If you see this message, it

means that no bootable partition was found. If you feel sure the boot loader is

on the disk, try using the fdisk command (probably from rescue media) to make

the partition bootable and try again.

**Text-based GRUB prompt appears**—It is possible for the BIOS to start GRUB and go

straight to a GRUB prompt, with no operating system selections available.

One workaround to this problem, assuming grub.conf is on the fi rst partition of

the fi rst disk, is to list the contents of this file and enter the root, kernel, and

initrd lines manually. To list the file, type **cat (hd0,0)/grub/grub.conf**.

If that doesn’t work, try hd0,1 to access the next partition on that disk (and so

on) or hd1,0 to try the fi rst partition of the next disk (and so on). When you fi nd

the lines representing the grub.conf file, manually type the root, kernel, and

initrd lines for the entry you want (replacing the location of the hard drive you

found on the root line). Then type **boot**. The system should start up, and you can

go and manually fi x your boot loader files. See Chapter 9 for more information on

the GRUB boot loader.

**To select a different kernel**—When RHEL installs a new kernel via yum, it always

keeps at least one older kernel around. If the new kernel fails, you can always boot

the previous, presumably working, older kernel.

**To select a different kernel**—When RHEL installs a new kernel via yum, it always

keeps at least one older kernel around. If the new kernel fails, you can always boot

the previous, presumably working, older kernel.

Starting the kernel:

If you want to

watch messages detailing the boot process scroll by, press the Esc key.

At this point, the kernel tries to load the drivers and modules needed to use the hardware

on the computer. The main things to look for at this point (although they may scroll by

quickly) are hardware failures that may prevent some feature from working properly.

Although much more rare than it used to be, there may be no driver available for a piece of

hardware, or the wrong driver may get loaded and cause errors.

In addition to scrolling past on the screen, messages produced when the kernel boots are

copied to the *kernel ring buffer*. As its name implies, the kernel ring buffer stores kernel

messages in a buffer, throwing out older messages after that buffer is full. After the

computer boots up completely, you can log into the system and type the following command

to capture these kernel messages in a file (then view them with the less command):

In Linux systems that support systemd, kernel messages are stored in the systemd

journal. So instead of using the dmesg command, you can run journalctl to see kernel

messages from boot time to the present. For example, here are kernel messages output from

a RHEL 7 system:

Troubleshooting the initialization system

The fi rst process to run on a system where the kernel has just started depends on the

initialization facility that system is using. For System V init, the fi rst process to run is

the init process. For systemd, the fi rst process is systemd. Depending on which you

see running on your system (type ps -ef | head to check), follow either the System V

or systemd descriptions below. RHEL 6, which contains a hybrid of Upstart and System V

init, is used in the example of System V initialization.

**Troubleshooting systemd initialization**

The latest versions of Fedora, RHEL, and soon Ubuntu use systemd instead of System V

init as their initialization system. When the systemd daemon (/usr/lib/systemd/

systemd) is started after the kernel starts up, it sets in motion all the other services that

are set to start up. In particular, it keys off the contents of the /etc/systemd/ system/

default.target file. For example:

# **cat /etc/systemd/system/default.target**

...

[Unit]

Description=Graphical Interface

Documentation=man:systemd.special(7)

Requires=multi-user.target

After=multi-user.target

Conflicts=rescue.target

Wants=display-manager.service

AllowIsolate=yes

[Install]

Alias=default.target

The default.target file is actually a symbolic link to a file in the /lib/systemd/system

directory. For a server, it may be linked to the multi-user.target file; for a desktop, it

is linked to the graphical.target file (as is shown here).

Unlike with the System V init facility, which just runs service scripts in alphanumeric

order, the systemd service needs to work backward from the default.target to determine

which services and other targets are run. In this example, default.target is a

symbolic link to the graphical.target file. When you list the contents of that file, you

can see the following:

■ The multi-user.target is required to start fi rst.

■ The display-manager.service is started after that.

By continuing to discover what those two units require, you can fi nd what else is required.

For example, multi-user.target requires the basic.target (which starts a variety

of basic services) and display-manager.service (which starts up the display manager,

gdm) to launch a graphical login screen.

To see services the multi-user.target starts, list contents of the /etc/systemd/

system/multi-user.target.wants directory. For example:

# **ls /etc/systemd/system/multi-user.target.wants/**

abrt-ccpp.service chronyd.service nfs.target

abrtd.service crond.service nmb.service

abrt-oops.service cups.path remote-fs.target

abrt-vmcore.service httpd.service rngd.service

abrt-xorg.service irqbalance.service smb.service

atd.service mcelog.service sshd.service

auditd.service mdmonitor.service vmtoolsd.service

autofs.service ModemManager.service vsftpd.service

avahi-daemon.service NetworkManager.service

These files are symbolic links to files that defi ne what starts for each of those services. On

your system, these may include remote shell (sshd), printing (cups), auditing (auditd),

networking (NetworkManager), and others. Those links were added to that directory

either when the package for a service is installed or when the service is enabled from a

systemctl enable command.

Keep in mind that, unlike System V init, systemd can start, stop, and otherwise manage

unit files that represent more than just services. It can manage devices, automounts, paths,

sockets, and other things. After systemd has started everything, you can log into the

system to investigate and troubleshoot any potential problems.

After you log in, running the systemctl command lets you see every unit file that

systemd tried to start up. Here is an example:

# **systemctl**

UNIT LOAD ACTIVE SUB

DESCRIPTION

proc-sys-fs-binfmt\_misc.automount loaded active waiting

Arbitrary Executable File Formats File System

sys-devices-pc...:00:1b.0-sound-card0.device loaded active plugged

631xESB/632xESB High Definition Audio Control

sys-devices-pc...:00:1d.2-usb4-4\x2d2.device loaded active plugged

DeskJet 5550

...

-.mount loaded active mounted

/

boot.mount loaded active mounted

/boot

...

autofs.service loaded active running

Automounts filesystems on demand

cups.service loaded active running

CUPS Printing Service

httpd.service loaded failed failed

The Apache HTTP Server

From the systemctl output, you can see whether any unit file failed. In this case, you can

see that the httpd.service (your Web server) failed to start. To further investigate, you

can run journalctl -u for that service to see whether any error messages were reported:

# **journalctl -u httpd.service**

...

Sep 07 18:40:52 host systemd[1]: Starting The Apache HTTP Server...

Sep 07 18:40:53 host httpd[16365]: httpd: Syntax error on line 361 of

/etc/httpd/conf/httpd.conf: Expected </Director> but saw

</Directory>

Sep 07 18:40:53 host systemd[1]: httpd.service:

main process exited, code=exited, status=1/FAILURE

Sep 07 18:40:53 host systemd[1]: Failed to start The Apache HTTP

Server.

Sep 07 18:40:53 host systemd[1]: Unit httpd.service entered failed

state.

From the output, you can see that there was a mismatch of the directives in the httpd.

conf file (I had Director instead of Directory). After that was corrected, I could start the

service (systemctl start httpd). If more unit files appear as failed, you can run the

journalctl -u command again, using those unit filenames as arguments.

The next section describes how to troubleshoot issues that can arise with your software

packages.