**Question 1: Explain Booting procedure or steps in Linux?**  
  
  
1. Once System powered on, it automatically invokes BIOS   
  
2. BIOS will start the processor and perform a POST [power on self test] to check the connected device are ready to use and are working properly.  
  
3. After POST , BIOS will check for the booting device. The boot sector is always the first sector of the hard disk and BIOS will load the MBR into the memory.   
MBR holds the boot loader of the OS.  
  
4. Then boot loader takes the control of the booting process.  
  
5. GRUB is the boot loader for Linux.   
  
6. Depending on the boot option selected the kernel is loaded first.  
  
7. After kernel is loaded the kernel will take the control of the booting process  
  
8. Initrd will be loaded which contains drivers to detect hardware (its called Initialization of RAM Disk)  
  
9. Then it will initialize all the hardware including I/O processors etc.  
  
10. Kernel will mounts the root partition as read-only  
  
11. INIT is loaded as the first process.  
  
12. INIT will mount the root partition and other partitions as read/write and checks for file system errors.  
  
13. Sets the System Clock, hostname etc..  
  
14. Based on the Runlevel, it will load the services and runs the startup scripts which are located in /etc/rcX.d/ (Network, nfs, SSH etc.)  
  
15. Finally it runs the rc.local script & Now the login prompt will appear.

**Question 2: What is stage 1.5 boot loaded in linux?**

The great thing about GRUB is that it includes knowledge of Linux file systems. Instead of using raw sectors on the disk, as LILO does,   
GRUB can load a Linux kernel from an ext2 or ext3 file system. It does this by making the two-stage boot loader into a three-stage boot loader.  
  
A. Stage 1.5 boot loader , it contains extra coe to allow cylinders above 1024, or LBA type drives, to be read.  
B. It will be stored on MBR or Boot partition .  
C. Stage 1 (MBR) boots a stage 1.5 boot loader that understands the particular file system containing the Linux kernel image.  
D. Basically this module will load the knowledge of Filesystem to Grub to read the kernel  
  
so ,  
  
Stage 1 Boot loaded is : MBR  
Stage 1.5 Boo loader : e2fs\_stage1\_5   
Stage 2 Boot loader is : GRUB

**Question 3: How to reinstall GRUB?**

A.Boot up using RHEL4 disk.  
  
B.Enter into rescue mode  
  
#linux rescue (hit ok)  
  
C. Then follow below commands  
  
# chroot /mnt/sysimage  
  
# grub  
  
# find /boot/grub/stage1 or find /grub/stage1  
  
root(hd0,0) //example o/p  
  
Now install the GRUB  
  
# setup (hd0)   
  
# EXIT   
  
Another Method  
  
#linux rescue  
  
# chroot /mnt/sysimage  
  
# /sbin/grub-install /dev/hda

**Question 4: Linux Booting Issues : How to solve ??**

Option 1: init not found error  
Option 2: Run fsck on all FS in rescue mode  
Option 3: Reinstall GRUB  
Option 4: Recover grub.conf / grub configuration  
  
**Option 1: For normal panic and "init not found" error.**  
  
Error : "init not found" displayed  
  
1) Launch the system to Bash shell prompt   
  
Reboot the server and interrupt to edit the GRUB.  
  
Edit grub and enter the below in last   
  
init=/bin/bash   
  
Then save and exit and boot the server. This will launch you straight into a Bash shell prompt.Then you can remount “/” file system and check /var/log/messages for any error.  
  
Note : init=/bin/bash (Grub boot loader) or linux init=/bin/bash (if Lilo boot loader).   
  
2) Once server booted and if it is in Bash shell prompt   
  
#mount -o remount,rw /  
  
3) Now you can check the log messages and try to find the reason for server pacnic or error.  
  
#more /var/log/messages  
  
  
**Option 2: If the above option not helped then follow the next**  
  
1) Boot from the Linux First CD (boot CD).  
  
2) Type “boot rescue” at Linux boot prompt.  
  
3) After the bash shell prompt show up, type the below command   
  
# chroot /mnt/sysimage  
  
a) Run fsck and Check for any disk error  
  
#fdisk -l /dev/sda //check how many partion you have   
  
then run fsck on each partition  
  
#fsck -y /dev/sda2'  
  
**Option 3: If the above also not helped then reinstall grub and retry.**  
  
In rescue mode.  
  
# chroot /mnt/sysimage  
  
# /sbin/grub-install /dev/hda  
  
  
  
 **Option 4: If a system has issues with the GRUB configuration**   
  
(possibly caused by incorrect changes to the the GRUB configuration file, installation of another OS, changes to device ordering due to hardware or BIOS changes, etc.)   
  
# grub> find /boot/grub/grub.conf (or) grub>find /grub/grub.conf (or) find /boot/grub/stage1  
(hd0,1)  
(hd1,2)  
  
>> This tells us that we have two /boot partitions. Then we have to reinstall the GRUB config on disk (one by one) and try.  
  
#grub> root (hd0,1) //Write the GRUB bootloader on the MBR of the first disk  
grub> setup (hd0)  
grub>quit  
  
If you have doubt as to where the root partition is located then try to find a file in /etc.   
  
#grub> find /etc/fstab  
(hd0,1)  
  
  
Note: You must pay attention to your devices, for me "hd0" is the root disk and (hd0,1) is /boot partition , and (hd0,1) is my ROOT (/) partition. mostly / "root" partion will be on LVM.  
You might not even have "hd0" mapped out. Review your "/boot/grub/device.map" file  
  
#cat /boot/grub/device.map

**Question 5 : How to recover or rest Root password in LINUX?**

While booting   
  
1. Select the kernel  
2. Press the "e" key to edit the entry  
3. Select second line (the line starting with the word kernel)  
4. Press the "e" key to edit kernel entry so that you can append single user mode  
5. Append the letter "S" (or word Single) to the end of the (kernel) line  
6. Press ENTER key  
7. Now press the b key to boot the Linux kernel into single user mode  
8. At prompt type passwd command to reset password:  
  
You need to mount at least / and other partitions:  
# mount -t proc proc /proc  
# mount -o remount,rw /  
  
Change the root password,  
  
# passwd  
  
thenreboot system:  
# sync  
# reboot  
  
  
  
**Question 6: What is super Block and how will u recover it ?**  
  
  
The blocks used for two different purpose:  
  
1. Most blocks stores user data aka files (user data).  
2. Some blocks in every file system store the file system's metadata.   
  
So what the hell is a metadata?  
  
File system type  
Size  
Status  
Information about other metadata structures  
  
  
**To find super block**  
  
#dumpe2fs /dev/sda3|grep -i superblock  
  
or   
  
# mke2fs -n /dev/sda3  
  
  
  
To repair file system by alternative-superblock use command as follows:  
  
# e2fsck -f -b 8193 /dev/sda3

### linux boot process

**Linux boot process:**  
**1. loading mbr info about bootloader**-->mbr is the first sector of your primary HDD (1sector = 512bytes),it contains 64B of partition info(what is located where on you drive)  
Bootloader is GRUB in my case.Its a 2 stage process, the first stage is pointing to the conf file (menu.lst or grub.conf) that tells you what all options (kernel,OS) you have that you can boot into.   
  
**2. bootloader choose of kernel version**-->Now you have made a choice.The stage 2 of bootloader proceeds to copy the initrd and kernel intot the memory.   
**3. initializing kernel and initrd image  
4. loading kernel modules**initrd loads the required modules in the kernel.  
**5. discovering hardware  
6. looking for disks  
7. R/O mount of / partition**intird has lsmod built into it and it is mainly responsible for loading the required modules into the kernel.Once all modules are loaded the kernel is able to talk to all the initrd is relived off the memory to reduce the memory footprint and the kernel proceeds to mount the root file system (/) on the HDD.Once / is mounted it does a find for init in the /sbin directory and runs the init process.  
**8. init process spawn  
9. /etc/inittab read and executing  
10. mounting all FSes from /etc/fstab  
11. runlevels running (based on default**

**1) What is GRUB**  
  
  
Ans GNU GRUB is a Multiboot boot loader. It was derived from GRUB, the GRand Unified Bootloader,   
which was originally designed and implemented by Erich Stefan Boleyn.  
  
  
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 the Hurd or Linux). The kernel, in turn, initializes the rest of the operating   
system (e.g. GNU)  
  
  
**2) Explain Linux Boot Process**  
  
  
Ans http://www.thegeekstuff.com/2011/02/linux-boot-process/  
  
  
**3) Which files are called for user profile by default when a user gets login**  
  
  
Ans **$HOME/.bash\_profile, $HOME/.bash\_bashrc**  
  
  
**4) Which file needs to update if srequired to change default runlevel 5 to 3**  
  
  
Ans File is /etc/inittab and required to change below lines:  
  
  
**id:5:initdefault: to id:3:initdefault:**  
  
  
**5) What command used for showing user info like Login Name, Canonical Name, Home Directory,Shell etc..**  
  
  
Ans **FINGER** command can be used i.g; finger username  
  
  
**6) What is inode number**  
  
  
Ans An inode is a data structure on a traditional Unix-style file system such as UFS or ext3. An   
inode stores basic information about a regular file, directory, or other file system object  
  
  
iNode number also called as index number, it consists following attributes:  
  
  
**File type (executable, block special etc)  
Permissions (read, write etc)  
Owner  
Group  
File Size  
File access, change and modification time (remember UNIX or Linux never stores file creation   
time, this is favorite question asked in UNIX/Linux sys admin job interview)  
File deletion time  
Number of links (soft/hard)  
Extended attribute such as append only or no one can delete file including root user   
(immutability)  
Access Control List (ACLs)**  
  
  
Following command will be used to show inodes of file and folders:  
  
  
**ls -i**  
  
  
Following command will show complete info about any file or folders with inode number  
  
  
**stat file/folder**  
  
  
Files/Folders can also be deleted using inode numbers with following command:  
  
  
find out the inode number using 'ls -il' command then run below command  
  
  
**find . -inum inode\_number -exec rm -i {} \;**  
  
  
**7) How can we increase disk read performance in single command**  
  
  
Ans **blockdev command**  
  
  
This is sample output - yours may be different.  
# Before test  
$ blockdev --getra /dev/sdb  
256  
$ time dd if=/tmp/disk.iso of=/dev/null bs=256k  
2549+1 records in  
2549+1 records out  
668360704 bytes (668 MB) copied, 6,84256 seconds, 97,7 MB/s  
  
  
real 0m6.845s  
user 0m0.004s  
sys 0m0.865s  
  
  
# After test  
$ blockdev --setra 1024 /dev/sdb  
  
  
$ time dd if=/tmp/disk.iso of=/dev/null bs=256k  
2435+1 records in  
2435+1 records out  
638390272 bytes (638 MB) copied, 0,364251 seconds, 1,8 GB/s  
  
  
real 0m0.370s  
user 0m0.001s  
sys 0m0.370s  
  
  
**8) .... command to change user password expiration time**  
  
  
Ans CHAGE  
  
  
**9) Command used to lock user password**  
  
  
Ans usermod -L username  
  
  
**10) How many default number of Shells available and what are their names?**  
  
  
Ans SH, BASH, CSH, TCSH, NOLOGIN, KSH  
  
  
**11) Which file defines the attributes like UID, PASSWORD expiry, HOME Dir create or not while**   
**adding user**  
  
  
Ans /etc/login.defs  
  
  
**12) ...... command used for changing authentication of linux system to LDAP/NIS/SMB/KERBOS**  
  
  
Ans authconfig  
  
  
**13) ...... command used for changing the attributes of any file**  
  
  
Ans chattr  
  
  
**14) What is the path of network (ethX) configuration files**  
  
  
Ans /etc/sysconfig/network-scripts/ethX  
  
  
**15) How can we change speed and make full duplex settings for eth0**  
  
  
Ans We can do this with below given 2 methods:  
  
  
ethtool -s eth0 speed 100 duplex full  
ethtool -s eth0 speed 10 duplex half  
  
  
OR  
  
  
mii-tool -F 100baseTx-HD  
mii-tool -F 10baseT-HD  
  
  
**16) File which stores the DNS configuration**  
  
  
Ans /etc/resolve.conf  
  
  
**17) Main configuration file and command used for NFS enabling exported directories and deamons**  
  
  
Ans /etc/exports and exportfs -av , deamons are quotad, portmapper, mountd, nfsd and nlockmgr/status  
  
  
**18) What is command to check ports running/used over local machine**  
  
  
Ans netstat -antp  
  
  
**19) What is the command to check open ports at remote machine**  
  
  
Ans nmap  
  
  
**20) What is the difference between soft and hard links**  
  
  
Ans Soft Links => 1) Soft link files will have different inode numbers then source file  
2) If original file deleted then soft link file be of no use  
3) Soft links are not updated  
4) Can create links between directories  
5) Can cross file system boundaries  
  
  
Hard Links => 1) Hard links will have the same inode number as source file  
2) Hard links can not link directories  
3) Can not cross file system boundaries  
4) Hard links always refers to the source, even if moved or removed  
  
  
**21) How to setup never expired user password**  
  
  
Ans chage -E never username  
  
  
**22) Restricting insertion into file if full permission are assigned to all**  
  
  
Ans chattr +i filename  
  
  
**23) Display or Kill all processes which are accessing any folder/file**  
  
  
Ans Display User who are using file/folder : fuser -u file/folder  
Kill All Processes which are using file/folder: fuser -k file/folder  
  
  
**24) Kill any user's all processes**  
  
  
Ans killall -u username  
  
  
**25) How can we have system analysis and reports over mail**  
  
  
Ans Use logwatch  
  
  
**26) What we have to do if we do required to rotate logs without moving and creating new log file**  
  
  
Ans We can use "logrotate"'s "copytruncate" option which will simply copy original file and   
truncate original file :)  
  
  
**27) Command to collect detailed information about the hardware and setup of your system**  
  
  
Ans dmidecode , sysreport  
  
  
**28) Command to check PCI devices vendor or version**  
  
  
Ans lspci  
  
  
**29) What is the difference between cron and anacron**  
  
  
Ans Cron :  
1) Minimum granularity is minute (i.e Jobs can be scheduled to be executed  
every minute)  
2) Cron job can be scheduled by any normal user ( if not restricted by super  
user )  
3) Cron expects system to be running 24 x 7. If a job is scheduled, and  
system is down during that time, job is not executed  
4) Ideal for servers  
5) Use cron when a job has to be executed at a particular hour and minute  
  
  
Anacron :  
1) Minimum granularity is only in days  
2) Anacron can be used only by super user ( but there are workarounds to  
make it usable by normal user )  
3) Anacron doesn’t expect system to be running 24 x 7. If a job is scheduled,  
and system is down during that time, it start the jobs when the system  
comes back up.  
4) Ideal for desktops and laptops  
5) Use anacron when a job has to be executed irrespective of hour and  
minute  
  
  
**30) Default Port numbers used by ssh,ftp,http,https,telnet,smtp,pop3,pop3s,imap,imaps**  
  
  
Ans SSH 22, ftp 20/21, http 80, https 443, SMTP/SMPTS 25/465, POP3/POP3S 110/995, IMAP/IMAPS 143/993  
  
  
**31) How to setup ACLs in following case:**  
**1) Create a file FILE1 and this should be read,write,executable for all user but Read only for user USER1**  
**2) Copy FILE1 ACLs to FILE2 ACL**  
**3) Delete a USER1's rule for FILE1 which were setup in step 1)**  
  
  
Ans 1) touch FILE1 ; chmod 777 FILE1 ; setfacl -m u:USER1:r FILE1  
2) getfacl FILE1 | setfacl --set-file=- FILE2  
3) setfacl -x u:USER1 FILE1  
  
  
**32) How to make USB bootable?**  
  
  
Ans Write efidisk.img from RHEL 6 DVD images/ subdirectory to USB  
  
  
dd if=efidisk.img of=/dev/usb (usb device name)  
  
  
**33) How can we check disk/device status/failure/errors using smartctl utility?**  
  
  
Ans Try following to check:  
  
  
Enable/Disable SMART on device/disk : **smartctl -s on /dev/sda**  
Check device SMART health : **smartctl -H /dev/sda**  
Check device SMART capabilities : **smartctl -c /dev/sda**  
Enable/Disable automatic offline testing on device : **smartctl -o on/off /dev/sda**  
Show device SMART vendor-specific Attributes and values : **smartctl -A /dev/sda**  
Show device log [TYPE : error, selftest, selective, directory,background,   
scttemp[sts,hist]] : **smartctl -l TYPE /dev/sda**  
Run test on device [TEST: offline short long conveyance select,M-N pending,N   
afterselect,[on|off] scttempint,N[,p] : **smartctl -t /dev/sda**  
  
  
**34) What is the difference between ext2 vs ext3 vs ext4?**  
  
  
Ans <http://www.thegeekstuff.com/2011/05/ext2-ext3-ext4/>  
  
  
**35) Disable ping to avoid network/ICMP flood**  
  
  
Ans Set following in /etc/sysctl.conf : net.ipv4.icmp\_echo\_ignore\_all = 1  
  
  
Then "sysctl -p"  
  
or   
  
**echo "1" > /proc/sys/net/ipv4/icmp\_echo\_ignore\_all**  
  
  
**36) What is SYN Flood, ICMP Flood**  
  
  
Ans **SYN Flood** : A [SYN flood](http://en.wikipedia.org/wiki/SYN_flood) occurs when a host sends a flood of TCP/SYN packets, often with a   
fake/forged sender address. Each of these packets is handled like a connection request, causing the   
server to spawn a [half-open connection](http://en.wikipedia.org/wiki/Half-open_connection), by sending back a TCP/SYN-ACK packet(Acknowledge), and   
waiting for a packet in response from the sender address(response to the ACK Packet). However,   
because the sender address is forged, the response never comes. These half-open connections   
saturate the number of available connections the server is able to make, keeping it from responding to   
legitimate requests until after the attack ends  
  
  
**ICMP Flood** : There are three types of ICMP Flood :  
  
1) Smurf Attack : <http://en.wikipedia.org/wiki/Smurf_attack>  
2) Ping Flood : <http://en.wikipedia.org/wiki/Ping_flood>  
3) Ping of Death : <http://en.wikipedia.org/wiki/Ping_of_death>  
  
  
**37) What is the difference between Unix vs Linux Kernels?**  
  
  
Ans Please find below given link :  
  
  
<http://www.thegeekstuff.com/2012/01/linux-unix-kernel/>  
  
  
**38) How to setup Password less remote login/ssh?**  
  
  
Ans Use "ssh-keygen -t dsa or rsa" at local system for creating public and private keys  
  
  
Then copy /root/.ssh/id\_dsa.pub to remote\_server by name /root/.ssh/authorized\_keys  
Change permissions of /root/.ssh/authorized\_keys file at remote\_server "chmod 0600 ~/.ssh/authorized\_keys"  
  
Now try to login from local system to remote\_server "ssh root@remote\_server"  
  
  
**39) Command to see default kernel image file**  
  
  
Ans "grubby --default-kernel"  
  
  
**40) How to create lvm mirror**  
  
  
Ans lvcreate -L 50G -m1 -n LVMmirror vg0  
  
  
**41) Command to check last runlevel**  
  
  
Ans who -r  
  
  
**42) What do you mean by File System?**  
  
  
Ans File System is a method to store and organize files and directories on disk. A file system can have different formats called file system types. These formats determine how the information is stored as files and directories.  
  
  
**43) What is the requirement of udev daemon?**  
  
  
Ans Create and remove device nodes or files in /dev/ directory  
  
  
**44) What are block and character devices?**  
  
  
Ans Both the devices are present in /dev directory  
  
  
Block device files talks to devices block by block [1 block at a time (1 block = 512 bytes to 32KB)].  
Examples: - USB disk, CDROM, Hard Disk (sda, sdb, sdc etc....)

Character device files talk to devices character by character.  
Examples: - Virtual terminals, terminals, serial modems, random numbers (tty{0,1,2,3......})

**45) How to Convert ext2 to ext3 File System?**

Ans tune2fs -j /dev/{device-name}  
  
  
**46) File required to modify for setting up kernel parameters permanent**  
  
  
Ans /etc/sysctl.conf  
  
  
**47) Commands used to install, list and remove modules from kernel**   
  
  
Ans Installing/adding a module:   
insmod mod\_name  
modprobe mod\_name  
  
  
List installed modules : lsmod  
Removing a module : modprobe -r mod\_name  
  
  
**48) How to create swap using a file and delete swap**  
  
  
Ans Adding swap :  
  
  
dd if=/dev/zero of=/opt/myswap bs=1024 count=4  
  
  
mkswap /opt/myswap  
  
  
swapon -a  
  
  
For adding this myswap at boot time, add following in /etc/fstab file:  
  
  
/opt/myswap swap swap defaults 0 0  
  
  
  
  
Deleting Swap :  
  
  
Run "swapoff /opt/myswap" command  
Remove the entry from /etc/fstab file  
Remove /opt/myswap file (using rm command)  
  
  
**49) What vmstat show**  
  
  
Ans **vmstat** (virtual memory statistics) is a computer system monitoring tool that collects and displays summary information about operating system **memory, processes, interrupts, paging and block I/O**

**50) What is tmpfs File System**

Ans Reference : <http://en.wikipedia.org/wiki/Tmpfs>

**tmpfs** is a common name for a temporary file storage facility on many [Unix-like](http://en.wikipedia.org/wiki/Unix-like) operating systems. It is intended to appear as a mounted [file system](http://en.wikipedia.org/wiki/File_system), but stored in [volatile memory](http://en.wikipedia.org/wiki/Volatile_memory) instead of a persistent storage device. A similar construction is a [RAM disk](http://en.wikipedia.org/wiki/RAM_disk), which appears as a virtual disk drive and hosts a [disk file system](http://en.wikipedia.org/wiki/Disk_file_system).

Everything stored in tmpfs is temporary in the sense that no files will be created on the [hard drive](http://en.wikipedia.org/wiki/Hard_drive); however, swap space is used as backing store in case of low memory situations. On [reboot](http://en.wikipedia.org/wiki/Reboot_(computer)), everything in tmpfs will be lost.

The memory used by tmpfs grows and shrinks to accommodate the files it contains and can be swapped out to swap space.

**51) What is the difference between screen and script commands?**

Ans **Screen** is an screen manager with VT100/ANSI terminal emulation and used to take GNU screen session remotely or locally and while S**cript** make typescript of terminal session  
  
  
Screen : needs to be detached, should not be exited to access remotely/locally  
Script : creates a file and store all the terminal output to this file  
  
**52) How can we check which process is assigned to which processor?**  
  
  
Ans Run "ps -elFL" and find out the PSR column which is showing the processor number to the process  
  
  
**53) How can we check vendor, version, release date, size, package information etc... of any installed rpm?**  
  
  
Ans) rpm -qi package-name , for example:

rpm -qi ypbind-1.19-12.el5

# 6 Stages of Linux Boot Process (Startup Sequence)

by Ramesh Natarajan on February 7, 2011

Press the power button on your system, and after few moments you see the Linux login prompt.

Have you ever wondered what happens behind the scenes from the time you press the power button until the Linux login prompt appears?

The following are the 6 high level stages of a typical Linux boot process.  
  


### 1. BIOS

* BIOS stands for Basic Input/Output System
* Performs some system integrity checks
* Searches, loads, and executes the boot loader program.
* It looks for boot loader in floppy, cd-rom, or hard drive. You can press a key (typically F12 of F2, but it depends on your system) during the BIOS startup to change the boot sequence.
* Once the boot loader program is detected and loaded into the memory, BIOS gives the control to it.
* So, in simple terms BIOS loads and executes the MBR boot loader.

### 2. MBR

* MBR stands for Master Boot Record.
* It is located in the 1st sector of the bootable disk. Typically /dev/hda, or /dev/sda
* MBR is less than 512 bytes in size. This has three components 1) primary boot loader info in 1st 446 bytes 2) partition table info in next 64 bytes 3) mbr validation check in last 2 bytes.
* It contains information about GRUB (or LILO in old systems).
* So, in simple terms MBR loads and executes the GRUB boot loader.

### 3. GRUB

* GRUB stands for Grand Unified Bootloader.
* If you have multiple kernel images installed on your system, you can choose which one to be executed.
* GRUB displays a splash screen, waits for few seconds, if you don’t enter anything, it loads the default kernel image as specified in the grub configuration file.
* GRUB has the knowledge of the filesystem (the older Linux loader LILO didn’t understand filesystem).
* Grub configuration file is /boot/grub/grub.conf (/etc/grub.conf is a link to this). The following is sample grub.conf of CentOS.

#boot=/dev/sda

default=0

timeout=5

splashimage=(hd0,0)/boot/grub/splash.xpm.gz

hiddenmenu

title CentOS (2.6.18-194.el5PAE)

root (hd0,0)

kernel /boot/vmlinuz-2.6.18-194.el5PAE ro root=LABEL=/

initrd /boot/initrd-2.6.18-194.el5PAE.img

* As you notice from the above info, it contains kernel and initrd image.
* So, in simple terms GRUB just loads and executes Kernel and initrd images.

### 4. Kernel

* Mounts the root file system as specified in the “root=” in grub.conf
* Kernel executes the /sbin/init program
* Since init was the 1st program to be executed by Linux Kernel, it has the process id (PID) of 1. Do a ‘ps -ef | grep init’ and check the pid.
* initrd stands for Initial RAM Disk.
* initrd is used by kernel as temporary root file system until kernel is booted and the real root file system is mounted. It also contains necessary drivers compiled inside, which helps it to access the hard drive partitions, and other hardware.

### 5. Init

* Looks at the /etc/inittab file to decide the Linux run level.
* Following are the available run levels
  + 0 – halt
  + 1 – Single user mode
  + 2 – Multiuser, without NFS
  + 3 – Full multiuser mode
  + 4 – unused
  + 5 – X11
  + 6 – reboot
* Init identifies the default initlevel from /etc/inittab and uses that to load all appropriate program.
* Execute ‘grep initdefault /etc/inittab’ on your system to identify the default run level
* If you want to get into trouble, you can set the default run level to 0 or 6. Since you know what 0 and 6 means, probably you might not do that.
* Typically you would set the default run level to either 3 or 5.

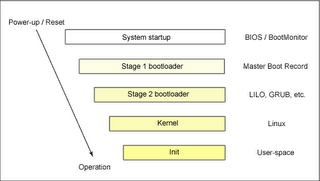
### 6. Runlevel programs

* When the Linux system is booting up, you might see various services getting started. For example, it might say “starting sendmail …. OK”. Those are the runlevel programs, executed from the run level directory as defined by your run level.
* Depending on your default init level setting, the system will execute the programs from one of the following directories.
  + Run level 0 – /etc/rc.d/rc0.d/
  + Run level 1 – /etc/rc.d/rc1.d/
  + Run level 2 – /etc/rc.d/rc2.d/
  + Run level 3 – /etc/rc.d/rc3.d/
  + Run level 4 – /etc/rc.d/rc4.d/
  + Run level 5 – /etc/rc.d/rc5.d/
  + Run level 6 – /etc/rc.d/rc6.d/
* Please note that there are also symbolic links available for these directory under /etc directly. So, /etc/rc0.d is linked to /etc/rc.d/rc0.d.
* Under the /etc/rc.d/rc\*.d/ directories, you would see programs that start with S and K.
* Programs starts with S are used during startup. S for startup.
* Programs starts with K are used during shutdown. K for kill.
* There are numbers right next to S and K in the program names. Those are the sequence number in which the programs should be started or killed.
* For example, S12syslog is to start the syslog deamon, which has the sequence number of 12. S80sendmail is to start the sendmail daemon, which has the sequence number of 80. So, syslog program will be started before sendmail.

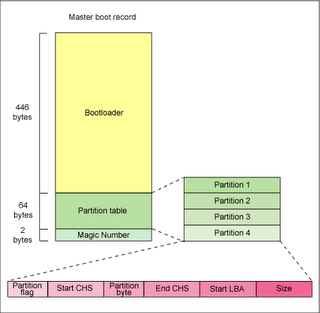
There you have it. That is what happens during the Linux boot process.

### [Linux Boot Process](http://www.sysadminshare.com/2011/03/linux-boot-process.html)

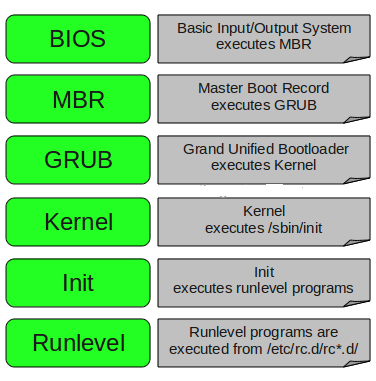
3:20 PM Posted byMadeswaran N

[](https://lh6.googleusercontent.com/-yIfId_eSMng/TXX7ReerJFI/AAAAAAAABk4/BDXLtxiNmcM/s1600/linux+boot+process.gif)

**1. POST :**  
  
booting Linux begins in the BIOS at address 0xFFFF0.  
It starts POST : to check the hardware & local device enumeration and initialization.  
  
BIOS has 2 parts:   
1. POST code and   
2. Runtime services (BIOS runtime searches for devices that are both active and bootable)  
  
BIOS looks MBR in hardisk and its loads the content of MBR to memory (RAM).and then it passes the control to MBR.  
  
MBR :MBR contains the primary boot loader.  
The MBR is a 512-byte sector,   
Located in the first sector on the disk (sector 1 of cylinder 0, head 0).   
  
**2. STAGE 1 Boot loader:**

[](https://lh3.googleusercontent.com/-cVtSME8tGyo/TXX7ro1jOiI/AAAAAAAABk8/UnJrbc5boFI/s1600/MBR+anatomy.gif)

**MBR** is a 512-byte image containing both program code and a small partition table.  
  
first 446 bytes : Primary boot loader (contains both executable code and error message text)  
Next 64 Bytes : The partition table (contains a record for each of four partitions )  
Last 2 Bytes : The magic number (0xAA55) : (magic number serves as a validation check of the MBR.)  
  
Job of the primary boot loader is to find and load the secondary boot loader (stage 2)  
  
**3.Stage 2 boot loader:**  
  
Task at this stage is to load the Linux kernel and optional initial RAM disk.  
  
The first & second-stage boot loaders are called Linux Loader (LILO) or GRand Unified Bootloader (GRUB).  
  
GRUB can load a Linux kernel from an ext2 or ext3 file system.   
GRUB : Understands file systems and configuration lives in /boot/grub/menu.lst or /boot/boot/menu.lst   
  
Stage 1 (MBR) boots a **stage 1.5 boot loader** that understands the particular file system containing the Linux kernel image.  
  
Once stage 2 loaded, GRUB can, upon request, display a list of available kernels (defined in /etc/grub.conf)  
  
Default kernel image and initrd image are loaded into memory. then the stage 2 boot loader invokes the kernel image.  
  
**4. Kernel :**  
  
Kernel initialize the devices and Loads initrd.  
  
Kernel executes /linuxrc to mount the root file system.And then it runs /sbin.init  
  
/sbin/init reads the content of /etc/inittab and run the boot scripts @ /etc/rc.d/rc.sysinit.  
  
/etc/rc.d.rc.sysinit :It loads modules, check root FS and mount RW, mount local FS, setup network, and mount remote FS.  
  
Finally based on the /etc/inittab entry system switches to default runlevel.

**Start-Up Sequence**  
  
Press the power button on your system, and after few moments you see the Linux login prompt.  
Have you ever wondered what happens behind the scenes from the time you press the power button until the Linux login prompt appears?  
The following are the 6 high level stages of a typical Linux boot process.[](http://4.bp.blogspot.com/-BPzX5H6jfD0/UIKIxa6j1YI/AAAAAAAAAEA/484nj_iRI7c/s1600/boot_pro.PNG)  
**1. BIOS**  
♣ BIOS stands for Basic Input/Output System  
♣ Performs some system integrity checks  
♣ Searches, loads, and executes the boot loader program.  
♣ It looks for boot loader in floppy, cd-rom, or hard drive. You can press a key (typically F12 of F2, but it depends on your system) during the BIOS startup to change the boot sequence.  
♣ Once the boot loader program is detected and loaded into the memory, BIOS gives the control to it.  
♣ So, in simple terms BIOS loads and executes the MBR boot loader.  
  
**2. MBR**  
♣ MBR stands for Master Boot Record.  
♣ It is located in the 1st sector of the bootable disk. Typically /dev/hda, or /dev/sda  
♣ MBR is less than 512 bytes in size. This has three components 1) primary boot loader info in 1st 446 bytes 2) partition table info in next 64 bytes 3) mbr validation check in last 2 bytes.  
♣ It contains information about GRUB (or LILO in old systems).  
♣ So, in simple terms MBR loads and executes the GRUB boot loader.  
  
**3. GRUB**  
♣ GRUB stands for Grand Unified Bootloader.  
♣ If you have multiple kernel images installed on your system, you can choose which one to be executed.  
♣ GRUB displays a splash screen, waits for few seconds, if you don’t enter anything, it loads the default kernel image as specified in the grub configuration file.  
♣ GRUB has the knowledge of the filesystem (the older Linux loader LILO didn’t understand filesystem).  
♣ Grub configuration file is /boot/grub/grub.conf (/etc/grub.conf is a link to this). The following is sample grub.conf of CentOS.  
#boot=/dev/sda  
default=0  
timeout=5  
splashimage=(hd0,0)/boot/grub/splash.xpm.gz  
hiddenmenu  
title CentOS (2.6.18-194.el5PAE)  
root (hd0,0)  
kernel /boot/vmlinuz-2.6.18-194.el5PAE ro root=LABEL=/  
initrd /boot/initrd-2.6.18-194.el5PAE.img  
♣ As you notice from the above info, it contains kernel and initrd image.  
♣ So, in simple terms GRUB just loads and executes Kernel and initrd images.  
  
**4. Kernel**  
♣ Mounts the root file system as specified in the “root=” in grub.conf  
♣ Kernel executes the /sbin/init program  
♣ Since init was the 1st program to be executed by Linux Kernel, it has the process id (PID) of 1. Do a ‘ps -ef | grep init’ and check the pid.  
♣ initrd stands for Initial RAM Disk.  
♣ initrd is used by kernel as temporary root file system until kernel is booted and the real root file system is mounted. It also contains necessary drivers compiled inside, which helps it to access the hard drive partitions, and other hardware.  
  
**5. Init**  
♣ Looks at the /etc/inittab file to decide the Linux run level.  
♣ Following are the available run levels  
♣ 0 – halt  
♣ 1 – Single user mode  
♣ 2 – Multiuser, without NFS  
♣ 3 – Full multiuser mode  
♣ 4 – unused  
♣ 5 – X11  
♣ 6 – reboot  
♣ Init identifies the default initlevel from /etc/inittab and uses that to load all appropriate program.  
♣ Execute ‘grep initdefault /etc/inittab’ on your system to identify the default run level  
♣ If you want to get into trouble, you can set the default run level to 0 or 6. Since you know what 0 and 6 means, probably you might not do that.  
♣ Typically you would set the default run level to either 3 or 5.  
  
**6. Runlevel programs**  
♣ When the Linux system is booting up, you might see various services getting started. For example, it might say “starting sendmail …. OK”. Those are the runlevel programs, executed from the run level directory as defined by your run level.  
♣ Depending on your default init level setting, the system will execute the programs from one of the following directories.  
♣ Run level 0 – /etc/rc.d/rc0.d/  
♣ Run level 1 – /etc/rc.d/rc1.d/  
♣ Run level 2 – /etc/rc.d/rc2.d/  
♣ Run level 3 – /etc/rc.d/rc3.d/  
♣ Run level 4 – /etc/rc.d/rc4.d/  
♣ Run level 5 – /etc/rc.d/rc5.d/  
♣ Run level 6 – /etc/rc.d/rc6.d/  
♣ Please note that there are also symbolic links available for these directory under /etc directly. So, /etc/rc0.d is linked to /etc/rc.d/rc0.d.  
♣ Under the /etc/rc.d/rc\*.d/ direcotiries, you would see programs that start with S and K.  
♣ Programs starts with S are used during startup. S for startup.  
♣ Programs starts with K are used during shutdown. K for kill.  
♣ There are numbers right next to S and K in the program names. Those are the sequence number in which the programs should be started or killed.  
♣ For example, S12syslog is to start the syslog deamon, which has the sequence number of 12. S80sendmail is to start the sendmail daemon, which has the sequence number of 80. So, syslog program will be started before sendmail

### Linux Boot Process

**BIOS (Basic Input/output System):**

 When we power on BIOS performs Power On Self Test (POST) for all different hardware components in the system to make sure that everything is working properly.

 Also it checks, computer is being started from off position (cold boot) or restart(warm boot).

 Retrieves information from **CMOS** (**C**omplementary **M**etal-**O**xide **S**emiconductor) a battery operated memory chip on the motherboard that stores time, date, and critical system information.

 Once BIOS sees everything is fine, it will start searching for boot loader.

 It looks for boot loader in floppy, cd-rom, or hard drive. You can press a key (typically F12 of F2, but it depends on your system) during the BIOS startup to change the boot sequence.

 Once the boot loader program is detected and loaded into the memory, BIOS gives the control to it.

 So, in simple terms BIOS loads and executes the MBR boot loader.

**MBR (Master Boot Record):**

 MBR stands for Master Boot Record.

 It is located in the 1st sector of the bootable disk. Typically /dev/hda, or /dev/sda

 MBR is less than 512 bytes in size. This has three components 1) primary boot loader info in 1st 446 bytes 2) partition table info in next 64 bytes 3) mbr validation check in last 2 bytes.

 It contains information about GRUB (or LILO in old systems).

 So, in simple terms MBR loads and executes the GRUB boot loader.

**Boot Loader:**

 A boot loader, also called a boot manager, is a small program that places the [operating system](http://searchcio-midmarket.techtarget.com/definition/operating-system) (OS) of a computer into [memory](http://searchmobilecomputing.techtarget.com/definition/memory).

 GRUB stands for Grand Unified Boot loader:

o GRUB displays a splash screen, waits for few seconds, if you don’t enter anything, it loads the default kernel image as specified in the grub configuration file.

o GRUB has the knowledge of the file system (the older Linux loader LILO didn’t understand filessystem).

o Grub configuration file is /boot/grub/grub.conf (/etc/grub.conf is a link to this).

 LILO(Linux Loader):

o LILO is a linux boot loader which is too big to fit into single sector of 512-bytes.

o So it is divided into two parts: an installer and a runtime module.

o The installer module places the runtime module on MBR.The runtime module has the info about all operating systems installed.

o When the runtime module is executed it selects the operating system to load and transfers the control to kernel.

o LILO does not understand filesystems and boot images to be loaded and treats them as raw disk offsets

**Kernel:**

 Mounts Root file system.

 Initializes devices and loads initrd module.

 initrd stands for Initial RAM Disk.

 initrd is used by kernel as temporary root file system until kernel is booted and the real root file system is mounted. It also contains necessary drivers compiled inside, which helps it to access the hard drive partitions, and other hardware.

 init was the 1st program to be executed by Linux Kernel, it has the process id (PID) of 1.

**Init:**

 First process which is started in linux is init process.

 Looks at the /etc/inittab file to decide the Linux run level.

 Following are the available run levels

o 0 – halt

o 1 – Single user mode

o 2 – Multiuser, without NFS

o 3 – Full multiuser mode

o 4 – unused

o 5 – X11

o 6 – reboot

 Init identifies the default initlevel from /etc/inittab and uses that to load all appropriate program.

**Run Levels:**

 When the Linux system is booting up, you might see various services getting started. For example, it might say “starting sendmail …. OK”. Those are the runlevel programs, executed from the run level directory as defined by your run level.

 Depending on your default init level setting, the system will execute the programs from one of the following directories.

o Run level 0 – /etc/rc.d/rc0.d/

o Run level 1 – /etc/rc.d/rc1.d/

o Run level 2 – /etc/rc.d/rc2.d/

o Run level 3 – /etc/rc.d/rc3.d/

o Run level 4 – /etc/rc.d/rc4.d/

o Run level 5 – /etc/rc.d/rc5.d/

o Run level 6 – /etc/rc.d/rc6.d/

**What is the major difference between ext2 and ext3 file systems?**

The main difference between ext2 and ext3 is, ext3 allows journaling. (Journaling is a type of log file, which tracks all the file system changes. so that you can recover in case of filesystem crash)

**Explain Linux Booting Process?**

* When the computer is switched on, it automatically invokes BIOS [a ROM chip embedded in the motherboard].
* The BIOS will start the processor and perform a POST [power on self test] to check whether the connected device are ready to use and are working properly.
* Once the POST is completes BIOS will check for the booting device. The boot sector is always the first sector of the hard disk and BIOS will load the MBR into the memory. MBR holds the boot loader of the OS.
* From here the boot loader takes the control of the booting process.
* GRUB is the boot loader for Linux.
* Depending on the boot option selected the kernel is loaded first.
* After kernel is loaded the kernel will take the control of the booting process
* Initrd will be loaded which contains drivers to detect hardware (Initialization of RAM Disk)
* Then it will initialize all the hardware including I/O processors etc.
* Kernel then mounts the root partition as read-only
* INIT is loaded as the first process.
* INIT will mount the root partition and other partitions as read/write and checks for file system errors.
* Sets the System Clock, hostname etc..
* Based on the Runlevel, it will load the services and runs the startup scripts which are located in /etc/rcX.d/ (Network, cups, nfs, SSH etc.)
* Finally it runs the rc.local script.
* Now the login prompt will appear.

**How do you extend a LV? For example, How do you expand /var file system with additional 2GB space?**

* Check which logical volume (LV) holds the /var file system using df –h
* Now, find out this particular LV belongs to which VG using lvdisplay <lv\_name>
* Check the free space available in that Volume Group (VG) using vgdisplay <vgname>. Look for "**Free PE / Space"** line in the ouput.
* If the free space available in VG, now you can expand the LV using lvextend –L +2G <lv\_name>. Now, Logical Volume has been expanded. Now we have to expand the file system using **resize2fs /var.** All these can be done in online without unmounting the filesystems.

**How do you find out hardware errors inside Linux?**

* dmesg
* /var/log/messages
* dmidecode –t system
* IML (Integrated Management Logs) - An iLO console feature
* hpacucli - To check RAID array status
* use grep or less commands on
  + /var/log/messages and /var/log/warn
  + /var/log/debug
  + /var/log/kern.log
  + /var/log/mcelog

**How do find out what are the files inside an RPM before installing it?**

rpm -qlp package.rpm (Example : $ rpm -qlp rpm -qlp gnupg-1.4.5-1.i386.rpm)