**Explain in detail the Linux booting procedure**

* When a Linux [machine](javascript:void(0);) is powered on BIOS loads up first. It will prompt you to select boot device which can be Hard disk, CD-ROM, Floppy drive, Network etc. By default generally it will boot with hard disk
* Next comes your MBR. This will load and execute the GRUB boot loader menu. GRUB stands for Grand Unified Boot Loader. This will display the a splash screen with the contents of /boot/grub/grub.conf
* List of available and installed kernels will be shown, if not selected default kernel will be loaded
* Kernel: Mounts the root files system as specified by "root=" parameter inside /boot/grub/grub.conf file
* Next it will execute /sbin/init program which will boot the linux machine in the default run level as specified by /etc/inittab
* Runlevel: All the scripts loaded inside the selected runlevel from step 5 will be executed These scripts are placed inside /etc/rc.d/rcx.d/. Here x is the runlevel value which will be varying from 0-6
* Scripts starting from S would load at startup and those starting with K would kill the process at shutdown. These incident will take place as per the numerical value assigned to them. For eg: s13network will load prior than s15 sendmail
* Next your login screen will come up

**To know about Linux Boot process in more detail follow the below link**  
[Step by Step Linux Boot Process Explained In Detail](http://www.golinuxhub.com/2014/03/step-by-step-linux-boot-process.html)

### [Step by Step Linux Boot Process Explained In Detail](http://www.golinuxhub.com/2014/03/step-by-step-linux-boot-process.html)

Posted by Deepak Prasad Monday, March 24, 2014 [23 comments](http://www.golinuxhub.com/2014/03/step-by-step-linux-boot-process.html#comment-form)

You can find a number of articles explaining you the Linux Boot Process. Well I am not going to tell any thing different but I would like to show my point of perspective on some extra details of the **Linux Booting procedure.**  
  
*The stages involved in Linux Booting Process are:*  
**BIOS  
Boot Loader**  
  **- MBR  
    - GRUB**

[**Kernel**](javascript:void(0);) **Init  
Runlevel scripts**

### BIOS

* This is the first thing which loads once you power on your machine.
* When you press the power button of the machine, CPU looks out into ROM for further instruction.
* The ROM contains JUMP function in the form of instrucion which tells the CPU to bring up the BIOS
* BIOS determines all the list of bootable devices available in the system.
* Prompts to select bootable device which can be [Hard Disk](javascript:void(0);), CD/DVD-ROM, Floppy Drive, USB [Flash Memory Stick](javascript:void(0);) etc (optional)
* [Operating System](javascript:void(0);) tries to boot from Hard Disk where the MBR contains primary boot loader.

### Boot Loader

To be very brief this phase includes loading of the boot loader (MBR and GRUB/LILO) into memory to bring up the kernel.  
  
**MBR (Master Boot Record)**

* It is the first sector of the Hard Disk with a size of 512 bytes.
* The first 434 - 446 bytes are the primary boot loader, 64 bytes for partition table and 6 bytes for MBR validation timestamp.

**NOTE:** Now MBR directly cannot load the kernel as it is unaware of the filesystem concept and requires a boot loader with file system driver for each supported file systems, so that they can be understood and accessed by the boot loader itself.  
  
To overcome this situation GRUB is used with the details of the filesystem in /boot/grub.conf and file system [drivers](javascript:void(0);)  
  
**GRUB (GRand Unified Boot loader)**  
This loads the kernel in 3 stages  
  
GRUB stage 1:

* The primary boot loader takes up less than 512 bytes of disk space in the MBR - too small a space to contain the instructions necessary to load a complex operating system.
* Instead the primary boot loader performs the function of loading either the stage 1.5 or stage 2 boot loader.

GRUB Stage 1.5:

* Stage 1 can load the stage 2 directly, but it is normally set up to load the stage 1.5.
* This can happen when the /boot partition is situated beyond the 1024 cylinder head of the hard drive.
* GRUB Stage 1.5 is located in the first 30 KB of Hard Disk immediately after MBR and before the first partition.
* This space is utilized to store file system drivers and modules.
* This enabled stage 1.5 to load stage 2 to load from any known location on the file system i.e. /boot/grub

GRUB Stage 2:

* This is responsible for loading kernel from /boot/grub/grub.conf and any other modules needed
* Loads a GUI interface i.e. splash image located at /grub/splash.xpm.gz with list of available kernels where you can manually select the kernel or else after the **default timeout** value the selected kernel will boot

The original file is /etc/grub.conf of which you can observe a symlink file at /boot/grub/grub.conf  
  
Sample /boot/grub/grub.conf  
default=0  
timeout=5  
splashimage=(hd0,0)/grub/splash.xpm.gz  
hiddenmenu  
title Red Hat Enterprise Linux Server (2.6.18-194.26.1.el5)  
        [root](javascript:void(0);) (hd0,0)  
        kernel /vmlinuz-2.6.18-194.26.1.el5 ro root=/dev/VolGroup00/root clocksource=acpi\_pm divisor=10  
        initrd /initrd-2.6.18-194.26.1.el5.img  
title Red Hat Enterprise Linux Server (2.6.18-194.11.4.el5)  
        root (hd0,0)  
        kernel /vmlinuz-2.6.18-194.11.4.el5 ro root=/dev/VolGroup00/root clocksource=acpi\_pm divisor=10  
        initrd /initrd-2.6.18-194.11.4.el5.img  
title Red Hat Enterprise Linux Server (2.6.18-194.11.3.el5)  
        root (hd0,0)  
        kernel /vmlinuz-2.6.18-194.11.3.el5 ro root=/dev/VolGroup00/root clocksource=acpi\_pm divisor=10  
        initrd /initrd-2.6.18-194.11.3.el5.img  
  
**For more information on GRUB and LILO follow the below link**  
[What is GRUB Boot Loader ?](http://www.golinuxhub.com/2014/03/what-is-grub-boot-loader.html)

### Kernel

This can be considered the heart of operating system responsible for handling all system processes.  
  
Kernel is loaded in the following stages:

1. Kernel as soon as it is loaded configures hardware and memory allocated to the system.
2. Next it **uncompresses** the initrd image (compressed using **zlib** into  zImage or bzImage formats) and mounts it and loads all the necessary drivers.
3. Loading and unloading of kernel modules is done with the help of programs like insmod, and rmmod present in the initrd image.
4. Looks out for hard disk types be it a LVM or RAID.
5. **Unmounts** initrd image and frees up all the memory occupied by the disk image.
6. Then kernel mounts the **root partition** as specified in grub.conf as **read-only**.
7. Next it runs the **init** process

**For more information on kernel follow the below link**  
[What is a Kernel in Linux?](http://www.golinuxhub.com/2014/01/what-is-kernel-in-linux.html)

### Init Process

* Executes the system to boot into the run level as specified in /etc/inittab

**Sample output** defining the default boot runlevel inside /etc/inittab  
# Default runlevel. The runlevels used by RHS are:  
#   0 - halt (Do NOT set initdefault to this)  
#   1 - Single user mode  
#   2 - Multiuser, without NFS (The same as 3, if you do not have networking)  
#   3 - Full multiuser mode  
#   4 - unused  
#   5 - X11  
#   6 - reboot (Do NOT set initdefault to this)  
#  
id:**5**:initdefault:  
As per above O/P system will boot into **runlevel 5**  
  
You can check current runlevel details of your system using below command on the terminal  
# who -r  
         run-level 3  Jan 28 23:29                   last=S

* Next as per the **fstab** entry file system's integrity is checked and **root partition is re-mounted** as read-write (earlier it was mounted as read-only).

### Runlevel scripts

A no. of runlevel scripts are defined inside /etc/rc.d/rcx.d  
Runlevel Directory  
0 /etc/rc.d/rc0.d  
1 /etc/rc.d/rc1.d  
2 /etc/rc.d/rc2.d  
3 /etc/rc.d/rc3.d  
4 /etc/rc.d/rc4.d  
5 /etc/rc.d/rc5.d  
6 /etc/rc.d/rc6.d

* Based on the selected runlevel, the init process then executes startup scripts located in subdirectories of the /etc/rc.d directory.
* Scripts used for runlevels 0 to 6 are located in subdirectories /etc/rc.d/rc0.d through /etc/rc.d/rc6.d, respectively.
* **For more details on scripts inside /etc/rc.d follow the below link**  
  [What are the s and k scripts in the etc rcx.d directories](http://www.golinuxhub.com/2014/01/what-are-s-and-k-scripts-in-etc-rcxd.html)
* Lastly, init runs whatever it finds in /etc/rc.d/rc.local (regardless of run level). rc.local is rather special in that it is executed every time that you change run levels.

**NOTE:** rc.local is not used in all the distros as for example Debian.

Next if everything goes fine you should be able to see the **Login Screen** on your system

**How many commands do you know which can be**[**used to**](javascript:void(0);)**view the contents of any file?**

* less
* more
* tail
* head
* vi or vim
* cat

**For more information on the usage of above commands follow the below link**

[7 Commands to read or view the contents of a file using CLI in Linux](http://www.golinuxhub.com/2013/09/7-commands-to-read-or-view-contents-of.html)

**What is the difference between soft link and hard link?**

***Soft Link***

1. Using this only a link to the original file is created (shortcut).
2. The size of created shortcut is null.
3. If you delete the file then the created link (shortcut) won't work.
4. In case you delete the shortcut link then it won't affect the original file

***Hard Link***

1. Another copy of the file is created.
2. Both the file have same inode no.
3. Any changes made in either of the file will appear on the other file.
4. Deleting any of the one file won't affect the other file.

**How to find the bit size of your linux machine?**

# uname -m  
i686  
  
# uname -m  
x86\_64  
If you get i386, i586 and i686 in your reply than that signifies your machine is 32-bit but if you get x86\_64 or ia64 then your machine will be 64-bit.  
# getconf LONG\_BIT  
32  
  
# getconf LONG\_BIT  
64Here you will get an output of bit size either 32 or 64.  
  
**How can you add a banner or login message in Linux?**  
By editing these two files  
/etc/issue  
/etc/motd  
  
**For more details follow below link**  
[Add a Banner to your Linux machine](http://www.golinuxhub.com/2012/08/add-banner-to-your-linux-machine.html)  
  
**How will you check the release version of your Linux machine?**  
# cat /etc/redhat-release  
**What is the difference between normal kernel and kernel-PAE?**  
kernel in 32 bit machine supports max of 4 [GB RAM](javascript:void(0);) whereas

kernel PAE in 32 bit linux machine supports till 64 GB RAM  
  
**Tell me the command to find all the commands in your linux machine having only 2 words like ls, cp, cd etc.**  
# find /bin /sbin/usr/bin /usr/sbin -name ?? -type f  
**Which file is generally used to configure kickstart?**  
anaconda.cfg  
  
**What is the command use to**[**compress**](javascript:void(0);)**a dir using [gzip](javascript:void(0);) compression?**  
# tar -czvf myfil.tar.gzip orig\_file  
**What is the command use to compress a dir using bzip2 compression?**  
# tar -cjvf myfil.tar.bzip2 orig\_file  
**Which log file will you**[**check for**](javascript:void(0);)**all authentication related messages?**  
/var/log/secure  
  
**What is the command to create multiple directories using one command?**  
Using -p argument along with mkdir command  
  
**Which command would you prefer to monitor your system performance and why?**  
I personally prefer top and sar command to view the performance related factors of my Linux machine.  
  
The top program provides a dynamic real-time view of a running system. It can display system summary information as well as a list of tasks currently being managed by the Linux kernel.  
  
Also sar gives us a standard o/p of the running activity performance related factors in relation to paging, [NIC](javascript:void(0);), CPU, Memory etc  
  
**For more information on top and sar command follow the below links**

[10 examples to help you understand top command usage in Unix/Linux](http://www.golinuxhub.com/2014/03/8-examples-to-help-you-understand-top.html)

[Tutorial for Monitoring Tools SAR and KSAR with examples in Linux](http://www.golinuxhub.com/2014/02/tutorial-for-monitoring-tools-sar-and.html)  
  
**What is the command used to find the process responsible for a particular running file?**  
# fuser [filename](javascript:void(0);)  
# lsof filename  
**What is the command to take remote of any Linux machine?**  
# rdekstop  
**What are the three values shown in load average section of top command?**  
It shows the current, 5 min back and 15 min back load average value  
  
**How to check all the process running by a particular user?**  
# ps -u username  
**What is an orphan process?**  
An orphan process is a process that is still executing, but whose parent has died.  
  
**What is a defunct process?**  
These are also termed as zombie process. These are those process who have completed their execution but still has an entry in the process table. When a process ends, all of the memory and resources associated with it are de-allocated so they can be used by other processes.After the zombie is removed, its process identifier (PID) and entry in the process table can then be reused.  
  
Zombies can be identified in the output from the Unix ps command by the presence of a "Z" in the "STAT" column  
  
**What is the command in sar to monitor NIC devices received/transmitted packets?**  
# sar -n DEV 1 5

This will show 5 consecutive output each with a time interval of 1 sec for all the ethernet devices  
  
**To learn more about SAR command follow the below link**  
[Tutorial for Monitoring Tools SAR and KSAR with examples in Linux](http://www.golinuxhub.com/2014/02/tutorial-for-monitoring-tools-sar-and.html)