1. **Booting procedure :-**
2. **File system and Partition :-**

* **RAID**
* **LVM**
* **Disk Quota**

1. **Process Management :-**

* **To clear RAM memory cache, buffer :-**
* **Job**
* **System Monitoring :**
* **vmstat**
* **sar**
* **netstat**
* **lsof**

1. **Job scheduling :-**
2. **User Administration**

* **PAM**

1. **S/W management 1) rpm 2) yum**
2. **TCP wrappers, Limiting access to sudo**
3. **NFS**
4. **Samba**
5. **ssh or slogin : 1)**

**Installation procedure ??**

- Partition

- Boot loader

- Network configuration

- Firewall configuration

- Time zone

- Root account configuration

|  |  |  |
| --- | --- | --- |
| **Minimum Partition creation and sizes for basic installation**  **Partition Name** | **Size For 32 Bit** | **Size For 64 Bit** |
| / (root) | 8 to 10 GB | 15 to 20 GB |
| /boot | 200/100 MB | 200 MB |
| SWAP | Twice of RAM | Twice of RAM |

1. Enter into **BIOS** setting and make CD/DVD Drive as first boot device

* It will ask for **Install or upgrade an existing system** and install using GUI mode.
* It will ask to **begin testing media,** we have to **skip** the testing and then **select language.**
* It will ask type of devices like **basic or specialized** storage devices. Then **select basic.**
* Then assign hostname, if wants to give **IP** address select **configure network.**
* Then assign **root password.**

B) Will ask type of **partition**, then to create our own partition select **custom layout**

**-** It will prompt **standard, RAID, LVM** partition. We have to **select standard partition**. Then create mount point **/(root), /boot**, and file system **swap**.

- Then **format the partition** and select write changes to disk.

C) Install **boot loader**

- select **desktop for GUI** env. And select **customize later** for further customize the s/w selection.

- After successfully installed then **reboot. Updates** available for proper functioning of system and installation of these updates is recommended after reboot.

- Firewall : If want to enable security then enable or else disable it.

- SElinux : disable it

- **Create a normal user** account for regular use of system

- select **date and time**

**[hostname /directory]$halt / $shutdown –h : -** logout and shut down the system

**$shutdown :** shut down and reboot the system

1. **Booting procedure :-**

when system boots, BIOS start and normally load GRUB from hard drives, form first **446** bytes of disk.

When machine is booted, the kernel is loaded from disk into memory. Kernel remains in memory until machine off. When machine is turned off the kernel stored on the machine’s hard disks.

When we run command, kernel loads from machine’s hard disk into memory.

After machine on, system **looks for all peripheral** and then goes through series of steps

* **six** stages of Linux boot process **:-**

1. **BIOS :**- **Basic Input/output system**

* Perform system integrity checks
* Search, loads, execute bootloader program. It looks for bootloader in flopy, CD-Rom, Hard drive. Press a key during BIOS change booting sequence
* Once bootloader detected and loaded from hard drive into memory, BIOS gives control to it.

1. **MBR :- Master Boot Record**

* First sector of disk is reserved for MBR, in that first **446** bytes contain **bootloader** that start the system, followed by **64** bytes which contain **partition table**. mbr validation checks in last 2 bytes

1. **GRUB :- Grand Unified Bootloader**

**Bootloader** : A program that loads operating system kernel into memory and execute it.

In linux, **grub** is the bootloader of which configuration file is **/boot/grub/grub.conf**

* It loads default kernel images as specified in grub conf. file into memory

Grub loads and execute kernel and initrd(**Initial RAM DISK**) images.

**Initrd** is used by the kernel as **temporary root F.S.** until kernel booted and real F.S. is mounted.

* We have to specify where the OS kernel is located, which hard drive and what partition on that hard drive. Info **root (hd0,0)** means 1st hard drive(hda) and the first partition.

1. **Kernel :-**

* Mount root F.S. as specified in “root= “ grub.conf
* Executes **/sbin/init** program

Init is the first program executed by kernel, it has PID **1**

1. **Init :-**

* Looks at **/etc/inittab** file to decide the default **run level(**state of a system that determine which services are available**)**
* **0 - halt**
* **1 - single user mode(**use for trouble shoot, also, press **E** while booting to go in also, Use **s** or **S** to enter single user mode with startup scripts run**)**

allowing access only to the superuser, and does not run any network services also without running startup scripts for services like DNS.

We can use **Linux -s** at the LILO prompt to enter run level 1

* **2 - Multiuser,** without NFS or **xinetd**(Don’t have network services to connect another server)
* **3 - Full multiuser** and remote file sharing access **(**Have network connection hence, used as a default, **not GUI** only command line interface**)**

We can use **#start** to start GUI

* **4 - Unused**
* **5 - X11 (**GUI, multiuser, network)
* **6 - shut down and reboot**

Note :- The default runlevel read from **/etc/inittab (id : 5 : initdefault:)** Also, we can **change** default run level by making entry in this file and then **reboot** the system.

We can use **init** or **telinit** to change the runlevel. However, we should not use to go from higher to lower runlevel. Also, we can use **Linuxconf** or **Webmin** to change the default startup runlevel

**Shutdown** command bring the system in **single user** mode. **shutdown** command gives users on the system a warning or can specify an exact time.

The **–h(halt)** option, shuts down and go in **runlevel 0**, whereas the **-r** option shuts down the system and then reboots it and go in **runlevel 6**.

$ **who –r** or **runlevel** to know in which runlevel the system are. **runlevel** will list the previous state followed by the current one. Ex : **N 3** indicating no previous state

1. **Runlevel program :-**

The state of a system that defines which services are available

System will execute the run level program **/etc/rc.d** based on our default init run level Ex : **Run level 3 - /etc/rc.d/rc3.d**

**/etc/rc.d/rc.sysinit** : Hold commands for Initialization of system

**/etc/rc.d/rc.local :** we can place our own commands. This is the last initialization file excecuted

**/etc/rc.d/init.d :** Holds daemons that both start up and shut down.

**Note** :- When shutdown, the **halt** file, which contain the commands to do is called. The files in **init.d** are called to shutdown daemons, and F.S are unmounted.

* **Login process :-**

When we login first time, two(login & passwd) prompts are displayed that presented by a program **getty.** Give these two prompt to the **login** command and exit. After, login looks through **/etc/passwd** for entry matching. If it match, login executes a shell & exit. If no match found, login issue a error maessage & exit.

When the login program executes shell, that shell is **uninitialized.** The shell undergoes initialization that involves the shell reading /**etc/profiles**  & **.profile.**

As soon as both files have been read, shell display a prompt **$**

**--------------------------------------------------------------------------------------------------------------------------Recover lost passwd and corrupt bootloader ???**

**--------------------------------------------------------------------------------------------------------------------------**

1. **File system and Partition :-**

Hard disk are normally divided into partition. So, that different parts if it can be formatted with different F.S. or used for different purposes. If we place data in two separate F.S. in two separate partition, If one partition containing home area gets filled, another partition may have space.

If more partition needed, one of **primary** partition can be converted into **extended**, which is divided into multiple **Logical**  [**P**(3) **- > E**(1) **-> L**(11)(many)]

Note :- We are limited not more than **15** partition, **14** usable for F.S.

* First partition on the disk will be **MBR** (contain **IPL**[Initial program loader] and **PTI**[Partition table info.])
* **Primary Partition:** Holds the **operating system.** Only one amongst 3 can be **active** which will be booted by MBR to load the O.S.
* **Extended Partition:** After creating 3 primary partition allocate remaining space as an extended partition. So, we able to create logical partition. There can be only **one extended partition** per disk. This partition **can’t be formatted with F.S**. because they contain other partition.

**Note** :- We need to **reboot the system** after creating Extended partition and before logical partition.

* **Logical Partition :** Created under Extended partition
* **Superblock:-**

**Procedure to create partition:-**

* Conform if disk has been added to the server, we can know it by

**#dmesg | tail (or)**

**#fdisk –l**

2) To view existing partition :

$ **fdisk –l** or **parted –l**

* **Creating new partition**

$ **fdisk**  <**disk name**>

* Use  **p** to list out the partition information
* Use **n** to create new partition [use **d** to delete partition]
* Use **w** to save the partition changes
* Update partition table without restarting the system and updates the kernel table

$ **partprobe**  /dev/sda or **partx -a** /dev/sda

Note > Partition ID : **83** normal partion, **82** swap, **8e** LV2

* 2) **Format partition with File system :-**

It is method of storing the data in an organized fashion on the disk.

**# mkfs <F.S. type> <Partition name>\**

To format Linux **ext3** partitions, you can use **mke2fs** **-j** instead of mkfs

Where, j stand for journaling filesystem which allow system to recover from crash

**Ext2 :** Doesn’t maintain journal. Slow recovery & no guarantee

**ext3 :** Is a journaling which maintain records in its journal. Fast recovery.

means don’t have to fsck after a crash. Max file size **16 GB** to **2 TB.**

**Ext4 :** Max file size **16 GB** to **16 TB**

**#mke2fs -t ext4 /dev/DEV**

Note :- we can convert ext2 F.S. to ext3 using **#tune2fs –j /dev/<block-device-name>**

* **3) Mounting a partition :-**

In order to **use F.S**. or to **access partition**, we need to associate it with mount point. An empty directory which is **already exist**. A mount point is a directory where the files on that partition are connected to the overall Linux file structure for your system.

🡪 Temporary mounting

# mkdir directory(empty directory)

# **mount <partition> <directory(mount pt)>**

When not sure of the type of file system also, to read windows & linux disk

#**mount –t auto /dev /mnt**

**🡪 Permanent mounting** : make entry in **/etc/fstab** file or use **Linuxconf**:

This file show what partition should have their F.S., mounted on what mount point, with which option. Keep info. Of all permanent mounted F.S.

**<Partition name> <mount point> <F.S. type> <defaults> <dump> <fsck>**

**Defaults :** Specify that device is **R/W**, it is asynchronous (**async**), it is a block device (**dev**), that it cannot be mounted by ordinary users (**nouser**), and that programs can be executed on it (**exec**).

**Default, noauto :** means F.S. not automatic mounted whereas, **auto** means **mount -a** command executed when the system boots, in effect, mounts file systems automatically

**Dump :** backing up the F.S.

**Fsck :** If a F.S. should be checked at reboot time. 1 means a **boot** partition & 2 means **other** partion, 0 means fsck needn’t check the F.S.

**Note :-** Once drive is formatted then add entry into fstab with **UUID** We can get UUID from

**#blkid /dev/sd^** (^ replicates respective partition)

**Note :**- If fstab file corrupt, then system will boot into a maintenance mode, only read access to partition. To gain R/W access, have to remont main partition.

**# mount -n -o remount, rw /**

**Ex : /dev/sda/ /data ext4 default 1 2**

**/etc/mtab** : stores information of all currently mounted F.S. It is dynamic and keep changing.

**Note** :- - To see file system is in use **$fuser –cu /home** and check for files which are open

**lsof /home** and then kill the open connections using **fuser -ck File/Dir**

* To see the size of File/Dir use **du –h File/Dir.**
* To see the usage info. of mounted partition use **df -h**

**Note :-** we can resize the size of a partition using # **parted /dev/sda print**

**# resize 3(minor no.) 1024 2048**

**------------------------------------------------**

**Swap partition :-** The part of disk used for temporary data storage

Swap space is used when amount of physical memory (**RAM**) **is full** or to **extend a system memory.** Swap space is located on **hard drive,** which have a slower access time than physical memory. It active automatically at **boot time**.

**Size :** RAM is less than or equal to(**<=**) **2**GB, then SWAP=**2x RAM**

RAM is more than(**>** ) 2GB, then SWAP= **2GB + RAM**

When we need to increase the swap space, we create additional swap space which will be added to the existing swap space to increase the size.

**Swap space creation** :

The swap partition should be the recommended minimum size of 64 Megs.

Method 1 :- If we have an additional hard disk

* Create normal partition and change **hex code 82** (use **t** in fdisk)for swap
* Format partition with swap file system

# **mkswap /dev/sda**

* Enable swap partition for usage

**# swapon /dev/sda**

* Make enrty in **fstab** file

**/dev/sda swap swap defaults 0 0**

Method2 :- If we don’t have any additional disk, we can create a file somewhere on filesystem, and use that file for swap space

Create a blank file of 100 MB

# **touch /root/myswapfile**

**# dd if=/dev/zero of=/root/myswapfile bs=1M count=100(size)**

To create swap file of size 4GB #**dd if=/dev/zero of=/swapfile bs=1024 counts=4**

**# mkswap /swapfile 8**

Note : permission should be **600,** only root can access

* Format this file with swap file system **# mkswap /root/myswapfile**
* Enable swap file **#swapon /root/myswapfile**
* Make entry in fstab file to keep it on after reboot

Note :- If we don’t want to reboot to check whether system take all swap partition or not in fstab file, then disable and enable all swap partition in fstab file.

**# swapoff -a**

**# swapon -a**

To see memory size and swap space size use **#free –m**

To see the swap usage # **swapon –s** or **cat /proc/swaps**

**--------------------------------------------------------------------------------------------------------------------------------------**

* **RAID(Redundant array of independent devices) :** Is a method of storing and retrieves data across several disk treated as one disk to provide greater efficiency and redundancy. Allows flexibility in adding or removing hard disk. Some of the disks are used as real-time mirrors, duplicating data.

RAID device is called an **md** device because it uses the MD driver. RAID devices are an array of hard disk partition, where each partition contains an entire hard disk to achieve redundancy, increase bandwidth and ability to recover data from hard disk crash. RAID distributes data across each hard drive in the array by breaking it into small **chunks**. Status of RAID listed in the **/proc/mdstat.**

Note: - Before use RAID, it should be implement on kernel. If not, have to reconfigure and install new version of kernel.

RAID can implemented at different level

**> Software RAID** is implemented under **OS Kernel level**. The Linux kernel contains an **MD driver** that allows the RAID to be completely hardware independent. The performance of a software-based array depends on the server CPU performance and load.

> The **hardware**-based RAID is **independent from the host**. A Hardware RAID device **connects to the SCS**I controller and appears to the host as a single disk.

**RAID 0 (Stripping):** For efficiency RAID stores data using **disk stripping**, where data is organized into standardized strips that can be stored across RAID devices for **faster access**. Grouping 2 or more hard disks into a single unit with total size equal to all disks used. Breaking data into fragments and write simultaneously on all disk which improve read/write performance.

**Advtg :** It **boost** **I/o** performance because 2 or more disk R/W single piece of data at a time

**Disadvtg :** No single disk contain entire info. This means if one disk fails, the entire RAID is inoperable, with unrecoverable loss of data.

**RAID 1 (Mirroring):** Redundancy is implemented using **mirroring,** the same data is written to each RAID drive in exactly same fashion, create identical clone, so that if one fails, other have data.

**Advtg :** Redundancy is improved, if only one disk is functional normal operation of system can be maintained

**Disadvtg :** Reduced write performance, as data has to be written n times.

**RAID 5 :** Data can be reconstructed using **parity information** which takes up the space equivalent

to one drive, leaving most of the space on the RAID drive free for storage.

Minimum 3 devices used, 2 or more devices configured in a RAID 0 setup, while 3rd device is a parity device. If one of RAID 0 device malfunction, the array will continue operating, using parity device as a backup.

**Advtg :** Improves write performance and redundancy

* **Creating RAID** :-

Then use **mkraid /dev/md0** command to create RAID devices

# ***mdadm --create /dev/md0 --level=5 --raid-devices=3 /dev/sdb1 /dev/sdc1 /dev/sdd1***

For create RAID devices on same hard disk, need to create the hard disk partition specify **fd** as F.S type and then as above configure RAID devices, format it, mount it and make entry in fstab file.

**mke2fs -j /dev/md0**

mkdir /data

**mount /dev/md0 /data**

To verify RAID partition

#**mdadm --detail /dev/md0** Or use **cat /proc/mdstat**

Add device into array **# mdadm /dev/md0 –add /dev/sdX**

To fail #**mdadm --fail /dev/md0 /dev/sdX**

Note :- To change RAID conf. or remove RAID partition, first have to un-mount and then stop **/md0**

**#umount /data**

**# mdadm --stop /dev/md0**  and

remove it **#mdadm --remove /dev/md0 /dev/sdX**

And then remove entry from **/etc/fstab** file and delete all the partition.

**---------------------------------------------------------------------------------------------------------------------------------**

* **Logical Volume Management (LVM) :**
* Only one F.S. can exist on disk. We can’t create F.S. larger that disk space.
* If there is short of space on disk, it is not possible to extend F.S. If we allocate more space than need, that space goes waste, there is no way to take it back.

**Defn** : Raw disk space in physical disk partition can be converted into **virtual partition** called LVM

* Ability to extend existing F.S. with more space without need to reformat it.
* we can resize volumes if you’re out of disk space  
  LVM provide more flexible way to manage disk space than physical disk partition
* It create **virtual** partition, allowing to grow, shrink or move those partition from drive to drive

**Physical volume (PV):**  A partition marked for LVM on MBR, with partition type **0x8e.** PV is a standard primary or logical partition

Physical partitions are known as **physical extents(PE)** and its default size is 4 MB**.** PV is divided into no. of equal sized PE

**Volume group (VG):** Group of one or more physical volume

**Logical volume (LV):** virtual partition of volume group formatted with F.S. and mount any F.S and used like partition. The size can easily increase/decreased as per requirement.

Logical partitions are known as **Logical Extents(LE)**

LAB Work :- - Create partition and change hex code to **8e**

* Create a PV **#pvcreate /dev/sda7(partition name)**
* Create a VG #**vgcreate ktvg(VG name) /dev/sda7(partition name)**
* Create a LV #**lvcreate -L <size of LV> -n <LV name> <VG name>**

**# lvcreate -L 300M -n ktlv ktvg**

* Adding F.S. to LV.

In order to make it **accessible** need to format it **#mkfs ext4 /dev/ktvg/ktlv**

* mounting it and make entry in fstab file for permanent mount

**# mount /dev/ktvg/ktlv /data/0**

* **Note :-** Maximum size of LV are -

For **2.4** kernel = **2 TB**

For **2.6** kernel 32-bit CPU = **16 TB**

For 2.6 kernel 64-bit CPU = **8 EB**

**\*** To create LVM snapshot **#vcreate -<size> -s –n <snapshot name> <LV name>**

* **Extending a volume group :**

It means adding new disk(PV) to the volume group. Need to create new partition and create a PV

**#vgextend <vg name> <partition> # vgextend ktvg /dev/sda8**

* **Increasing size of LV :**

**# lvextend/lvresize -L <+size> </dev/vg/lv> #lvextend –L +200M /dev/ktvg/ktlv**

**Update** F.S by **resize2fs** command **#resize2fs /dev/ktvg/ktlv**

**Case** 1 > Extending a logical volume if there are still unallocated extents in the volume group

First check free PV in volume group by using **#vgs** Once, we see there free space that can be used to extend.

Then extend **#lvextend –l +100%FREE /dev/sysvol-root**

**Case 2**> Extending a LV when there is no unallocated extents in the volume group. In this case we need to extend volume group first.

**Reducing LV :**

* LV size can’t be reduced online, it requires a down time i.e. **unmounting** the F.S.

# **umount /data/0**

* **Organized** the data before reducing LV **# e2fsck -f /dev/ktvg/ktlv**
* **Update** the F.S. about size i.e what size will be after reduction

**# resize2fs /dev/ktvg/ktlv 300M (**this is the total size of LV after reduction**)**

* Finally reduce the size **# lvreduce -L -200 /dev/ktvg/ktlv**
* Mount the LV
* **Move/Migrate LV from one PV to another :**
* Unmount the F.S. on that PV # **umount /data/0**
* Add new PV(create new partition **/dev/sda8**) which should be of same size or higher than replacing PV to the volume group **# vgextend ktvg /dev/sda8**
* Migrate PV’s contents to the new PV # **pvmove <old PV> <New PV>**
* Mount back the LV **#mount /dev/ktvg/ktlv /data/0**
* Remove faulty PV from volume group **# vgreduce ktvg /dev/sda7**
* **Deleting/Removing LV :**
* Unmount the F.S. & Remove entry from **fstab** file.
* **#lvremove /dev/ktvg/ktlv**
* **Deleting a volume group :**
* To delete the vg, there is no LV in it, it shouldn’t be mounted. It will also remove LV inside it.
* **#vgremove ktvg**
* **Deleting PV :**
* Deleting PV shouldn’t belong to any VG. We can delete PV which is **free**
* **#pvremove /dev/sda{6,7}**
* **Creating VG by specifying PE size** :
* Create partition and also create a PV
* **#vgcreate VG name -s <size of PE> <PV names>**

**#vgcreate ktvg2 -s 16 /dev/sda5**

* **Create LV of 400MB**
* Size of LE  **==**  size of PE i.e. **16**
* To know how many LE is required : **size of LV/size of PE** 400/16= 25 LE required
* **#lvcreate -l <no. of Lv> -n <Lv name> <VG name> #lvcreate -l 25 -n ktlv2 ktvg2**
* **Disk Quota :-** Disk Quota is used to limit a user’s or group’s consumption of disk space.

Quotas can only be created for partitions

Quota is of two types, user and group

**Soft Limit** : The user will notify when is in quota violation. Setting grace period(7 days), will act as an alarm, also need to set a hard limit. A grace period is the number of days a user is allowed to be above the given quota.

**Hard limit** : Hard limits are necessary only when using grace periods. Any attempt to consume resources beyond this limit will be denied. If you are not using grace periods, the soft limit is the maximum amount of available space for each user.

Quota can be applied to users and groups block size and inode number.

**Grace period :** For the soft limit, you can designate a grace period during which time the user has the chance to reduce his or her disk space below the limit. If the disk space still exceeds the limit after the grace period expires, then the user can be denied access to his or her account

**Steps to apply Disk quota** :-

1. Install quota packages
2. Edit /**etc/fstab** file as

/dev/hda7 /home ext3 defaults,**usrquota,grpquota** 0 0

1. Either **reboot** or **remount** the F.S. to enable quota

**# mount -0 remount,usrquota,grpquota,rw /home** and verify using **#mount | grep /home**

1. To scan /home, enable quota or create a quota file **aquota.user** under /home

**# quotacheck -cvugm /home**

1. Enable/switch quota on /home **#quotaon -avug**
2. Add quota soft and hard limits to any user **#edquota -u sbhuyar**

Note : This opens a temp file under /tmp, also –**t** option edit soft limits for FS

* To edit grace period # **edquota -t**
* If user wants to view his quota **$ quota** and for root user **#repquota -a**

**Step to disable quota:-**

* **# quotaoff -avug**
* Clear entry in /etc/fstab
* Remove the partition

------------------------------------------------------------------------------------------------------------------------------------------

1. **Process Management :**

n **instance** of running program

There are 3 types of process -:

1. **Interactive process** :- Invoked by a user and can interact with user and classified into foreground and background. The foreground process **interacting with user**, and using terminal as its stdin and stdout. The background process is **not interacting with user**.
2. **System process or Daemon** -: Don’t interact with console/user and it is run in the background. We can set the process run in the background by plcing **&** at the end of command. **Not associate with any terminal.**

Daemon listen at certain port for user request and then wake up when they receive input.

Note : Pause (**ctrl+z**) and kill(**ctrl+c**)

A process is either in user or kernel state. A process use temporary space for storing data called **stack.** Kernel maintain data structure known as **process table** which keep process attributes **owner & group, start time, priority.**

The total no. of process that can stored in the process table is determined by **nproc** variable

First process started at boot time is **init** and has **PID** is **1** is the parent of all process in the system. **Swapper** process is created at system startup and has **PID** of **0**.

When parent process dies while its children are still active, init becomes its parent.

Init process runs at nice value of **20.** PIDs are in the range **1...65535**

To start the server only when a request for its services is received, configure using **xinetd** daemon. Stopping xinetd shuts down all the server that this daemon manage.

To restart –.

# /etc/rc.d/init.d/xinetd restart

# service xinetd restart

By sending SIGHUP signal # **kill –HUP PID**

Logs are managed by the **syslogd** daemon.

* **Process state :-**

1. **Running :** when its text is being executes.
2. **Sleeping :** when resource or event that is currently not available.
3. **Ready to run :**when kernel wakes up a sleeping process, it’s become runnable.
4. **Stopped/suspend(ctrl+z) : stop** signal puts a process in stopped state. After receive **continue** signal stopped process goes into ready to run state.
5. **Zombie :** Process whose parent don’t wait for their death and remain in this state until parent receive child’s exit status. We **can’t kill** zombie process need to **reboot** the system.
6. **Orphan :** **Parent dies** before child and init becomes parent

**Note :-** Only one process active on CPU at a time.

* **Listing process:**

**# ps** option : list running process of the present terminal

**-a :** Total no. of process also, process of **all users**

**-u :** Process by **logged in** user

**-x :** To see which process attached with terminal and which are not.

**?** sign it **TTy** column shows **daemon(**not associated with terminal**)**

**-e :** All including **user** and **system**

* **Sending signal :**

O.S. communicate with kernel through signal. It is an interrupt that communicate some information. There are total **64** signal, the list can be seen by **kill –l**

**Kill** command is used to send signal to process. If no signal is specified, it send **SIGTERM**(15)bydefault.

Syntax: **kill** [-signal] process ID

**SIGHUP(1)**  Reloading the process

**SIGKILL(9)** kill process

**SIGCHILD(18**) child process terminate/stopped

**SIGSTOP(24**)

* To kill last background job kill **$!**
* To kill Login shell kill -9 **$$**
* Login shell PID **echo $$**
* Kill all process of name oracle #**killall -9 oracle**
* **Nice Value :-**

We can set **guideline** for the CPU to follow when it is looking at all the task it has to do called **nice value** which has range from **-20(**higher priority**) to 19.**

The default nice value is **zero** and **init** runs at nice value **20.** Every process inherit its nice value from parent.

**Note :-** **user** can adjust this value **down** as far as +19, but can’t increase it. However, **root can increase the priority** as high as -20

Schedular divide CPU time into time slices, in which each process get a turn to run on CPU, but we can **affect the priority** by setting Nice value for a process.

**nice** command is used to change the value of **new process** and **renice** is used to change the value of **running process**.

* To **list** the nice value use **# ps -efl**
* To **set** a nice value of **new** process **# nice -n <range> <command>**
* To change nice value of running process # **renice 2 PID**
* **top command** :- To monitor all process that are utilizing more CPU in the system

**1st** line : **uptime, no. of user logged, load avg(1, 5, 15 min)**

Load avg means no. of process in the ready to run in 1,5,15 min.

**2nd** line : No. of process and their current state

**3rd** line : **CPU** utilization for **user(**%**us)** process, **system**(%**sy**) process, **available** CPU(%**id**), CPU **waiting**(%**wa**)

**4th** line : **RAM** Memory

**5th** line : **SWAP** memory

Top field :

**PID, USER, PR, NI :** process ID, User ID, priority, Nice value

**VIRT :** virtual size of task

**RES :** size of **RAM** currently consumed by task

**SHR :** some memory area shared between **two or more** process. Ex shared library and SysV share memory

**S :** Task status

**%CPU, % MEM :** % of CPU and RAM consumed by the task

**TIME+ :** total **CPU time** used

* **Job :-** List process in the background and stopped process associated with the shell prompt. We can put job in background by placing ‘**&’** at the end of command.

When we execute any command in the background, a job no. and system process no. are displayed. Ex. $lpr mydata &

[1] 456

$ [1] + Running lpr mydata

**Note :**- ‘**+**’ indicate current background job

**‘--**‘ indicate next job to be executed

**fg %1** command send job to the foreground and only one process can run in F.G**.**

**kill %1** or **kill 456** To stop a job running in the background

to place running foreground job into the background. First need to suspend(ctrl-z), and then place in background with **bg** comm

> **Restricts user Process:-**

By using **/etc/security/limits.conf** file we can restricts the no. of processes a user/Group can generate.

**root soft nproc 40000**

**root hard nproc 50000**

user root can create max 5000 processes and get a warning on 4000

* **Services :**

Services are **daemons** that once started run in the background and listen on particular port and respond when it receive a request.

To enable or check the service

1. **Service daemon [status|start|stop|restart|reload]**
2. **# /etc/init.d/daemon [start|stop|restart|status]**

This directory store scripts files

1. **Kill –HUP PID** will reload the process
2. Systemctl (used in graphical mode)
3. **#ps –aef | grep service**

root 1916 1 0 Jan26 ? 00:00:15 /usr/sbin/sshd

Note :- Above **?** sign indicate that service not associated with any terminal and it is running in the background

**Reload** the services, may be required after doing some change in config file

**Restart** any service, required when reload does not work

To enable service so that it should automatically start when server boot or active in appropriate runlevels

# **chkconfig - -list | grep service** on which run level the service is available even after reboot.

# **chkconfig service on/off**

* > To make the service available on only particular run level

**#chkconfig - - level <1-6> <service> <on/off>**

* **To clear RAM memory cache, buffer :-**

can drop cache without rebooting the system. It will slow the system for second and every resources required by o.s. is loaded again in the disk cache.

1. clear page cache (disk cache)

**# sync ; echo 1> /proc/sys/vm/drop-caches**

1. Clear dentries and inodes

**#sync ; echo 2> /proc/sys/vm/drop-caches**

1. clear page cache, dentries and inodes

**#sync ; echo 3> /proc/sys/vm/drop-caches**

* **sync** will flush the F.S**. buffer**

Also, it redirect data from buffer to Disk

**drop-cache** will clean **cache** without killing any application/service

**echo** is doing the job of **writing to file**

* It is not good idea to clear RAM cache on production server. Most common way to free up Ram is **rebooting.**
* **Monitor virtual memory and OOM :-**

There are 3 things to measuring memory of system

* How much physical memory in use **#free –g** or **top**
* How much virtual memory in use **#free –t**
* See rate of memory pages moved from physical to disk **#vmstat (swap in & swap out)**

**When system is low memory, it will swap out a lot of blocks.** If this happen at a high rate, it will impact on performance.

In case of physical memory, If used memory from total memory is high then it is OK, So long as there is free virtual memory.

If SWAP space usage is high that can be dangerous, then if program needs more memory and is unable to get it then i.e OOM. Killer start kills processes.

* **System Monitering :**

A slow system due to slow **CPU,** less **memory,** congested **network.**

It need **sysstat** package install

* **Vmstat :** Used to moniter activity of virtual memory. Listing detailed about performance of system componants.

Disply o/p in **6** column

**1)procs - - 2)memory - -3)swap - - 4)i/o - - 5)system - - cpu**

**r b swapd free buff cache si so bi bo in cs us sy id wa st**

**r >** No. of process waiting for access to processor

**b >** No. of process in a **“sleep”** state or busy process.

**swapd >** How much memory swapped out to a swap file or disk

**free >** Amount of unallocated memory

**buff >** amount of allocated memory in use

**cache >** Amount of allocated memory that could be swapped to disk

**si >** Amount of memorymoved from swap to “real memory”

**so >** Amount of memory moved to swap from real memory“

**bi >** No. of blockes received from a disk

**bo >** No. of blocks sent to a disk.

**in >** No. of system interrupt

**cs >** No. of context switches

* **sar :** system activity report
* To monitor system over a period of time
* Shows no. of kernel parameters and CPU utilization
* It can log the system activity data in a disk file
* It provides overview of system which includes processor, memory, I/O devices and n/w info.

Overall CPU statistics 5 times every 1second **#sar –u 1 5**

How much **memory** server using **#sar -r 1 2**

To view **swap** space usage **#sar –s**

Monitering **I/O** activity **#sar -b**

\* To display Run queue and load avg **sar –q**

CPU usage of individual core or cpu **#sar –p ALL**

To report each and every activity related to diff devices attached to the system **#sar -d**

* To fetch data of a particular day and for specific time

**#sar -f /var/log/sa/sa27 -s 2.20 –e 3.20**

Sar runs through cron **/etc/cron.d/systat**

* **Netstat :** To list all network connection on a system. It list tcp,udp connection, also list listening port
* To list all current connections **$ netstst -an**
* **lsof :** List of open files by which process or belong to all active process
* when a disk can’t be unmounted as it says the files are being in use. We can easily identify the files which are in use. **# lsof /home** or **fuser /home**

**command PID USER IO TYPE DEVICE SIZE?OFF NODE Name**

* List of all opened files of specific user

**# lsof -u username**

Find processes running on specific port(22) # **lsof –i TCP:22**

* List of all n/w connections #**lsof -I**
* List processes which listening on particular port **#lsof –i:25**
* Search by PID **#lsof -p 1**
* Kill all process for specific user **#kill -9 `lsof -t -u sagar`** where ‘t’ for list only PID of process
* List process which opened a specific file **#lsof filename**
* List opened files under a directory **#lsof +D /directory**
* list opened files based on process name **#lsof -c ssh**

-------------------------------------------------------------------------------------------------------

1. **Job Scheduling :-**

Schedule a task that is done on a regular basis. Task can be scheduled using **cron** which run as a **daemon** and is started at **runlevel** **2** boot time.

* Types of job :-

1. **at :** schedule task that is run **only once**

creates file that are **removed** after execution of program

1. **crontab :** schedule task that is run **reapetedly**

create file that are read and execute according to the schedule contained within file and execute regular

cron **read** crontab and at file created in **/var/spool/cron** adding the name of the user who issued the command at startup. Ex: **/var/spool/cron/sbhuyar**

Root user can control use of crontab command by regular user with **/etc/cron.allow** file.

cron daemon creates **log** file in **/var/adm/cron**

Startup **script** for cron daemon is **/sbin/init.d/cron**

* **crontab format -:**

**/etc/crontab** file stored all scheduled jobs

**Min hour day month day of weak command**

**(0-59) (0-23) (1-31) (1-12) (0-6)[Sunday=0/7]**

Command :-

Crontab **-e** **Edit** crontab file this open crontab file in /var/spool/cron

**-l** **display**

**-r remove**

**-u** combine with above for **user**

Options :

All possible value **\***

1-5 for the day-of-week field specifies Monday through Friday

Ex:-

1. To display current date for every 5 min on console

# **crontab -e** & enter field below and save it as in **vi**

**\*/5 \* \* \* \* date > /dev/pts/1**

**:wq!**

**# service crond restart**

1. To create directory under /root on “Sunday 22 oct at 1:30 am”

# **crontab -e 30 1 22 10 0 or 7 mkdir /root/dir1**

1. To run the backup script “bkpscript” on every “Saturday 12:30 pm”

30 12 \* \* 6 bkpscript

1. User “sagar” should get a mail regarding meeting on 24th, 29th and 31st oct. at 2:25 pm

**Crontab -e -u sagar**

**25 14 24,29,31 10 \* echo “Meeting”**

1. User “sagar” should get the mail from 15 to 20 and 25 to 30 Nov as a reminder of some session at 2:25 PM

# **crontab –e -u sagar**

**25 14 15-20, 25-30 11 \* echo “ meeting”**

1. backs up the **projects** directory at 2:00 A.M. every weekday:

0 2 \* \* 1-5 tar –cf /home/bckp /home/projects

* **At jobs** :-

Syntax : **# at <time>**

**Task**

**Ctrl + d** to save it

Ex : To get mail at 11:30 AM regarding meeting

**# at 11.30am**

**echo “meeting”**

* To check the list # **at –l**
* To check what is schedule  **# at -c <job id>**
* To remove a job **# atrm <job id>**

**5) User Management :**

To login as root user

**#** su –

Note - To get root’s environments and command paths and home directory use – hyphen sign with su commands.

**> Login Access:** We can control user login access to system with **/etc/login.access** file. The file

Consists of entries listing users, whether they are allowed access, and from where they can

access the system.

Ex: **+/-**:username:remote host/terminal

We can list more than one user or location by using **ALL EXCEPT** option.

> **Lock & Unlock user account after failed login** :-

- To Lock user account after 3 failed attempts :

Open **/etc/pam.d/system-auth** file

Now write this line “**auth required pam\_tally2.so onerr=fail deny=3”** just above the line

auth sufficient pam\_unix.so in the file.

- To unlock :

“**unlock\_time=100”** It will unlock the account after 100 sec.

\* To manually unlock the locked account use **#pam\_tally2 –r –u username** command

> **To disable user account #passwd bhuyars –l**

**To re-enable account # passwd bhuyars -u**

------------------------------------------------------------------------------------------------------------------------------------------

1. **S/W management 1) rpm 2) yum**

1) RPM(Redhat package manager) : For installing, verifying, updating s/w packages. s/w can be updated using RPM without having to reinstall then

- To check all installed packages in system.

# **rpm –qa** (q for query, a for all)

# **rpm -qi** info. Of installed packages

to remove/uninstall package

# **rpm -e** <package>

- To install package

First mount DVD(ISO image)

# **mount –t iso9660 –o ro /dev/cdrom /mnt**

First we need to be in cd /mnt/packages # rpm –ivh <package>

- To check packages of command

$ which command

#**rpm –qf /bin/cat**

- To see conf. file of installed package

# **rpm –qlc**(file)/d(dir) <package>

- To update package

#**rpm –Uvh** <package>

- If getting error “package is already installed” but, want to install package any how

# rpm -ivh –**replacepkgs** test-1.0

2) **YUM**(yellowdog Updater modified): Package management application for Linux. Packages are downloaded from repository.

Yum server will automatically resolve dependency of rpm package while installing them

1) First mount DVD on /mnt

# mount /dev/dvd /mnt OR # mount –t iso9660 –o ro /dev/cdrom /mnt

2) Copy entire DVD to /var/www/html/rpms with its defaults permission

# mkdir /var/www/html/rpms

# cd /mnt/packages

# cp –rvfp \*.rpm /var/www/html/rpms

Note :- /var should require minimum 4 GB space

3) Install below packages

#rpm -ivh deltarpm-

#rpm -ivh libxml2-python-

#rpm -ivh python-deltarpm-

#rpm -ivh createrepo-

4) createrepo –v /var/www/html/rpms : it will create index file and maintain dependancy

5) Create a repo file which will be functioning as repository address and conf. file

# cd /etc/yum.repos.d/

mkdir repos

mv CentOS\* repos/

vi local.repo

[local] => Shortname given to repo

name = local yum repo => Complete name for repo.

baseurl=file:///var/www/html/rpms

gpgcheck=0

enabled=1

5) Clean yum cache and check package list

# yum clean all

#yum list

If output displayed the server successfully configured.

- > On **Client side** :

# cd /etc/yum.repos.d

# mkdir repos

# mv CentOS\* repos/

# vi local.repo

[local]

name = local yum repo

baseurl=file:///var/www/html/rpms OR http://IP of server/html/rpms

gpgcheck=0

enabled=1

yum clean all

1. **TCP Wrappers :-**

Add another level of security to xinetd-managed servers. The server is wrapped with an level of security, monitoring connection, controlling access by remote host and verifying remote user identities. That monitor and restricts access to some service.

Connections are logged with the **syslog** daemon and found in **/var/log/secure** file. List of hosts are kept in **host.allow** and **host.deny** files and have the format **service:hostname:domain**.

If same hosts is listed in both files, then host.allow takes precedence and access is permitted.

Ex : **ftp:All** Allow access to all hosts to web service,http

* **Limiting access to sudo :-**

Sudo command allows users listed in **/etc/sudoers** to run administrator commands. A user with appropriate right can execute root commands with giving own password. To access this file in vi editor use **visudo.**

**Root ALL=(ALL) ALL** => root allowed full access with password

**% wheel ALL=(ALL) NOPASSWD:ALL=>** People in wheel group to run all commands with no password

**8) NFS : Network File System**

- Export/share File system to other machine (Linux<-> Unix) and client system mount one or more of the share directories to local directories called mount points. After the share is mounted, all I/O operations are written back to the server

**-**To mount remote host File system to our machine

- we can provide centralized storage

- It uses standard client/server architecture for File sharing between linux based machine

**>** Profile for NFS**:-**

**Packages**  nfs-utils, nfs-utilis-lib

**Port no.** 2049

**Conf. file** /etc/exports

**Daemons**  rpc.nfsd, rpc.mountd, rpc.statd

Steps:-

: On **server** side :

1. Install NFS package : 2 ways to install packages rpm, yum
2. Create a directory/partition and mount it and add data in it also, make entry in **fstab**.
3. Export the directory by editing **/etc/export** file

**/Directory(777) IP (rw,sync)** 🡪 Ip range for client machine

**Note :-** Directory should have all perm.

**\*(rw, sync)**🡪 To share all network

**\*** mount option –

Sync : All changes must be written to disk before command complete

No- delay : Forces the writing of changes immediately

Root-squash: Prevent root user

**# exportfs –avr** : To exports the directory

**a :** Export/unexport all directory

**r :** Reexport all directory

4) Restart service #**service nfs restart**

To start the service when system boot # **chkconfig nfs on**

\* To check which directory is exported for machine or to list all shared F.S for the system

#**showmount –e IP** : show only F.S. and IP

#**showmount –v** : what shared, on which client, what optioned used.

\* Check the directories which is exported in **/var/lib/nfs/etab or rmtab**

**: Client side** :

1. Install package
2. Start the NFS services and make it permanent
3. Check which directory is exported for machine

**# showmount -e <server IP addrss>**

1. Make directory and mount NFS over it.

**Mount –t nfs IP of server:/Dir /new Dir**

1. To make permanent entry in **/etc/fstab**

**<IP of server>:/Dir /mount Dir nfs defaults 0 0**

**\* Auto-mounting NFS directory :-**

1. **SAMBA :-**

* Implementation of a **common internet file system(CIFS)** protocol that can be run on almost every variant of Unix in existence. Microsoft will use this protocol to access files located on UNIX box.
* SAMBAs allows Linux computers to share files across a network
* Samba allows Linux computer to share files and directories across a network.

**Profile :-**

**Package** : samba, samba-common, samba-client

**Daemons** : smbd, nmbd **Port no :** 187, 138,139,445

**Config file :** /etc/samba/smb.conf

**Steps :**-

First check whether window system is connect to linux m/c or not using **ping.**

1. Install packages : using yum or rpm
2. Make directory(/test) and assign 777 perm
3. Check the **context** of directory and change it according to samba also, check with #**ls -ldz**

# **chcon -t samba\_share\_t /test**

1. Create or use existing user who allowed to login as samba user

* For new user # **useradd** and then # **passwd**
* For existing user # **smbpasswd -a <username>**

To delete a user -**x**

* To check all samba user **#pdbedit -L**

**Note :-** Samba user and Linux user are same

1. Open conf. file **/etc/samba/smb.conf** and copy last **7** lines and paste it at the end

**Comment** =public suff

**Path** = **/test :** share directory

**Public** = yes/no : public access(every user in n/w)

**Valid user** = username : Authorized user

**Writable** = yes : Permission

**host allow** = n/w range

1. Restart service and make it enable after reboot

# **service smb/nmb restart** and # **chkconfig smb/nmb on**

**Windows as a client :**

1. **Open computer** -> Map network drive

Give IP of samba server, it will prompt for username and passwd

1. Go to **window+r** => //IP of linux then ask for user and passwd

**Linux as a client :**

First, the file/data being shared need to share from windows

* To check how many samba server on Linux client m/c **# findsmb**
* To check **share name** of that samba server # **smbclient –L //IP of server**

To connect samba server # **smbclient -U user IP(**windows**)/data**

To mount samba dir. on client

**# mount -t cifs //IP of server/share name /Dir -o user=username**

**DNS :**

How it works:

* When type kt.com in browswr, browser sends a query over the internet to match the domain name with its IP address.
* The query interact with **recursive resolver**, which can be operated by **ISP(Internet service provider).** The ISP knows which DNS servers it needs to ask to answer query
* ISP interact with **root server** for **DNS information** about .com**.** The root servers are running all over the world and each one knows DNS info. about **top level domain** such as .com
* TLD servers answers with the **IP address of the** **Domain’s name server**
* Next, ISP sends the query to the Domain name server. DNS server knows the IP address for the full domain
* ISP tells the browser what the IP address is. Finally, browser send a request to the website to retrieve the website’s content.

**-----------------------------------------------------------------------------------------------------------------------------**

1. **SSH or slogin:**

when user wants to connect to network remotely such connections could be moniter and info. Such as login ID & password could be copied and used later to break in. Access a shell(command line) and execution of command on remote system. It can **encrypts** any communication between the remote user and a system and uses host key(public key) encryption for authentication purpose only..

SSH designed to replace remote access operations, such as **rlogin, rcp,** and **telnet, ftp.**

Using **sftp** we can transfer FTP files secured by encryption.

Ssh conf. file is /**etc/ssh/sshd\_config** and daemon is **sshd.** Port no. is **22**

First needs to **create** the public(**id\_dsa.pub**) and private key(**id\_dsa**) along with a passphrase to use for the authentication process. Public key is used to identify a user and its host.

Note:- version1 > public(**identity.pub**) and private key(**identity**)

Version2 > public(id\_dsa.pub) and private (id\_dsa)

1. Create ssh keys of type DSA/RSA using **ssh\_keygen** with **version 2** and it will **prompts for passphrase** only once per session, which use as a password to protect private key. It will create keys in **.ssh** of home directory

Note: - If need to change passphrase, use **–p** with ssh\_keygen command

1. Then **sends** public key to the **remote** account **~/.ssh/authorized\_keys2** they want to access and private key should be kept **secret** and **secure**.

To access remote account, will have know remote account’s SSH passphrase.

Local system 🡪 private key(600) **id\_dsa**

Remote system 🡪 public key(644) **id\_dsa.pub**

**#ssh-copy-id -i /root/.ssh/id\_rsa.pub** [**root@w.x.com**](mailto:root@w.x.com)**(IP)**

Or **scp**

1. Login to remote server and open file and make below changes

**vim /etc/ssh/sshd\_config**  
 PasswordAuthentication **yes**

A public key is used to identify a user and it’s host. If user wants to log in remotely from local account, first place their public key in the **.ssh/authorized\_keys** file on remote system.

Move to client system and check whether the key is copied or not.

**# cd /root/.ssh**

**# cat authorized\_keys** which will hold all the system which are authorized and will not be asked for Password.

Give perm. to the file **# chmod  600  ~/.ssh/authorized\_keys**

Restart ssh service to reflect the changes # **/etc/init.d/ssh restart**

If we login using ssh, it will ask us for passphrase.

If we exit the client session and login again, it will not ask us for passphrase.

# **ssh w.x.com -l username**

* **Remote file transfer :**

**> scp :** copy files across an ssh connections that will be encrypted, and secured.

* To copy a file to remote machine from source location (A > B), while logged in A

# **scp /Path of A user@B: /<location to copy file>**

* To copy a file to remote machine from source location (A > B), while logged in B

**# scp /Path of B user@A: /<location to copy file**

* To copy a file from remote machine to source location (B > A), while logged in A

**# scp user@B:/location /Path of A**

* To copy a file from remote machine to source location (B > A), while logged in B

# **scp /Path of B user@A:/Path**

Note:- Password will ask if public key is not saved on both location.

# **scp –r :** To copy a directory

**> rsync :** for backing up/mirroring a directory tree. It’s speed up file transfer by copying the difference between two files rather than copying an entire file every time. It will copy the updated files/directories rather than copying all files/dir, which saves times and speedup the transfer.

**# rsync <option> <encryption> <source file> <Dest. IP>:/Location**

**# rsync – rv -e ssh /ktdir IP:/root**

**-avz** To compress the data and send it in archive mode

**> Assign IP to server** : Numerical representation of a host address

4 Conf. parameters are required

1. IP 2) Netmask 3) Default Gateway 4) DNS

* Static conf : Assign manually, used with server
* Dynamic conf. : By DHCP , configure n/w part on the boot
* Network conf. tool:

Command line tool used to conf. IP on terminal

# system-config-network

* Configure IP through conf. file

Go to /etc/sysconfig/network-scripts/ eth0 file

BOOTPROTO= static

ON BOOT = yes : specify whether the device is activated during the boot process

IP ADDRSS=

NETMASK

Then # service network restart

Or # systemctl restart network