Global Knowledge®

Expert Reference Series of White Papers

Troubleshooting Made Easy In Linux

Troubleshooting Made Easy In Linux

David Egan, RHCX

Introduction

Part 1: Troubleshooting Overview

Part 2: Getting Started

Part 3: Getting Services Started

Part 1: Troubleshooting Overview

Every operating system portrays itself as powerful, able to leap tall buildings and applications in single strides, able to weather the storms of prolonged heavy usage, resilient to weathering, and so much more!

One of the necessary evils, albeit less often spoken of, in case we jinx something, is the topic of troubleshooting. What could possibly go wrong? Doesn't everyone believe in fair play, honorable intentions, and being openly honest and straightforward in all things virtual?

Where does one start when the operating system does not start, the application does not load or run correctly, the network hangs or a network service does not listen or perform?

A common practice is to reload the machine, either from the installation media and then patch it, or from some standardized image. Either way, the problem has only been gotten around, not solved.

What if you want to solve the problem itself? What if you need to solve the problem to move ahead?

Linux is noted for being rock solid, stable, and easy to manage but not as user friendly ... 'Where do I click?' is usually not the answer.

There is an ancient, and maybe black, dark, or mysterious 'art' form known as troubleshooting. The idea of troubleshooting problems when they occur, whether with the system bootup or setup, networking or network services, or applications in memory, is a daunting task for anyone. The clues are probably there somewhere, depending on experience. Finding the clues and working backwards from them is the intriguing part to the sleuth in anyone interested!

Starting Point

Troubleshooting starts with knowing what is normal for your system. You will just have to get dirty here. Open the hood and see what all the little moving gadgets actually do, how they work together—or not. What makes this baby tick?

Not every system will be identical, but most likely, your servers use one or two common operating systems, Linux being one of them. You need to know something, in case there are problems, about the startup, the network software running in memory (or not), the general management services creating logs and other output while keeping the clutter to a minimum, application management and how to 'watch' programs while they are running.

Trend of Compression

The trend is to do more with less ... run more programs with less equipment. Large box computers dedicated as a server were replaced by 'blade' servers. These blade servers reduced the size of the server box and allowed for more hosts in less space.

Additional applications were added to machines with low usage services. More applications mean more potential issues. Alternatively, many times, due to application collisions, separate servers were created to do just one service.

What if you could run more applications on one more powerful host? What do you do when something goes wrong? You cannot always back out and rebuild or re-image; you have to find the problem and fix it as soon as possible.

RHEL Troubleshooting Options Are Vast

What does Red Hat Enterprise Linux, RHEL, provide for troubleshooting?

There are many ways to troubleshoot in Linux depending on where the problem appears to be coming from: from the bootup to the services, application, and kernel.

This white paper will explore the various options that make Linux a 'fun' place to be when it comes to troubleshooting. The 'fun' is a relevant term. Having to work over the weekend or die trying is never really fun, but the cool array of tools and options for troubleshooting make it more 'fun' than some of the competition.

Just think, you probably will not have to reboot ... unless it is a bootup problem.

Usually No Reboot Necessary

In Linux, other than a few kernel issues, you are not forced to reboot to re-initiate. You do not have to reboot to replace old software with new, unless if it is specifically a kernel. You can access the source code if necessary. You can use a debugger or system tracing utility. You can dump memory in the extreme case.

Still Do Backups

Naturally, you still need to do backups, just in case hardware fails. But if the error seems to be program related, you want to have a starting point on where to begin. Fortunately, most configuration files are text.

General Troubleshooting Categories

Here is a synopsis of general troubleshooting categories or areas of concern, the concepts and techniques used.

Bootup

- Includes a Powerful Bootup service known as GRUB
- Bootup provides a command line interface before the Linux kernel is started

Mostly TEXT Configuration

- Configuration is almost always in a text file: simple to work on from anywhere
- Generally text configuration files provided in **/etc, /etc/sysconfig** for most services
- Simplified configuration scripts and files to regulate software started at bootup

Verification and Management

- Package management includes verification of all files
- Change Management with SCM like aide

Log Space Maintenance

- Log management and log reviews already setup
- Remote logging easy to configure

Networking and Services

- Networking uses simple, comment oriented variables, in text files
- Network services, Port scanning and network sniffing

Supporting Supported Packages

- Digital Signatures on all supported packages
- Configuring YUM and RPM for digital signatures
- All packages provided in source code for review and repairs
- Rebuilding of the kernel is always an option but discouraged

Debugging a program

- Tracing the system calls of any application
- Watching all applications combined kernel module access
- Kernel memory dumps

There is almost always a solution to every problem. How long it takes to resolve it or what must be done to ensure it does not occur again remains elusive forever. There have been, and probably will continue to be, many times when I have tackled a problem late in a day only to realize it is now just about dawn and it is still not fixed yet ... just a little more time, a few more tweaks ... I have to FIND IT!

Part 2: Getting Started

It all starts with getting the system started. The motherboard of an **Intel/AMD/Cyrex/. chipset** reads in the first sector of the first disk, called the Master Boot Record or MBR, after doing its own Pre-Operating System Tests, POST. This MBR program is just a small stepping stone to starting the real boot loader program that has to run in the DOS environment of a simple 8086 machine. This second-stage loader then loads the kernel and an initial ram disk file into memory, which turns over control to the loader kernel.

The kernel, once it is 'started', starts to initiate hardware by probing the hardware and starting the internal module. If no internal module is found within the kernel, it first looks in the initial ram disk-based file system for these modules to load as it is unable to see the actual disks until the kernel has loaded and initialized the appropriate driver for the disk drives. After it has the disk drive modules loaded, it can do a file system check for integrity, then it can mount the local disks as the real file system and drop the initial ram disk from memory. Next step is to load the init program. The init program reads its configuration file **/etc/inittab**, which defines the sequence of scripts needed to initiate all network services and start the various login processes for access to the system. Bootup usually takes under a minute to complete, if nothing goes wrong!

GRUB Starts It All Off

A rarely seen issue is with bootup. As of the last few releases, Red Hat Enterprise Linux, RHEL, has been using GRUB, the Grand Unified Bootloader, program-loading software for starting up the kernel.

The GRUB loader has a simple first stage loader stored in the Master Boot Record, MBR. The MBR is generally the first sector of the disk marked as the boot device.

This program loads its one point five and second stages into memory and presents a minimal management menu. You have 5 seconds in which to press any key to interrupt the default action of just booting the system using the default, defined kernel and initial RAM Disk file.

```
[root] # ls /boot/grub
device.map grub.conf minix_stage1_5 stage2
e2fs_stage1_5 iso9660_stage1_5 reiserfs_stage1_5 ufs2_stage1_5
fat_stage1_5 jfs_stage1_5 splash.xpm.gz vstafs_stage1_5
ffs_stage1_5 menu.lst stage1 xfs_stage1_5
[root] #
[root]# cat /etc/grub.conf
# grub.conf generated by anaconda
#
..
```

```
default=0
timeout=5
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu
title Default Red Hat Enterprise Linux Server (2.6.18-53.el5)
     root (hd0,0)
     kernel /vmlinuz-2.6.18-53.el5 ro root=/dev/VolGroup00/
LogVol00 rhgb quiet
     initrd /initrd-2.6.18-53.el5.img
[root]#
```

Bootup

Not seen in any other non-UNIX-like OS machine or architecture by the author, the first amazing tool is this GRUB loader program. GRUB includes a powerful, text-based, command line shell. This is not for the weak of heart.

You must exert your right to get to the command line interface supplied by GRUB by interrupting the normal bootup and then pressing a 'c' to get to the command line. This GRUB command line is available before the Linux kernel is started.

The GRUB command line has many options, including help, which lists the commands available.

```
GNU GRUB version 0.97 (640K lower / 3072K upper memory)
[ Minimal BASH-like line editing is supported. For the first word,
TAB
     lists possible command completions. Anywhere else
TAB lists the possible completions of a device/filename.]
grub> help
... [some entries not shown]
```

blocklist FILE	boot
cat FILE	chainloader [--force] FILE
clear	color NORMAL [HIGHLIGHT]
configfile FILE	device DRIVE DEVICE
displayapm	displaymem
find FILENAME	geometry DRIVE [CYLINDER HEAD SECTOR [
halt [--no-apm]	help [--all] [PATTERN ...]
hide PARTITION	initrd FILE [ARG ...]
kernel [--no-mem-option] [--type=TYPE]	makeactive

map TO_DRIVE FROM_DRIVE	md5crypt
module FILE [ARG ...]	modulenounzip FILE [ARG ...]
pager [FLAG]	partnew PART TYPE START LEN
parttype PART TYPE	quit
reboot	root [DEVICE [HDBIAS]]
rootnoverify [DEVICE [HD-BIAS]]	serial [--unit=UNIT] [--port=PORT] [--
setkey [TO_KEY FROM_KEY]	setup [--prefix=DIR] [--stage2=STAGE2_
terminal [--dumb] [--no-echo] [--no-ed	[--no-ed terminfo [--name=NAME --cursor-address
testvbe MODE	unhide PARTITION
uppermem KBYTES	vbeprobe [MODE]

Imagine, before you have booted your system, you can discover important system information, discover file names, DISPLAY text file content, test file loading, discover memory information, and even create a partition!

The 'geometry' command shows the **Cylinder/Heads/Sectors-per-platter** values, usually LBA-oriented.

```
grub> geometry (hd0,0)
geometry (hd0,0)
drive 0x80: C/H/S = 38913/255/63, The number of sectors =
625142448, /dev/sda
     Partition num: 0, Filesystem type unknown, partition type 0x7
     Partition num: 1, Filesystem type is ext2fs, partition type
0x83
grub>
grub> help partnew
help partnew
partnew: partnew PART TYPE START LEN
     Create a primary partition at the starting address START with
     the length LEN, with the type TYPE. START and LEN are in sec-
     tor units.
grub>
```

You can display some memory information with the 'displaymem' command.

```
grub> help displaymem
displaymem: displaymem
     Display what GRUB thinks the system address space map of the
```

```
     machine is, including all regions of physical RAM installed
grub> displaymem
     EISA Memory BIOS Interface is present vbeprobe [MODE]
     Address Map BIOS Interface is present
     Lower memory: 640K, Upper memory (to first chipset hole):
3072K
      [Address Range Descriptor entries immediately follow (values
are 64-bit)]
      Usable RAM: Base Address: 0x0 X 4GB + 0x0,
          Length: 0x0 X 4GB + 0xa0000 bytes
Reserved: Base Address: 0x0 X 4GB + 0xa0000,
          Length: 0x0 X 4GB + 0x60000 bytes
Usable RAM: Base Address: 0x0 X 4GB + 0x100000,
          Length: 0x0 X 4GB + 0x300000 bytes
grub>
```

The 'setup' command allows you to rebuild the MBR, not something you would need at this point.

The 'root' command expects a device name. This command sets the default device for all further file requests. You could precede every file request with the GRUB-style hardware identification or set it once with 'root (hd0,0)'. The 'hd0' is the first discovered disk controller. The second zero refers to the partition number. In both cases, these values are zero ordinal, they start at zero for the first controller and a zero for the first partition.

If your first disk partition would normally be mounted as **/boot**, then either of the following commands will print to standard out the grub.conf file in the grub subdirectory.

```
grub> cat (hd0,0)/grub/grub.conf
```

or

```
grub> root (hd0,0)
grub> cat /grub/grub.conf
```

Fortunately, most configuration files are maintained in a text format. This makes it easy to check them before you boot up with GRUB.

GRUB provides an amazing array of helpful options. You can check RAM, display files, rummage around. What more can you ask before you start your system?

Getting To Know What Is 'Normal'

It is important to understand what is the 'normal' mechanism used to start your system before trouble happens. By understanding what are the related programs and configuration files, you can follow their respective progress and possibly eliminate them from the trouble issue.

Part 3: Getting Services Started

After the kernel is in memory and in control, it loads the init program. The init program initiates all configured network services and starts the various login processes for access to the system. The services installed on a system are managed by one or more configuration files and a central management script usually.

Mostly TEXT Configuration

Configuration is almost always in a text file: simple to work on from anywhere and from any remote machine, if networking is enabled. Test configuration can be displayed at the GRUB command line or modified in single user mode access.

Generally, most service and network settings are in text configuration files and are usually provided in **/etc** in **SERVICENAME.conf** or **SERVICEdir/SERVICENAME.conf**. For most common network-oriented services, there is also a SERVICENAME file in **/etc/sysconfig** for how the service runs.

Examples of configuration files for Services:

DHCP has	/etc/dhcpd.conf,	/etc/sysconfig/dhcpd
NFS has	/etc/exports,	/etc/sysconfig/nfs
NIS has	/etc/{yp.conf,ypserv.conf},	/etc/sysconfig/yppasswd
Syslog	/etc/syslog.conf,	/etc/sysconfig/syslog
DNS has	/etc/named.conf,	/var/named/*zone

(but all DNS files are chrooted to /var/named/chroot by default)

Samba has	/etc/samba/smb.conf,	/etc/sysconfig/smb
Sendmail	/etc/mail/sendmail.cf,	/etc/sysconfig/sendmail
Squid	/etc/squid/squid.conf	/etc/sysconfig/squid

and a few oddities

FTP	/etc/vsftpd/vsftpd.conf	
VNC		/etc/sysconfig/vncservers

There can be many services installed on a Linux system, the above is just a short list of some common services.

For most services installed, there is a control script in one directory that has two reference names to it, **/etc/init.d** and **/etc/rc.d/init.d**. The two directories allow for historical UNIX compatibility. The service name can be accessed as **/etc/rc.d/init.d/SERVICENAME** or **/etc/init.d/SERVICENAME**

[root]# ls /etc/init.d/			
acpid	gpm	named	saslauthd
anacron	haldaemon	netconsole	sendmail
apmd	halt	netfs	setroubleshoot
apmd	halt	netfs	setroubleshoot
atd	hidd	netplugd	single
auditd	hplip	network	smartd
autofs	httpd	NetworkManager	smb
avahi-daemon	ip6tables	NetworkManagerDis-patcher	spamassassin
avahi-dnsconfd	ipmi	nfs	squid
bluetooth	iptables	nfslock	sshd
conman	irda	nscd	syslog
cpuspeed	irqbalance	ntpd	tux
crond	kdump	pand	vncserver
cups	killall	pcscd	vsftpd
cups-config-daemon	krb524	portmap	winbind
dc_client	kudzu	psacct	wpa_supplicant
dc_server	ldap	rdisc	xfs
dhcdbd	lvm2-monitor	readahead_early	xinetd
dhcpd	mcstrans	readahead_later	ypbind
dpcrelay	mdonitor	restorecond	yppasswdd
dovecot	mdmpd	rhnsd	ypserv
dund	messagebus	rpcgssd	pyxfrd
firstboot	microcode_ctl	rcpidmapd	yum-updated
functions	multipathd	rpcsvcgssd	
[root]#			

Some of the names of these scripts may not be obvious as to the related 'service' they represent unless you are intimately familiar with all the various things going on within your system. And this is a good place to start. In many cases, you can look at the main page for the service name listed above. Alternatively, you can find out which package supplied the script above and then query the package for a description of what the package does (after bootup has completed).

```
[root]# rpm -qf /etc/init.d/readahead_early
readahead-1.3-7.el5
[root]# rpm -qi readahead
Name : readahead  Relocations: (not relocatable)
```

```
Version : 1.3      Vendor: Red Hat, Inc.
Release : 7.el5    Build Date: Sun 04 Feb 2007 11:48:34 PM PST
Install Date: Sun 11 Jan 2009 09:57:49 PM PST   Build Host: hs20-
bc2-2.build.redhat.com
Group : System Environment/Base Source    RPM: readahead-1.3-7.
el5.src.rpm
Size : 261987      License: GPL/OSL
Signature : DSA/SHA1, Thu 08 Feb 2007 07:57:19 AM PST, Key ID
5326810137017186
Packager : Red Hat, Inc. http://bugzilla.redhat.com/bugzilla
Summary : Read a preset list of files into memory.
Description :
readahead reads the contents of a list of files into memory,
which causes them to be read from cache when they are actually
needed. Its goal is to speed up the boot process.
[root]# man readahead
READAHEAD(2) Linux Programmer's Manual READAHEAD(2)
NAME
readahead - perform file readahead into page cache
SYNOPSIS
#include
ssize_t readahead(int fd, off64_t *offset, size_t count);
DESCRIPTION
readahead() populates the page cache with data from a file so that
subsequent reads from that file will not block on disk I/O. The fd
argument is a file descriptor identifying the file which is to be
read.
..
```

Configuring Services At Bootup

For many first-time installers looking at the system after an installation, they will find a large number of services that have been installed but they are not necessarily running. For security purposes, the Red Hat installation leans toward locked down security rather than open door policy for services. Any service that is likely to be attacked or hijacked is either turned off or crippled after installation. This then requires that the superuser must properly configure these insecure services before they are used.

You can list the services by run level status using the **chkconfig --list** command.

```
[root]# chkconfig --list | head -5
NetworkManagerDispatcher       0:off 1:off 2:off 3:off 4:off 5:off 6:off
acpid        0:off 1:off 2:off 3:on 4:on 5:on 6:off
anacron      0:off 1:off 2:on 3:on 4:on 5:on 6:off
```

```
apmd            0:off 1:off 2:on 3:on 4:on 5:on 6:off
[root]#
```

From the list above, the first two named services, and they are probably related, are off for all run levels. The last three services are all on in the user-related run levels of 3 and 5. Although run level 2 and 4 exist, they are not standard.

Run Levels

One might think run level might be like a speed but, in fact, it refers to what services are started. The higher the number does typically imply more services.

Single User and Multi-User

In the early days of UNIX, the system used one long script to initiate network services in the BSD distributions. This script would start only required network services. This was referred to as 'multi-user mode.' All users could log in, and all networking services were configured.

They added in a pre multi-user 'phase' whereby the startup only went as far as getting the kernel into memory and all its helper modules but did not start any network services. This was known as Single User mode. Whoever initiated single user mode, and this could only be done at the console of the machine, was logged in as the superuser (root) without need of a password. The machine only allowed the one login, so they called it 'single user.' The Microsoft Windows world has duplicated this concept, they just gave it different names: safe mode (single user mode), safe mode with networking (run level 2 was sometimes configured this way), and normal mode (run level 3).

Most Linux distributions follow one other slight variation of this pattern. Run level 3 starts all configured network services EXCEPT X Window system. This is perfect for a back office server that simply provides networking services. There is no need to have the X Window system sitting in memory if it is not needed or used. Instead, most Linux distributions only turn on X when run level 5 is entered. This is a simple way to differentiate a user desktop machine, run level 5, and a backoffice server, run level 3 which provides a text interface for login only. The Microsoft Windows world did not have this concept in any previous version, but that has also changed. It is now possible to install the latest Microsoft Operating system such that a text interface is presented, no GUI appears. Go figure!

Get To Know What Is 'Normal'

It is important to understand what is the 'normal' state of your system before trouble happens. This makes looking for an error easier, you do not have to figure out as many services and files if you already know you can eliminate them from the trouble.

Summary

Simple text-based configuration scripts and files regulate software started at bootup. Service names vary as do the script names used to manage these services. Historical names have been maintained in most cases and may

not be obvious to new administrators of Linux. Getting to know what is normally started, which scripts and configuration files manage these services, is essential to being able to troubleshoot any system.

Learn More

Learn more about how you can improve productivity, enhance efficiency, and sharpen your competitive edge. Check out the following Global Knowledge courses:

RH033 Red Hat Linux Essentials

RH133 Red Hat Linux System Administration & Red Hat Certified Technician (RHCT) Lab Exam

RH142 Red Hat Linux Troubleshooting

RHD143 Red Hat Linux Programming Essentials

RH253 Red Hat Linux Networking and Security Administration

RH401 Red Hat Enterprise Deployment, Virtualization, and Systems Management

RH423 Red Hat Enterprise Directory Services and Authentication

RH436 Red Hat Enterprise Clustering and Storage Management

RH442 Red Hat Enterprise System Monitoring and Performance Tuning

RHS429 SELinux Policy Administration

For more information or to register, visit **www.globalknowledge.com** or call **1-800-COURSES** to speak with a sales representative.

Our courses and enhanced, hands-on labs offer practical skills and tips that you can immediately put to use. Our expert instructors draw upon their experiences to help you understand key concepts and how to apply them to your specific work situation. Choose from our more than 700 courses, delivered through Classrooms, e-Learning, and On-site sessions, to meet your IT and management training needs.

About the Author

David Egan, RCE, MCSE, PMP provides consulting services in Linux, UNIX, NT, Project Management, Systems Configuration, Security, and Networking.