Linux Administration I: System Initialization

This material is based on work supported by the National Science Foundation under Grant No. 0802551

National Science Foundation
WHERE DISCOVERIES BEGIN

Lesson Overview

A Linux administrator frequently encounters computer systems that have boot and initialization related problems. Savvy technicians will need to be familiar with the basic boot and initialization process in order to efficiently diagnose and repair these problems.

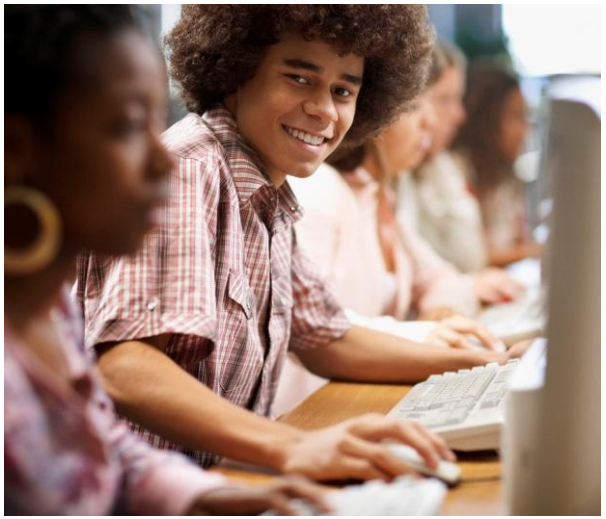
In this lesson, you will explore various topics about the boot process of Linux systems including basic fault isolation principles and repair techniques. It is important for you to understand these concepts so that you will become familiar with the inner workings of system initialization and be well-equipped to diagnose, troubleshoot, and fix related problems.

Select **PLAY** below to view an introductory video.

View Video VideoLesson5Introduction(C2L5S2).mp4

Objective

Given a computer experiencing boot problems, a student will be able to diagnose the cause, and recommend the appropriate fix that will allow the system to become functional.



Outline

In this lesson, you will explore:

- Boot terms
- Boot Process
- Boot Troubleshooting

Select **PLAY** below to view an introductory video.

View Video VideoLesson5TopicOutline(C2L5S13).mp4

BIOS

To learn system initialization, there are some essential concepts you must understand including:

- **BIOS**
  - (Basic Input Output System) The BIOS contains the startup diagnostic software (POST) and other instructions that allow hardware devices in a computer system to function properly.
  - The BIOS is the basic instruction set that allows your computer components such as hard drive, computer chips, disk devices, keyboard, and mouse to work together.

- **POST**
  - Power On Self Test – Initiated from the BIOS as part of the boot process, the Power ON Self Test checks the memory, hardware, power supply, and graphics card to make sure they are working correctly. If the hardware checks are good, the BIOS takes over and loads the boot loader.

Select **PLAY** below to view a video on the BIOS.

View Video VideoLesson5BIOS (C2L5S14).mp4

Supplemental Reading:

- How BIOS works
- BIOS

Boot Concepts

- **Boot Sector**
  - A boot sector is the portion of a hard disk or floppy disk that has the code stored on it to boot special programs and to reference other key features to keep the disk working. There are many types of boot sectors, but there are two main ones: the master boot record and the volume boot record.
- **Master Boot Record (MBR)**
  - A master boot record (MBR), or partition sector, is the 512-byte boot sector that is the first sector of a partitioned data storage device such as a hard disk. The MBR contains the instructions to either load the operating system or dictate where/how to find additional boot parameters to select from a variety of boot options. The MBR looks for information on where the different partitions may be located and transfers control of the boot process to that specific partition in order to load an operating system of choice.

Supplemental Reading:

- Boot sector
- Partition boot sector
- Master boot record
- Necessity of MBR
- Repair corrupt MBR

Boot Programs

In Linux, small programs assist with the boot process:

- **GRUB**
  - (Grand Unified Boot loader) - An all purpose, easy to use boot loader. Contains the instructions for loading an operating system from a target device such as hard drive, CD-Rom, or floppy disk.
  - Briefly, a boot loader is the first software program that runs when a computer starts. It is responsible for loading and transferring control to the operating system kernel software (such as Linux). The kernel, in turn, initializes the rest of the operating system.
- **LILO**
  - Linux Loader – One of the oldest boot loaders still used today. It is normally included in default Linux installations, especially ones with multiple operating systems or additional load parameters.

Select **PLAY** below to view a video on boot loaders.

View Video VideoLesson5BootLoaders (C2L5S16).mp4

Supplemental Reading:

- GNU GRUB
- GRUB 2 Tutorial
- Booting operating systems
- Configuring LILO
- Boot loaders made simple
- LILO error codes

Bootstrap

- **Bootstrap**
  - A small program or script on an EPROM, ROM, hard drive/disk that loads another program such as an operating system
- **Chainloading**
  - Chainloading – The process of allowing device one to ask the boot loader of device two to use its boot loader and boot process. Chainloading is how Grub loads a Windows OS.

Select **PLAY** below to view a video on bootstrap.

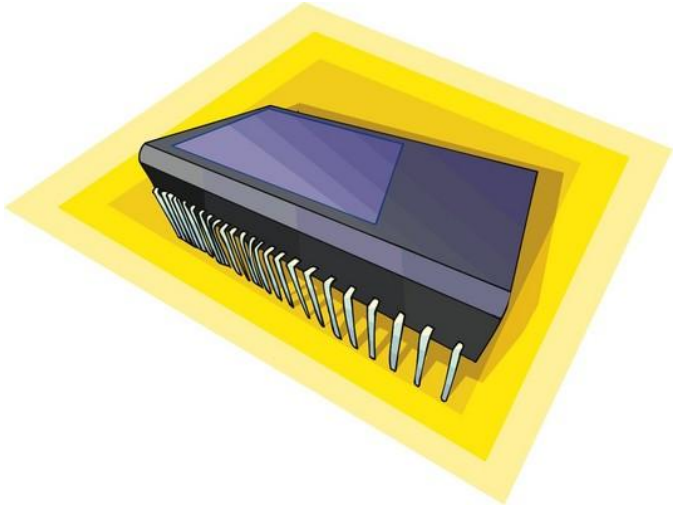
View Video VideoLesson5Bootstrap(C2L5S17).mp4

Supplemental Reading:

- Bootstrap loader
- Bootstrap
- Chainloading in Linux

Rom & CMOS

- **Flash/Rom**
  - Flash/Rom is basically software on a chip—a special set of instructions written to a read-only memory device. Advances in technology now allow users to "flash" their ROM chips via easy-to-use software on the chip. Making changes to ROM chips allow manufacturers to easily update/upgrade commercial products in an efficient and cost effective manner.
- **CMOS**
  - (Complementary metal-oxide-semiconductor) - the semi-conductor based technology used to store information such as time, date, serial number, model #, and configuration information within electronic devices.



Supplemental Reading:

- Flashrom
- Flash updates with Linux
- What is CMOS
- CMOS

Upstart

The following concepts are also important:

- **Upstart**
  - Upstart is an event-based replacement for the /sbin/init daemon which handles starting of tasks and services during boot, stopping services during shutdown, and supervising services while the system is running. It was originally developed for the Ubuntu distribution but is suitable for deployment in all Linux distributions as a replacement for the venerable *System-V init*.
- **Embedded Linux**
  - Embedded Linux is the use of Linux in dedicated or reduced functionality computer systems such as mobile phones, personal digital assistants, media players, set-top boxes, and other consumer electronics devices, networking equipment, machine control, industrial automation, navigation equipment, and medical instruments.
- **Legacy**
  - Legacy is a generic term for old computer devices or applications that continue to be used today. Examples include modems, specialized serial devices, and older peripheral devices.

Select **PLAY** below to view a video on Upstart.

View Video
VideoLesson5Upstart (C2L5S19).mp4

Supplemental Reading:

- Upstart overview
- Upstart manager
- Embedded Linux
- Legacy systems

Runlevels

- **Runtime**
  - The period during which a computer program is executing, or the duration of that period.
- **Runlevel**
  - The term *runlevel* is defined as the different operating levels of a computer OS. The different levels operate independently of each other and can be used for troubleshooting and fault isolation. A great example would be dropping from a GUI-driven application to a terminal program to ping another computer. The terminal program would be operating at a more primitive runlevel and require less resources than the GUI based application. By reducing the level of resources used by the GUI, a smaller "target" of possible root fault causes could be created making fault isolation easier.
  - A better example would be to user the "telinit 1" command to drop from runlevel 5 to runlevel 1, SingleUserMode. Beware, runlevel 1 will most likely shutdown remote access services.

Select **PLAY** below to view a video on runlevels.

View Video VideoLesson5Runlevels (C2L5S20).mp4

Supplemental Reading:

- Runtime
- Runtime definition
- Runlevel
- What are run levels?
- Runlevel definition
- Runlevels explained

Linux OS & Kernel

- **Linux Kernel**
  - The Linux kernel provides the basic services and device drivers used by all other programs running on a Linux OS system.
  - If you are really enthusiastic about the Linux Kernel and wish to learn more, browse this free online book that explores the Kernel in detail

- **Live CD**
  - A CD or DVD that contains a complete operating system that can be run from the disk or device. Examples include Ubuntu, Knoppix, DSL, Mint, and Suse.

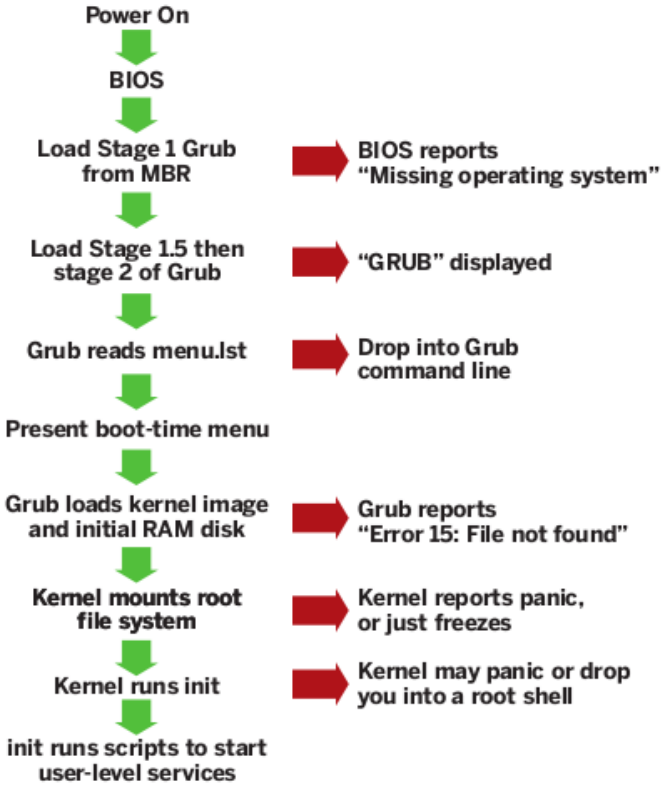
Supplemental Reading:
- Linux Kernel wiki
- Kernel.org
- Linux online

Boot Process

When the PC is first powered up, BIOS is the first program that runs. All the other programs must be loaded into RAM before they can be used. The BIOS consist of the following parts:

1) **POST** (power on self test) - the power on self test is a set of instructions that tests the basic hardware components of a computer system. These components include memory, hard drive, CD-ROM, keyboard, mouse and display. Think of it as a basic check to make sure you have the bare minimum to run a computer.

2) **Setup menu** - The setup menu allows you to select from a variety of options for booting your computer. These options include whether to boot from CD, hard drive, USB, or network. Other options included in the setup menu might include utilities that allow you to modify and select different hardware components, security settings, time and date, and other items necessary to run a computer.



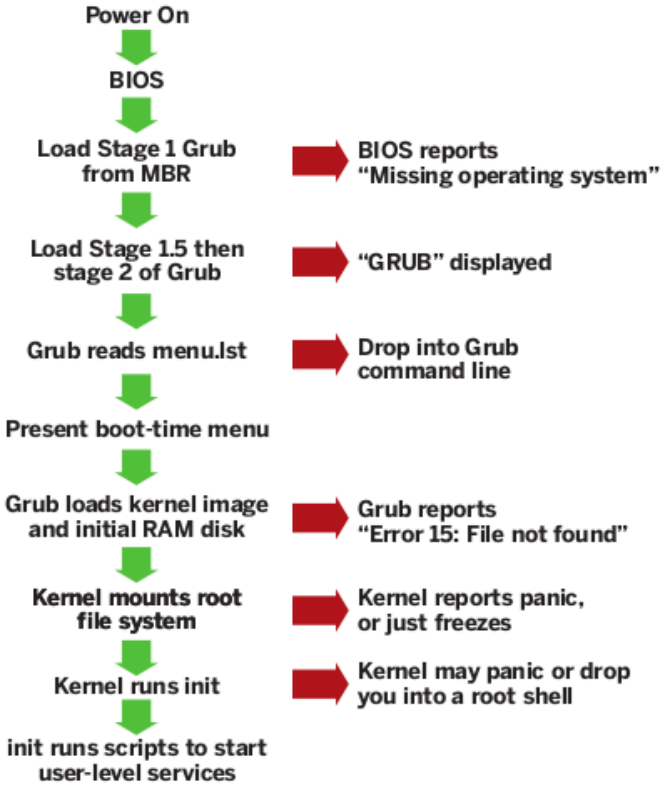
Boot Process (Contd)

3) **The boot sector loader** - The boot sector loader loads the first 512 byte sector from the boot disk into RAM and jumps to it. At this point, the first stage of the GRUB loader activates. This is where the boot loaders described in this lesson start.

4) Stage 1.5 and then Stage 2 of the Grub loader activates. **"Grub"** is displayed on the user screen and the process stops if something goes wrong in this stage.

5) Grub reads the **menu.lst** file. Grub will drop into a command line if a process fails at this point.

6) Boot time menu appears for the user to select a boot option.



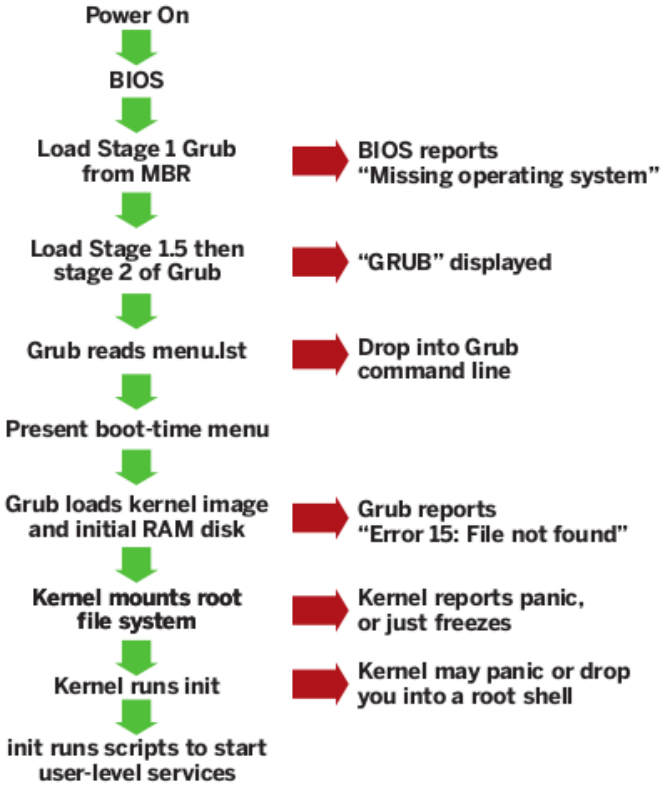
Boot Process (Contd)

7) Grub loads the Kernel image and initial RAM disk. Grub will report "Error 15:" file not found if something goes wrong at this point.

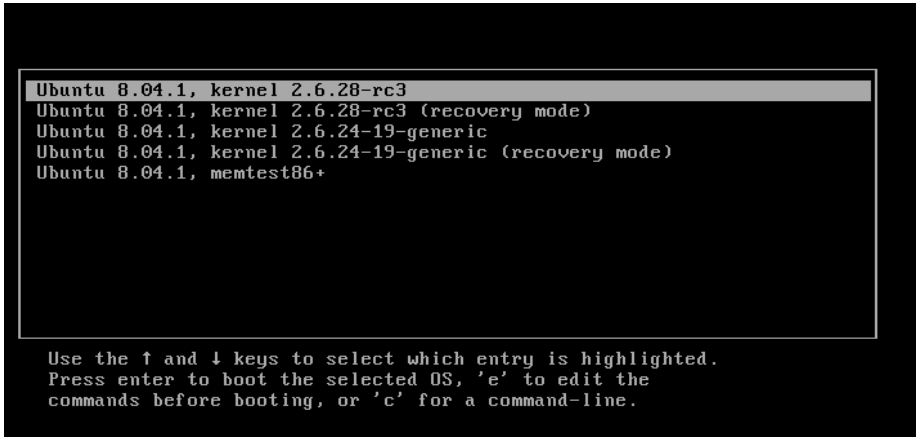
8) Kernel mounts root file system. A failure at this point may generate a Kernel panic report or be indicated by a frozen Kernel.

9) Kernel runs *init*.
Failure at this point would be indicated by a Kernel panic or the Kernel will drop into shell mode.

10) init runs scripts to start user-level services.



Details on GRUB



GRUB operating system selection example

The main purpose of Grub is to find and load the Linux kernel. Grub is now the preferred boot loader by most administrators because it is easy to use and understand. Grub features both graphical and command line interfaces, and is dynamic in nature, meaning it can be configured at boot time. One of the main features of Grub is that it can be configured to boot both Linux and Windows-based systems and can easily replace the native Windows bootloader.

Grub can be easily modified and is included as a default in most Linux setups. The most difficult aspect of Grub during installation is remembering where the *menu.lst* file is installed by default. This file can be configured with any text editor by the system administrator.

GRUB Configuration File

The following sample configuration file is used to explain the basic Grub setup:

The default = option tells GRUB which image to boot by default after the timeout period. For an entry to be valid, it must contain at the very least a title, root and kernel directives.

Timeout specifies how long (in seconds) the menu will be displayed before the default entry is executed.

Note: when testing or making adjustments you may want to extend this out to 20 so you have time to check your notes and do research/monitor what happens.

The **splashimage** option specifies the location of the image for use as the background for the GRUB GUI.

```
default=0
timeout=10
splashimage=(hd1,2)/grub/splash.xpm.gz

title Linux
root (hd1,2)
kernel /vmlinuz-2.6.23-13 quiet root=LABEL=/
initrd /initrd-2.6.23-13.img

title Windows XP
rootnoverify (hd0,0)
chainloader +1
```

Fig 1. GRUB configuration file example

GRUB Configuration File (Contd)

Title is the text that is displayed in the menu for the entry that follows.

Note: It is a best practice to modify the title and include detailed information if the title was created as part of a default installation. Most Linux distributions will create a generic title by default such as "Linux 2.4 generic" or something similar. If you plan to create several partitions and do a multiboot configuration, it may get confusing for you and especially the end user.

Kernel specifies what kernel will be booted once that entry is selected. Following the kernel are the directive options that will be passed to the kernel. In our case, the quiet directive means that debugging information will not be displayed while the kernel loads.

The **initrd** option specifies the file that will be loaded at boot time as the initial RAM disk. In this case, initrd-2.6.23-13.img will be loaded into ram disk and will be executed once the kernel is loaded.

```
default=0
timeout=10
splashimage=(hd1,2)/grub/splash.xpm.gz

title Linux
root (hd1,2)
kernel /vmlinuz-2.6.23-13 quiet root=LABEL=/
initrd /initrd-2.6.23-13.img

title Windows XP
rootnoverify (hd0,0)
chainloader +1
```

Fig 1. GRUB configuration file example

GRUB Configuration File 2

- The **default 0** statement simply points to which choice will run if the user makes no choice within the timeout period.
- The **lang=** is unnecessary unless you are booting a foreign distro and want to see it in your native language.
- The **root (hd0,0**) needs to be adjusted to point to the partition containing the kernel.
- The **root=/dev/hda1** needs to be adjusted to point to the partition that will be mounted as root after booting. **Note :** it is expressed in Linux notation, not in grub notation.
- Grub's default menu file is **/boot/grub/menu.lst**. You can use a different file, but it would require making many changes to your files and setup configuration. Plus, it may confuse those who may work on the system. Use the default.

```
timeout 30

# By default, boot the first entry.
default 0

# For booting GNU/Linux
title  GNU/Linux
root (hd0,0)
kernel /boot/vmlinuz-2.6.7 root=/dev/hda1
lang=us
```

Fig 2. GRUB configuration file 2 example

GRUB Warning

Special notice on how Grub recognizes different hard drive types:

Prior to kernel 2.6.20, Linux differentiated HDDs by type—IDE, SCSI, SATA etc. This naming convention has been corrected from Kernel 2.6.20. HDDs are now called "sd(x)."

Consequently, you need to pay attention to which kernel you are loading or you may get a fault on booting. For example, if you are using kernel 2.4, your IDE hard drive will probably be seen by the kernel as HDA, and if you are using the same computer system with kernel 2.6.20 that same hard drive will be seen as SDA. This is probably the most challenging aspect of Grub!

```
timeout 30

# By default, boot the first entry.
default 0

# For booting GNU/Linux
title  GNU/Linux
root (hd0,0)
kernel /boot/vmlinuz-2.6.7 root=/dev/hda1
lang=us
```

Fig 2. GRUB configuration file 2 example

Troubleshooting GRUB

Troubleshooting Grub is not very difficult once you understand the basics of the Grub configuration. Over time, experienced administrators will become accustomed to the various Grub commands and most administrators will prefer Grub since it allows you to pass various special directives to the kernel. Grub is an excellent boot loader for multi-boot systems.

Before we continue, there is one more thing that needs to be clarified. Make sure you know the difference between Grub terminology and Linux terminology when discussing partitions and booting. For example, the Linux partition containing the kernel is **/dev/hda1** but in Grub it is referred to as (**hd0,0**). See Table 1.

Grub partition specification	Linux partition specification
(hd0,7)	/dev/hda8
(hd0,10)	/dev/hda11
(hd1,10)	/dev/hdd11 Note: In this case, we refer to hd1 as hde because this computer has two hard drives… one is hda and the other is mapped as hdd

Table 1. GRUB & Linux Terminology Differences

Troubleshooting GRUB (Contd)

The majority of boot problems can be solved using the 'e' GRUB command. At boot time, the 'c' command accesses the GRUB command line, or shell. Thankfully, rebooting is not necessary to access the GRUB shell.

Log into Linux as root and execute **/sbin/grub** from the BASH prompt. The 'c' command allows a user to input various command line options for testing the correct load parameters without rebooting and reloading a shell!

In order to troubleshoot a Grub configuration, the following must be present:

1. The partition containing the kernel
2. Within that partition, the directory path and filename of the kernel should be known.
3. The partition must contain **/sbin/init**

Grub has a find tool to help us find files and directories.
The main thing we need to find is our kernel. We will assume the kernel is **vmlinuz**. Once the kernel is found, you can edit the command parameters to get a good system load.

```
grub> find /vmlinuz
 (hdo,o)
 (hdo,11)
 (hd1,11)

grub>
```

Fig 3. Using the **Find** Tool

Supplemental Reading:

- GRUB Error messages

Troubleshooting Links

The following links offer helpful Ubuntu Linux booting and networking support:

- Questions for Ubuntu
- Bootdisk Howto
- Boot problems
- Troubleshooting toolkit
- Ubuntu troubleshooting
- Hotkeys troubleshooting
- Network troubleshooting
- Wifi troubleshooting

Lesson Summary

In this lesson, you explored system initialization. You learned that a computer goes through several diagnostic levels to ensure all the hardware and software in the system are working correctly.

You learned that the BIOS contains basic instructions about hardware and software, and that LILO and Grub are two popular boot loaders for Linux that contain instructions to load the operating system.

You were also introduced to important areas of a hard disk such as the MBR and the boot sector that contain information about the operating system. You learned to load various versions of Linux including DSL and Linux on a live CD. Finally, you explored the boot process and troubleshooting options for GRUB.

For help troubleshooting Linux boot problems, numerous websites and forums are available to help with specific questions.

Select **PLAY** below to view a summary.

View Video VideoLesson5Summary (C2L5S33).mp4