# Storage Testing Tool

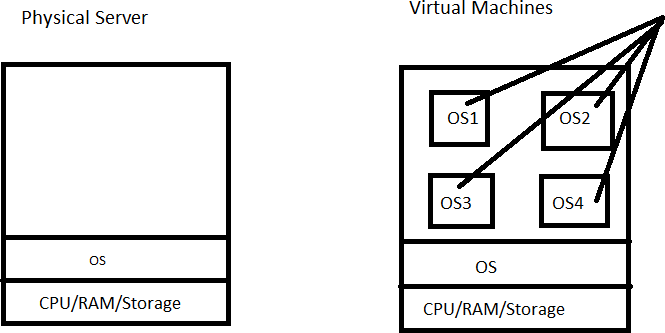
**\*\*\* Virtualization \*\*\***

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

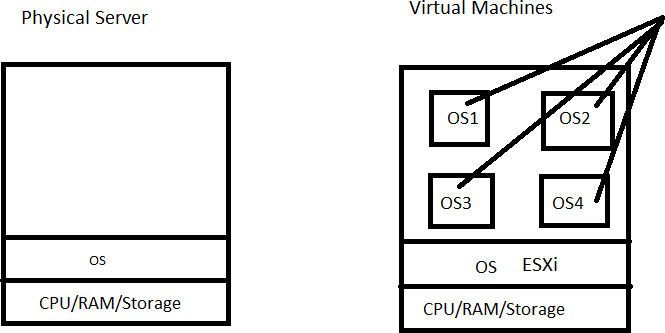
##### Virtualization ::

Laptop > 16GB, 1TB

Physical Laptop :: Virtualization :: multiple virtual machines [Acts as a single server] 16GB :: VM1 vm2 VM3

4GB 4GB 4GB

Vmware

Esxi [Elastic Sky X ] :: OS

##### Physical Server ::

It’s a high end version of laptop which serves the client requests Physical Resources:

1. RAM :: 16 GB
2. Storage :: 1TB
3. CPU
4. Virtualization

**Virtualization :**

Virtualization is a software technology, which shows physical resources into virtual resources. Physical Resources, First introduced by IBM in 1960’s

1. CPU
2. Memory
3. Storage
4. Network

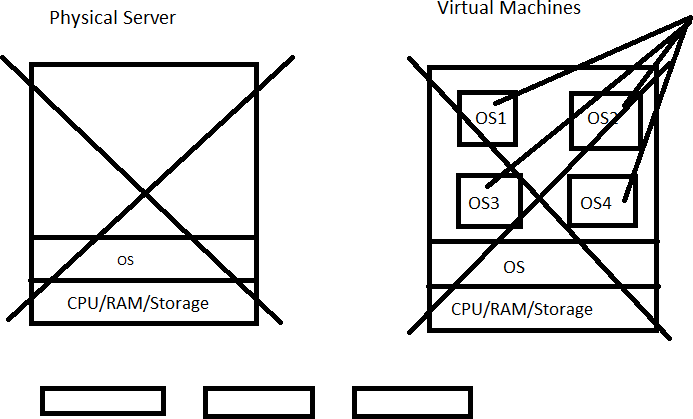
It is a technology which makes us to install and run multiple operating systems [Guest Operating System] and applications on same physical server at a same time.

##### Advantages :

1. Effective use of the server resources [Server consolidation]
2. The maintenance cost of data centers will come down
3. Power usage is less
4. No need of more engineers for maintenance

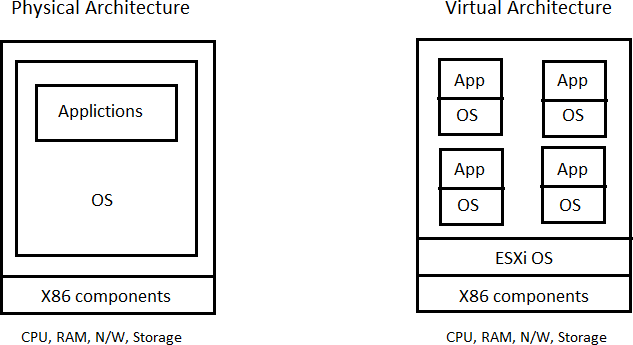
##### Disadvantages :

1. The major disadvantage of virtualization is if any physical server fails all the virtual machines which are running top of that Physical Server also will be gone



##### Types of Virtualization :

1. Server Virtualization
2. Storage Virtualization



##### Different types of servers :

1. Tower Server
2. Rack Server
3. Blade Server

Tower Server



Rack Server



Blade Server



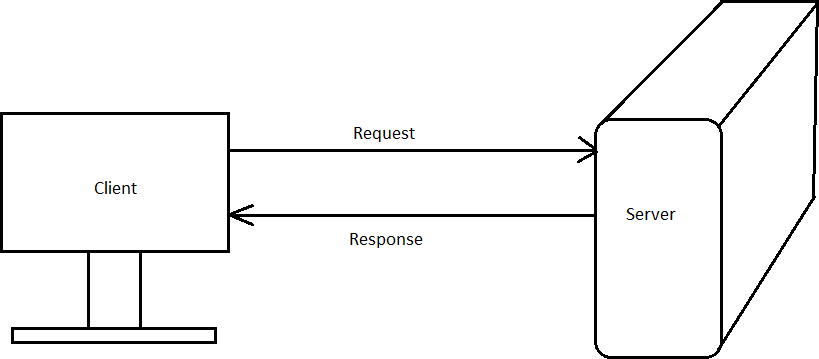
##### Client and Server Architecture ::

###### Client ::

Client is person, who sends the request to server

###### Server ::

Server is the one, which serves the client request



###### ESXI ::

ESXI stands for **Elastic Sky X Integrated**

ESXI is a enterprise operating system from VMware

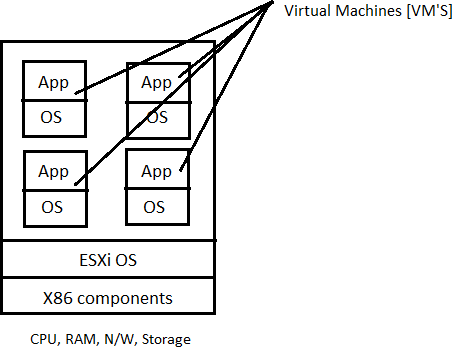
The current version of ESXI is 7.0 U3 [Release 3] [Jan 2022]

ESXI is a host operating system that we can install on top of severs It is a server operating system

###### What is the VM ::

**VM stands for Virtual Machine**

1. VM is a container where we can install and run Guest Operating system
2. VM is a Equivalent to the Physical sever
3. Physical sever shares the resources with Virtual machine
4. VM is a hardware independent
5. Easy to maintain
6. We can create up to 300 VM’s per 1 Esxi Host [Depends on the resource availability ]



**Guest Operating System**::

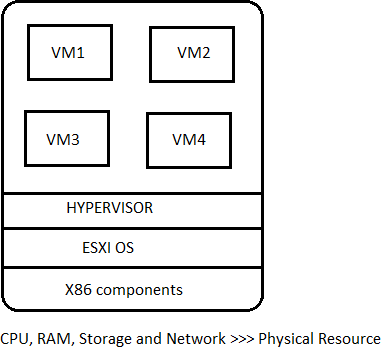
The operating system which we install on VM’s is called as guest operating system

Ex : Windows, Linux

###### Write difference between Physical server and Virtual Machine ::

**Hypervisor::**

Hypervisor is a Virtualization layer which helps us to create VM’s and run multiple Operating system on top of single operating system [ESXI]



There are two types of hypervisor are available

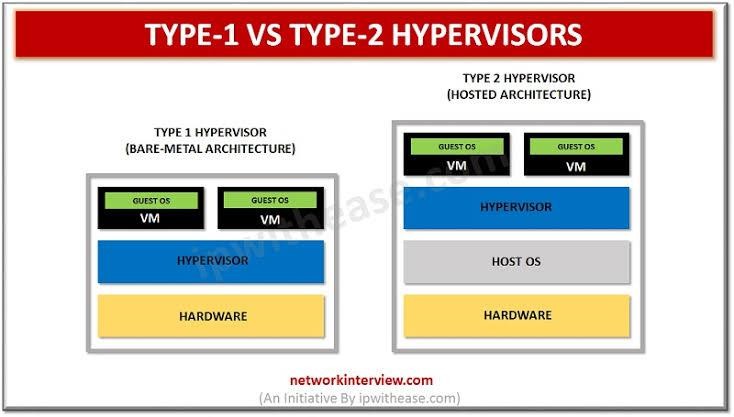
1. Type I Hypervisor
2. Type II Hypervisor

**Type I Hypervisor**::

It can be run directly on top of Bare Metal Server [Hardware]

**Type II Hypervisor**::

It can be run on top of Host Operating system [ESXI]



**BareMetal Server:**

The Server which is having no operating system is called as Bare-metal Server

###### ESXI Host = Physical Server

**How to Manage a ESXI Host Steps to create a Virtual Machine**

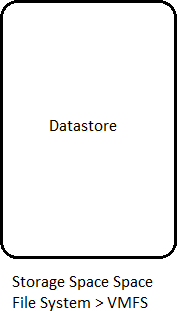
1. Login to vSphere Clint
2. Click on Inventory > Host
3. Select “Create New Virtual Machine”
4. Select Typical or Custom mode
5. Click on Next
6. Select VM name and Location
7. Select Storage [Data Storage]
8. Select Vm Version [Vm10]
9. Select Guest Operating System
10. Select CPU
11. Select Memory
12. Select Network
13. Select SCSI Controller [Optional]
14. Select Storage
15. Click on Create
16. Verify the progress in progress bar

After Creating a VM, it will be in power off state. We need to power on the VM to use

###### State of VM ::

1. Power-off
2. Power-on
3. Suspend
4. Reset
5. Shutdown
6. Restart

###### Data Store::

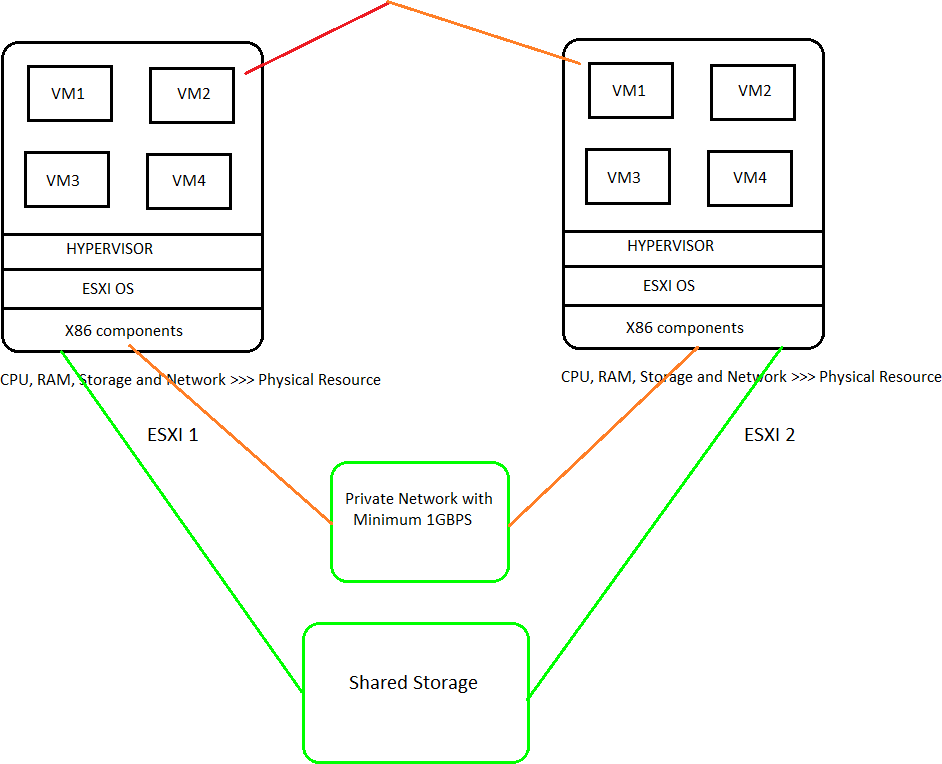
The storage space which is formatted with VMFS [VM File system] is called as Data Store. Data Store stores virtual machine files, ISO images, configuration files & Log files

###### Features of a VM

1. VMotion
2. SV Motion
3. XV Motion

**Vmotion**

Powered on VM’s Migrating from one ESXI Host to the another ESXI Host



**Requirements :**

1. Shared network and shared storage
2. Same CPU in the source and destination ESXI Host
3. Vmkernal Port Group must be configured

###### Why Vmotion ::

1. Hardware maintenance
2. To balance the load on VM’s

###### How to perform V Motion

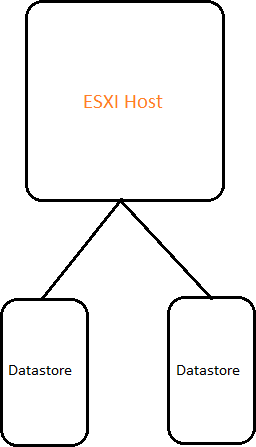
1. Login to Vsphere / Vecntre
2. Select VM
3. Right Click
4. Select Migration type

###### Vmkernal Port group

1. It helps VM’s to communicate with outside world

###### SV Motion [Storage V Motion]

Powered on VM files migrating from one Datastore [ESXI] to another Datastore [ESXI] is called as SV Motion



###### Requirements

1. Multiple Datastores must be connected to same ESXI host
2. Datastore needs enough space for migration

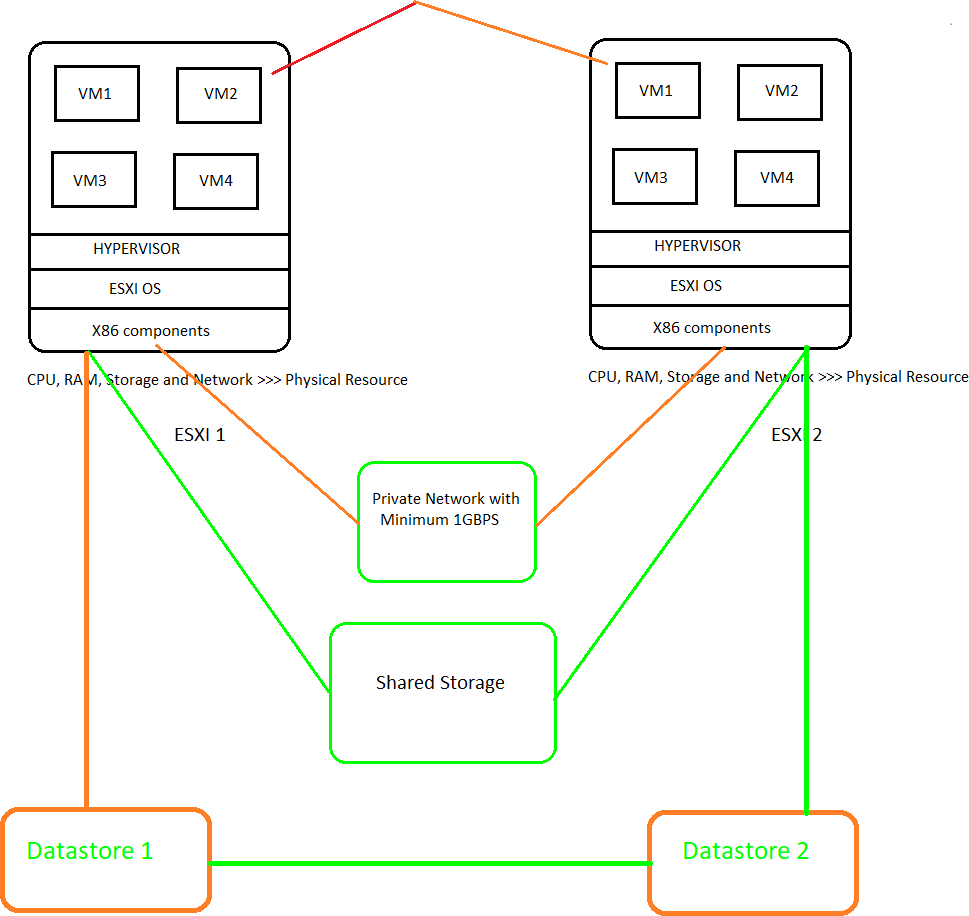
###### How to perform SV Motion

1. Login to Vsphere/Vcentre
2. Select VM
3. Right click
4. Select Migration Type [Select only storage/Files]

###### XV Motion::

Powered-on VM’s migrating from one ESXI host to another ESXI host and Files migrating from one data store to another data store

XV motion = Vmotion + SV Motion



###### Requirements

1. Shared network and shared storage
2. Same CPU in the source and destination ESXI Host
3. Vmkernal Port Group must be configured
4. Multiple Datastores must be connected to same ESXI host
5. Datastore needs enough space for migration

vMotion, Svmotion [Storage vMotion], and Cross vMotion (Xvmotion) are features provided by VMware for live

migration of virtual machines (VMs) between ESXi hosts or datastores within a VMware vSphere cluster. These . operations are performed at the cluster level, not at the individual VM level.

**Data Centre** :: Group of clusters will sit in Data Centres

**Cluster** :: Group of ESXI Hosts

* When we create clusters by adding multiple ESXI Hosts, the ESXI Host resources will become part of cluster resources. After esxi 8.0 we can manage cluster from vsphere only

**ESXI Host** :: Single Physical server with ESXI OS installed

## \*\*\*\*\* Vcentre ::

It is VMware enterprise Application used to manage multiple ESXI Hosts [Clusters]

**Virtual Machine Files**

After creating VM

1. .VMX :: Configuration file
2. .vmdk :: Datastore related info
3. Vmware.log :: all the event logs

**Features of Vcentre / Cluster**

1. **HA [High Availability] ………Enabled in cluster level**
2. **FT [Fault tolerance] ……….Enabled in VmLevel**
3. **DRS [Distributed Resource Scheduler]…….Enabled in Cluster level**

**HA [High Availability] ::**

HA stands for High availability

Whenever the ESXI host fails, The VM’s which are running on ESXI host will be restarted in another ESXI host based on resource availability

1. HA should be enabled in cluster level
2. When HA is enabled in cluster level, Vcentre installs the HA agent called as FDM [Fault Domain Manager] on all ESXI hosts within the cluster
3. These HA Agents [FDM’s] will be in a continuous sync with each other and exchanges heart beats
4. When one HA agent stops sending the heart beat, other HA agent understands that other ESXI host got failed and starts restarting VM’s on working ESXI host
5. There will be a downtime in HA as VM’s need to restarted on other ESXI host

###### Requirements :

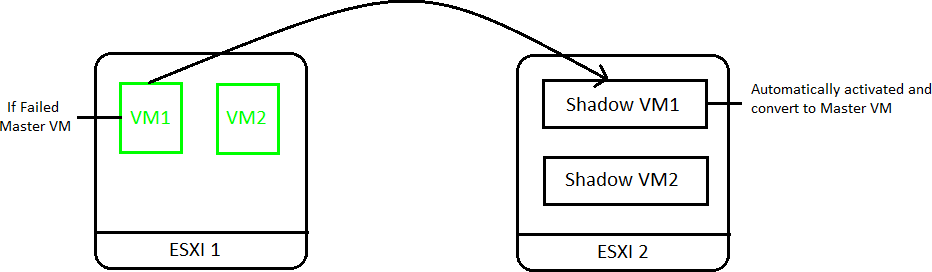
1. There will be enough resources in the cluster
2. HA must be enabled in cluster level
3. At least 2 ESXI hosts must be there in cluster

###### Disadvantage :

1. There will be a downtime for VM’s

###### FT :: [Fault Tolerance ]

* FT stands for Fault Tolerance
* It avoids the complete downtime that is there in HA
* FT can be enabled on VM level
* We need to power-of the VM to enable FT
* After enabling FT if VM is power-on, the shadow VM will be automatically created on other ESXI host
* In case of Master VM fails, shadow VM becomes master VM and again shadow VM will be created for that particular VM
* The shadow VM will be in inactive state



###### How to enable FT :

* 1. Right click on VM
  2. Select Fault Tolerance
  3. Click on “Turn on Fault Tolerance”

###### Advantages::

1. No downtime in FT

###### Requirements:

1. Shared storage
2. Vm kernel port group must be configured
3. Same CPU on source and destination ESXI hosts

###### DRS [Distributed Resource Scheduler] ::

* + DRS stands for Distributed Resource Scheduler
  + It is a algorithm applied on group of ESXI hosts [Cluster]
  + Whenever there is a load in balance on ESXI host, the VM’s which are running will be automatically migrated to the other ESXI host DRS is nothing but automatic Vmotion
  + We need to enable DRS on cluster level

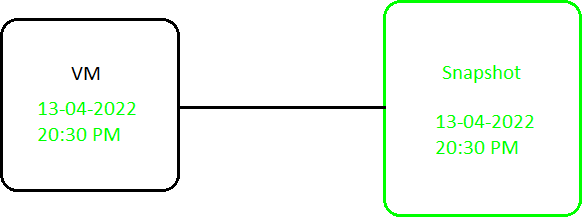
###### Advantages ::

1. Automatic Vmotion
2. Used to manage the load on ESXI Host

###### Requirements ::

1. Shared network and shared storage
2. Same CPU in the source and destination ESXI Host
3. Vmkernal Port Group must be [enabled on VM’s] configured
4. Multiple Datastores must be connected to same ESXI host
5. Datastore needs enough space for migration
6. There will be enough resources in the cluster
7. HA must be enabled in cluster level
8. At least 2 ESXI hosts must be there in cluster
9. Same CPU on source and destination ESXI hosts [All VM's Requirements]
10. Snapshot
11. Clone
12. Template

###### Snapshot :: \*\*\*

* + Snapshot is nothing but a point in time copy of Virtual machine
  + It captures the state of VM at a specific point in time
  + The state includes VM power states and VM files
  + We can take up to 32 snapshots per VM
  + Snapshot size will be less than the VM size

.vmsn file

After creating a snapshot, .vmsn file will be created in data store We can do following operations on snapshots

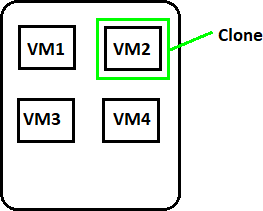
1. Take snapshot
2. Revert snapshot
3. Delete snapshot

**\*\*\*\*\*\***Snapshots in VMware environments are taken at the virtual machine (VM) level, not at the cluster level. . you take a snapshot of a VM, you capture its current state, including disk, memory, and settings, at that . . specific point in time.

###### How to take snapshot

1. Login to Vcentre/vsphere
2. Right click on VM
3. Click on snapshot
4. Select “Take Snapshot”

###### Clone :: \*\*\*

* + It is a exact copy of the VM
  + When we clone a VM it creates duplicate copy of the VM
  + Clone requires equal amount of diskspace in datastore
  + It is used for backup of data/vm
  + We can take clone when VM is power-on /off state
  + Power-on : 1 Clone
  + Power-off : 8 Clones

###### There are two types of clones

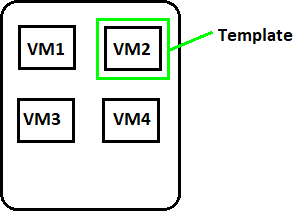
1. Cold Clone
   1. Taking a clone when the VM is in power-off state
2. Hot Clone
   1. Tacking a clone when the VM is in power-on state

###### How to take clone

1. Login to Vcentre/vsphere
2. Right click on VM
3. Select clone
4. Give target data store / name of the VM
5. Select
   1. Customize OS
   2. Customize hardware
   3. Power on VM on startup
6. Click on create
7. **Just read and understand**
8. **VM Level Cloning**:
9. In VM level cloning, you clone individual virtual machines within your virtualization environment. This is the most common method and is typically performed through the management interface of your virtualization software (e.g., VMware vSphere, VirtualBox, Hyper-V). VM level cloning allows you to create copies of specific VMs with their configurations, operating systems, and applications intact.
10. **Cluster Level Cloning**:
11. In some advanced virtualization environments, particularly those using technologies like VMware vSphere with VMware vCenter Server, cloning can be performed at the cluster level. This means that you can clone entire clusters of VMs, which may include multiple virtual machines across different hosts and datastores. Cluster level cloning is more complex and typically requires advanced management and orchestration tools provided by the virtualization platform.

###### Template :: \*\*\*

* + Template is a golden copy of the virtual machine
  + Template is used to create multiple VM’s at a time
  + By using Vcentre we can create templates
  + Templates takes less time compared to the Clone
  + Template is not VM, So we can’t power-on it



**OVF Template** :: Open virtualization format Take template

Deploy template

###### How to convert VM to Template

1. Login to vcentre
2. Right click on vm / cluster
3. Convert vm to template

We can deploy a template and create multiple virtual machines

OVA : Open virtual appliance

OAF : Open virtual format

**Understand below matter ::**

**Vswitch Distributed Vswitch**

**VRA [V realize automation]**

**Vswitch ::**

virtual switch, plays a critical role in managing network connectivity for virtual machines VM.

**Standard Vswitch ::**

Standard vswitchh establishes the connection between vm’s with in esxi host

**Distributed Vswitch ::**

Distributed Vswitch will used to establish the connection vm’s across the esxi hosts in cluster level

**VRA [V realize automation]**

vRealize Automation (vRA) is a cloud automation platfo3rm developed by VMware. It allows organizations to automate the delivery . of IT infrastructure and applications across multi-cloud environments, including private, public, and hybrid clouds. Here's an over . . view of vRealize Automation:

**Kernal ::** The kernel acts as a mediator between the hardware of a server and the software applications

# \*\*\* Storage \*\*\*

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

The Place where we can keep our things pen drives, Memory Cards, Hard disks, CD/DVD’s 64GB, 128GB, 500GB (Day to day life)

##### Enterprise companies::

Amazon Google Facebook

Storage Devices:: **Storage Arrays**:: Data Centers The first storage device is invented by IBM **Storage Manufacturing Companies ::**

IBM

Dell EMC HP

NetApp Seagate

SanDisk (Western Digital) Samsung

Toshiba Hitachi Fujitsu

**Solution provider companies::**

Amazon Microsoft Google Veritas

Netapp DDN

**Storage::**

The Place where we will keep out data The first storage device is invented by IBM **Storage Device::**

The device which we storage our data Examples:

Pen drives, Memory Cards, Hard drive, Storage Arrays Kilo Byte [KB] = 1024 Bytes

Mega Bytes [MB] = 1024 KB’s GB = 1024 MB’s

TB = 1024 GB’s PB = 1024 TB’s ZB = 1024 PB’s

#### Types of storage::

1. Primary Storage
2. Secondary Storage

##### Primary Storage::

* + The storage which is available with the laptop/server is called as primary storage
  + It is also known as main memory (storage space)
  + We will install OS on primary storage devices

**Ex:** Internal Storage space of Mobiles, Internal Hard drive, Boot drive

##### Secondary Storage::

* + The storage which we connect external to the laptop/server

**Ex:** External Hard drive, JBOD, EBOD, Storage Array

1. Local Storage
2. Remote Storage

##### Local Storage::

The storage space which is available locally with the system is called as local storage. Ex: Hard drives which is connected to laptop, DAS (Direct Attached Storage), Pen drive, JBOD **Remote Storage::**

The storage space which we can access remotely is called as remote storage

Ex: Google Drive, One drive, SAN (Storage Area N/W) and NAS (Network Attached Storage)

**Redundancy::** In case of single drive failure we can still access the data from other drives

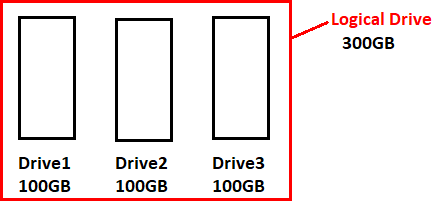
## \*\*\*RAID ::

Raid stands for **Redundant Array of Independent disks Array ::** Group

**Independent Drive ::** Drive which is not part of any RAID group

**Redundant ::** In case of one drive failure we can retrieve the data from another drives

* RAID is a technology where we will apply RAID on top of the group of physical drives and create a logical drive called RAID Group



###### Why we are going for RAID?

To achieve Redundancy (To retrieve the data in case of drive failures) To avoid the data loss in case of drive failure

**Array::** Group of drives

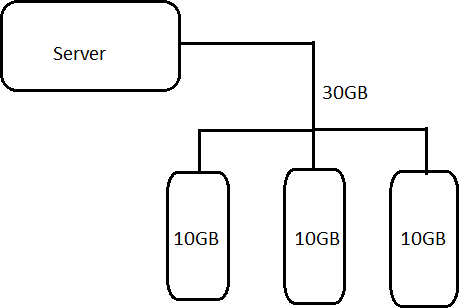
**Independent Drive::** The drive which is not part of any RAID group

###### RAID Techniques::

1. Striping
2. Mirroring
3. parity

**Striping::**

Distributing the data equally all the drives in a RAID Group is called as Striping



**Mirroring::**

Copying the same data on all the drives called as Mirroring

##### RAID Levels::

RAID 0 > R0 RAID 1 > R1 RAID 2 > R2 RAID 3 > R3 RAID 4 > R4 RAID 5 > R5 RAID 6 > R6

##### Nested RAID Levels::

RAID 0 + RAID 1 :: RAID 01 [R01]

RAID 1 + RAID 0 :: RAID 10 [R10]

RAID 5 + RAID 0 :: RAID 50 [R50]

RAID 6 + RAID 0 :: RAID 60 [R60]

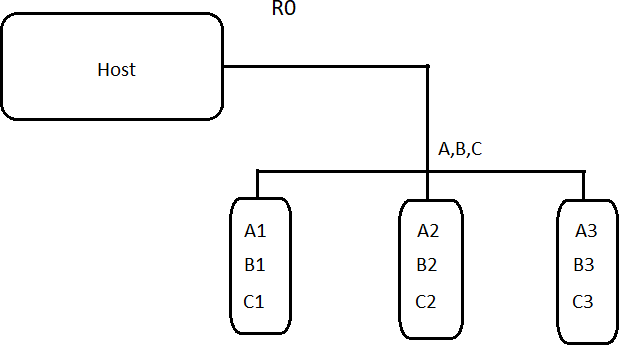
##### Fault Tolerance (FT)::

No. of drives failures that RAID Level can withstand

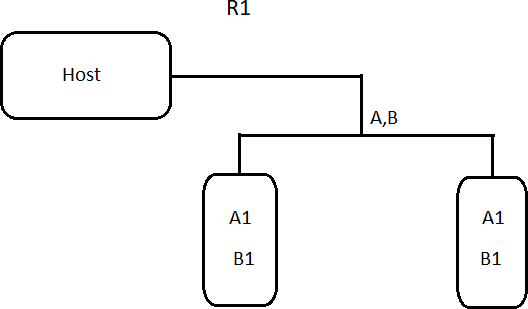
Command for create sfware raid ::”” **mdadm -C /dev/mdo -l 0 -n 2 /dev/sdd /dev/sdb “” [C==**create**, l=**raid-level**, n=**no.of drives**,**

**/dev/mdo=** raid name**,, /dev/sdd=**drive name**]**

###### RAID 0 ::

* RAID 0 follows striping technique
* It requires minimum of 1 drive
* It supports maximum drives upto 32 ( based on Vendor Specific)
* There is no redundancy in RAID 0, So RAID 0 is called as non-redundant RAID level and not an true RAID level
* Fault tolerance of RAID 0 is zero
* Performance of R0 is better than R1
* Redundancy of R0 is less than R1
* Total available space = (no. of drives-F.T)\*size of the drive [(N-FT)\*size]
* Total available space of R0 = N\*size of the drive

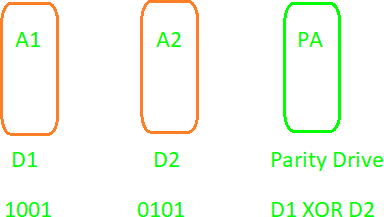
###### RAID 1::

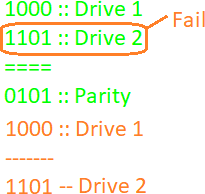
* R1 follows Mirroring techniques
* Whatever the data comes from the host it will copy on all the drives
* It requires minimum of 2 drives
* It supports maximum drives also 2
* It can withstand with 1 drive failure
* Fault tolerance of R1 is 1
* 50% of the drive space is getting wasted for mirroring
* Total available space is N/2\*size of the drive
* Performance of R1 is less than R0
* Redundancy of R1 is better than R0

#### Parity::

It calculates the difference between 2 drives and stores the output in another drive XOR Method Calculation





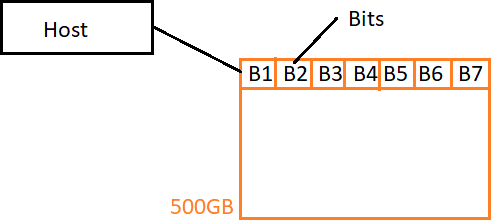


###### Storage measurements

KB, MB, GB, TB, PB, ZB

1 KB = 1024 Bytes

1 Byte = 1024 Bits



###### Distributing the data equally among all the bits is called as bit level striping

**Dedicated Parity::**

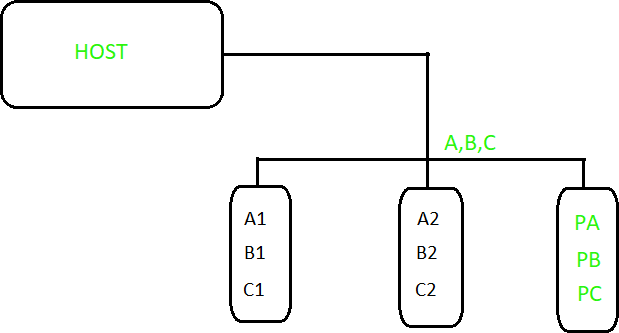
Writing all the parity data in one drive is called as Dedicated Parity

###### Distributed Parity::

Writing Parity data on all the drives is called as Distributed Parity

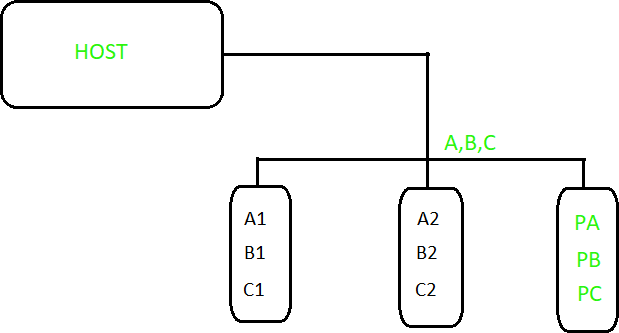
**RAID 3::**

* Follows Byte level Striping with Dedicated parity
* It requires minimum of 3 drives
* It supports maximum drives upto 32



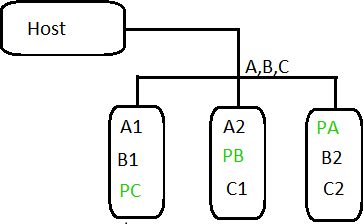
* R3 can withstand with 1 drive failure
* So fault tolerance of R3 is 1
* Total available space is (N-1)\*size of drive (N = No. of drives and 1 = Fault Tolerance)
* Performance of R3 is less than R4
* Redundancy of R3 and R4 are same

**RAID 4**::

* R4 Follows block level striping with Dedicated Parity
* It requires minimum of 3 drives
* It supports maximum drives upto 32
* It can withstand with 1 drive failure
* Fault tolerance of R4 is 1
* Total available space = N-1\*size of the drive
* Performance of R4 is better than R3
* Redundancy of R3 and R4 are same

###### RAID 5::

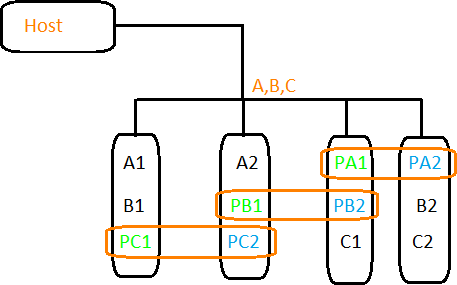
* R5 follows Block Level Striping with Distributed parity



* It requires minimum of 3 drives
* It supports maximum drives upto 32
* It uses XOR algorithm to calculate parity
* It can withstand with single drive failure
* So, Fault tolerance of RAID 5 is 1
* Total available space = (N-1)\*size of drive
* R5 uses XOR algorithm for calculating the parity
* Performance of R5 is better than R6
* Redundancy of R5 is less than R6

###### RAID 6::

* R6 follows Block Level Striping with Dual Distributed parity



* It requires minimum of 4 drives
* It supports maximum drives upto 32 [Vendor Specific]
* It can withstand with 2 drives failure
* So, Fault tolerance of RAID 6 is 2
* Total available space = (N-2)\*size of drive
* For first parity calculation R6 uses XOR Algorithm
* For second parity calculation R6 uses Read Solomon algorithm
* Redundancy of R6 is better than R5
* Performance of R6 is less than R5

Total Available Space

No. of drives –Fault tolerance \*size of the drive 100GB 100GB 100GB :: R0, R5

R0 :: (3-0)\*100 = 300GB R5 :: (3-1)\*100 = 200GB

If we have different capacity drives RAID will always take least capacity drive size as a actual size for all the drives.

100GB

500GB

200GB

R5 :: (3-1)\*100 = 200GB

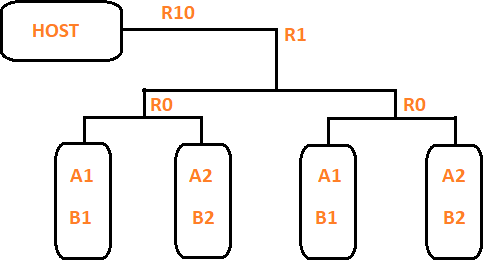
**Nested RAID Levels::**

Combination of two RAID Levels

1. RAID 1 + RAID 0 [R10]
2. RAID 0 + RAID 1 [R01]
3. RAID 5 + RAID 0 [R50]
4. RAID 6 + RAID 0 [R60]

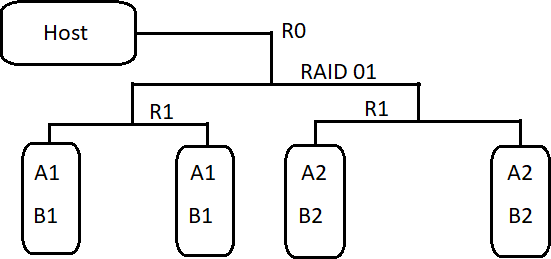
To overcome the limitations [Low performance / Redundancy] we are combining 2 RAID levels and creating nested RAID Level

1. **RAID 10 [RAID 1 + RAID 0]::**
   * To increase the performance of R1 we are combining R1 with R0
   * RAID 10 follows Striping across mirroring



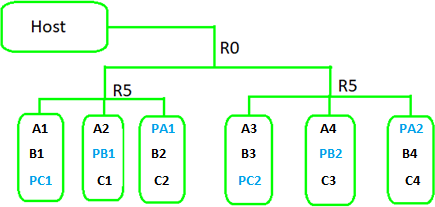
* + It requires minimum of 4 drives
  + It supports maximum drives upto 32 [Vendor Specific]
  + R10 can withstand with 2 drives failures, But not form the same set
  + Fault tolerance of R10 is 2
  + Total available space = (N-2)\*size of drive
  + The performance of R10 is better than R01
  + Redundancy of R10 and R01 are same

1. **R01 (RAID 0 + RAID 1)::**
   * It follows Mirroring across the striping



* + It requires minimum of 4 drives
  + It supports maximum drives upto 32 [Vendor Specific]
  + R01 can withstand with 2 drives failures but not from the same set
  + Fault tolerance of R01 is 2
  + Total available space = (N-2)\*size of the drive
  + Performance of R01 is less than R10
  + Redundancy of R01 is same as R10

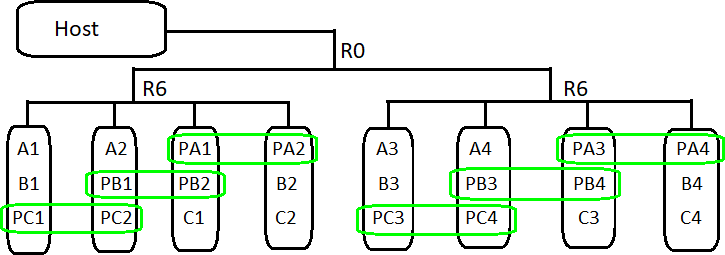
1. **R50 [RAID 5 + RAID 0]::**
   * Striping across R5 sets
   * R50 follows striping across R5 sets



* + It requires minimum of 6 drives
  + It supports maximum drives upto 32 [Vendor Specific]
  + It can withstand with 2 drives failures
  + Fault tolerance of R50 is 2
  + Total available space = (N-2)\*size of the drive
  + Performance of R50 is better than R60
  + Redundancy of R50 is less than R60

**R60 [RAID 6 + RAID 0] (Striping across R6 sets)::**

* + It follows striping across R6 sets



* + It requires minimum of 8 drives
  + It supports maximum drives upto 32 [Vendor Specific]
  + It can withstand with 4 drives failures but not from the same set
  + Fault tolerance of R60 is 4
  + Total available space = (N-4)\*size of the drive
  + Redundancy of R60 is better than R50
  + Performance of R60 is less than R50

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **S. No** | **RAID's** | **Minimum Drives** | **Maximum Drives** | **Performance** | **Redundancy** | **Fault Tolerance** | **Total Available Space** |
| 1 | R0 | 1 | 32 | R0 is better than R1 | 0 (R0 is less than R1) | 0 | N\*size of the drive |
| 2 | R1 | 2 | 2 | R1 is less than R0 | R1 is better than R0 | 1 | N/2\*size of the drive |
| 3 | R3 | 3 | 32 | R3 is less than R4 | R3 and R4 same | 1 | (N-1)\*size of the drive |
| 4 | R4 | 3 | 32 | R4 is better than R3 | R3 and R4 same | 1 | (N-1)\*size of the drive |
| 5 | R5 | 3 | 32 | R5 is better than R6 | R5 is less than R6 | 1 | (N-1)\*size of the drive |
| 6 | R6 | 4 | 32 | R6 is less than R5 | R6 is better than R5 | 2 | (N-2)\*size of the drive |
| 7 | R01 | 4 | 32 | R01 is less than R10 | R01 and R10 same | 2 | (N-2)\*size of the drive |
| 8 | R10 | 4 | 32 | R10 is better than R01 | R10 and R01 same | 2 | (N-2)\*size of the drive |
| 9 | R50 | 6 | 32 | R50 is better than R60 | R50 is less than R60 | 2 | (N-2)\*size of the drive |
| 10 | R60 | 8 | 32 | R60 is less than R50 | R60 is better than R50 | 4 | (N-4)\*size of the drive |

## RAID States:: \*\*\*\*

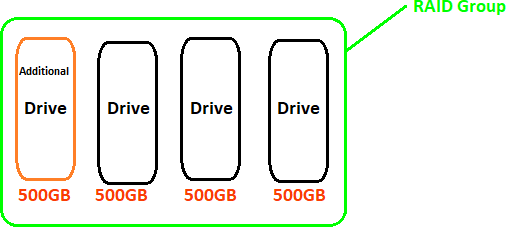
1. **Initialization:**
   * As soon as we create raid the state of the RAID goes to **initialization** [converting physical drives into logical drives / RAID group]
   * We can’t perform read/write operations in initialization state
2. **Online:**
   * After initialization the state of the RAID goes to **online**
   * We can perform read and write operations in online state
3. **Degraded:**
   * If drive fails within the fault tolerance level in RAID level, the state of the RAID goes to **Degraded**
   * We can perform read and write operations if the RAID is in degraded state Example ::
     1. One drive failure in R5
     2. Two drive failures in R6
4. **Offline:**
   * If drive fail more than it’s fault tolerance level the state of the RAID goes to **Offline**
   * We can not perform read and write operations if the RAID is in offline state Example ::
     1. One drive failure in R0
     2. Two drive failures in R5
     3. Three drive failure in R6
5. **Reconstruction / Rebuilding:**
   * After replacing failed drive with working drive the data in failed drive will start rebuilding [copying] in good drive then the start of the RAID goes to rebuilding / reconstruction.
   * We can perform read and write operations if the RAID is in rebuilding state

**RAID Features::**

1. ORCE [Online RAID Capacity Expansion]
2. ORLM [Online RAID Level Migration]
3. Hot spare
4. Hot swappable/Hot pluggable

**ORCE:**

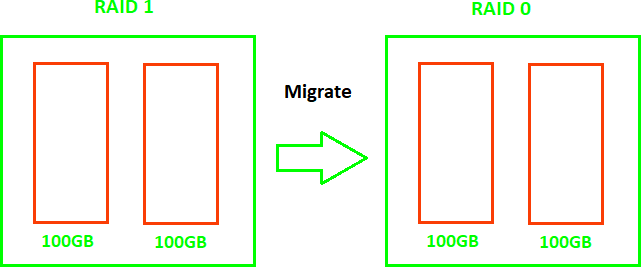
* + It stands for online RAID capacity expansion
  + When RAID level is online state, we can increase the RAID capacity by adding additional drives to the RAID group this is called ORCE

 **[mdadm --add /dev/md0 /dev/sde]**

**ORLM:**

* + ORLM stands for Online RAID Level Migration.
  + When RAID level is in online state we can migrate RAID level from one RAID level to another supported RAID level based on the drive availability is called ORLM.

**[mdadm --grow /dev/md0 –level=5]**



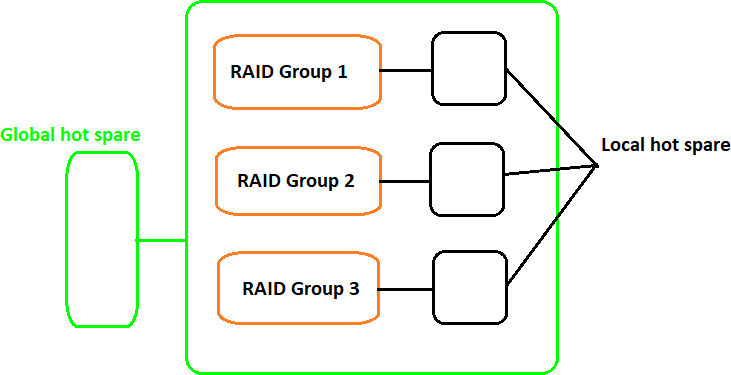
**Hot spare:**

* + If we assign hot spare to the RAID group, when ever drive fails in RAID group the hot spare drive will automatically becomes the part of RAID group
  + Assigning the spare drive to RAID group is called as hot spare
  + Spare drive :: which is not part of any RAID Group

There are two types of hot spares

1. Local hot spare
2. Global hot spare

###### Assigning Spare drive to the individual RAID group is called as local hot spare Assigning Spare drive to the group of RAID group is called as global hot spare



If drive fails and both local and global hot spares are assigned, local hot spare first becomes the part of RAID Group

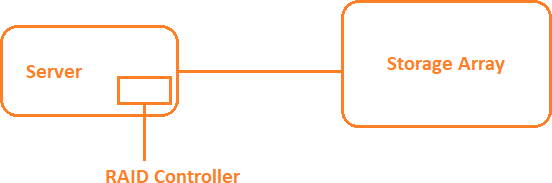
**Hot swappable/Hot pluggable:**

* + When everything is up and running, we can remove and re insert the drives from RAID group is called as Hot swappable/Hot pluggable

## Types of RAID Groups:

1. Software RAID
2. Hardware RAID

Software RAID Hardware RAID



* + Creation of RAID by using the Operating System (OS) is called as **software RAID**
  + Creation of RAID by using the external device (RAID Controller) is called as **Hardware RAID**

###### How to create Software RAID and Hardware RAID:

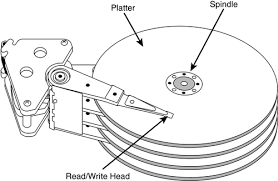
1. Software RAID “mdadm” [Tool]
2. Hardware RAID “Storcli” [Tool]

|  |  |
| --- | --- |
| **Software RAID** | **Hardware RAID** |
| Creation of RAID by using the operating system (OS) is called as Software RAID | Creation of RAID by using the external device (RAID Controller) is called as Hardware RAID |
| Software is OS dependent | Hardware is OS independent |
| Software RAID is less reliable | Hardware RAID is more reliable |
| Implementation of software RAID is simple and less expensive | Implementation of Hardware is expensive |
| More Burden on CPU | Less Burden on CPU |

## Types of drives::

1. HDD [Hard Disk Drive]
2. SSD [Solid State Drive]

###### HDD:

* + HDD stands for Hard Disk Drive
  + It works based on the mechanical components



###### Components of HDD:

1. Platter
2. Spindle
3. Read/Write head
4. Actuator Arm
5. Actuator

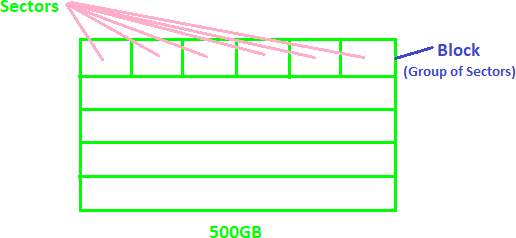
**Platter:** It is a circular disk [rotating component] where user data will set

It is a rotating circular disk where actual data will be stored in HDD

**Spindle:** It is a motor inside the HDD which helps platter to rotate **Read/Write head:** Which is used to write the data in platter **Actuator:** Actuator receives data from host and send it to platter

###### How Data writes in HDD:

* + In HDD data stores in platters in a form of **Sectors**, Group Sectors are called as **blocks**



* + We can define our own block size
  + The standard block size is 4k
  + HDD is less expansive
  + HDD’s are more reliable [As they use Mechanical components]
  + Life span of HDD is more
  + Form factor of SSD is 3.5 inches

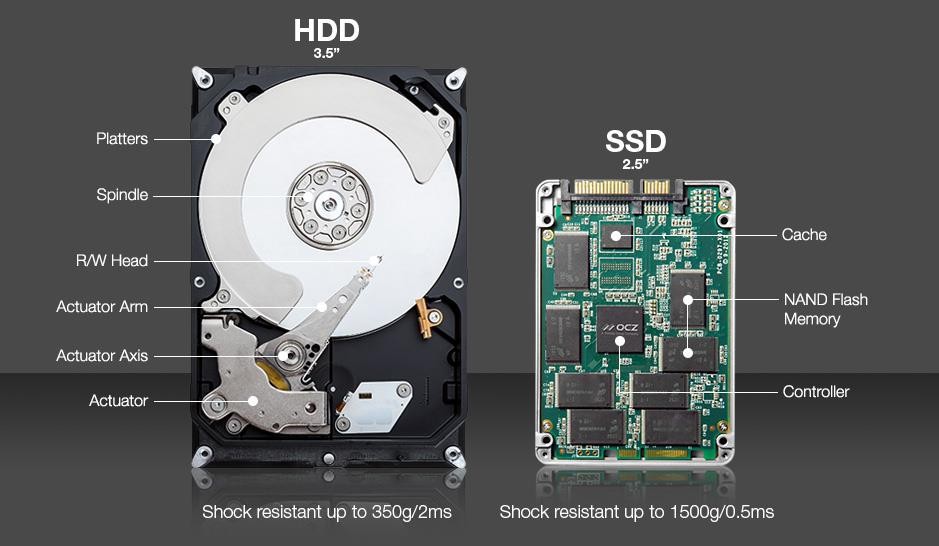
## \*\*\*\*SSD [Solid State Drive]:

SSD Stands for Solid State Drive

It works based on the Flash Memory Types of Flash Memory

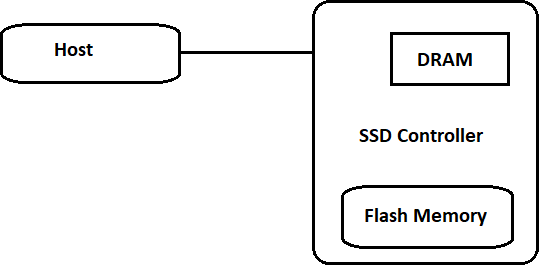
1. NAND
2. NOR

SSD’s uses NAND based Flash Memory

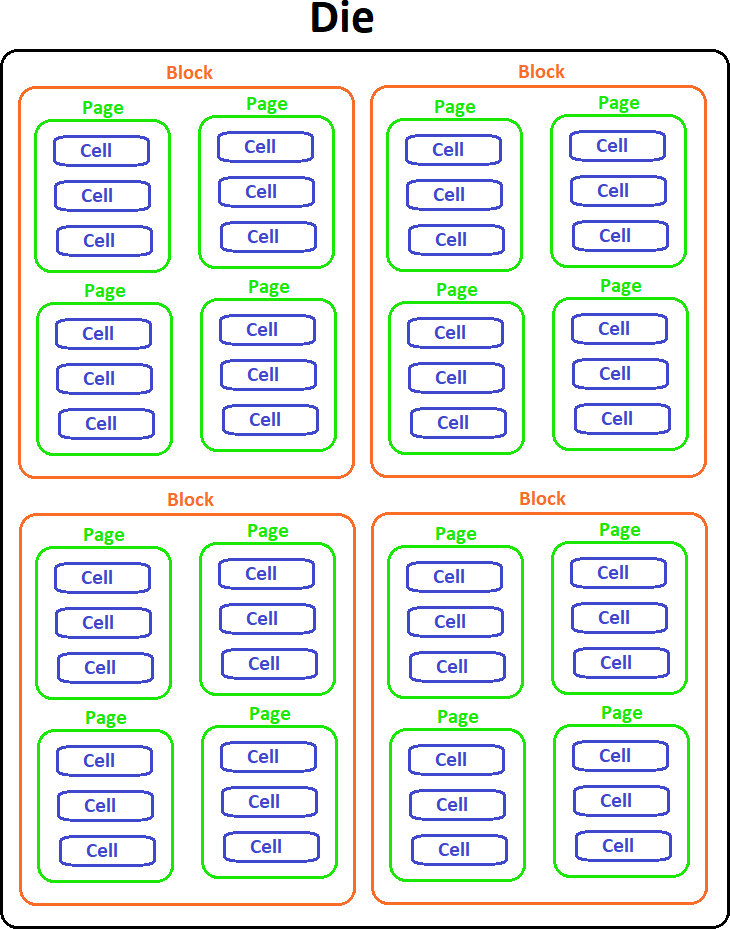


###### Components of SSD

1. DRAM [Cache]
2. SSD Controller
3. Flash Memory [NAND]



###### How SSD Works

* + In SSD data stores in Flash Memory
  + From host data comes to DRAM [Cache Memory]
  + First the data will be written in a DRAM [Cache Memory]
  + SSD Controller takes the data from DRAM and writes in Flash Memory [NAND]
  + In NAND Data stores in a form **cells**
  + Group cells are called as **Pages**
  + Group of pages are called as **Blocks**
  + Group of Blocks are called as Die
  + Lifespan of SSD is less than HDD
  + Form factor of SSD is 2.5 inches

###### Difference between HDD and SSD

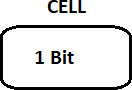
|  |  |
| --- | --- |
| **HDD** | **SSD** |
| HDD stands for Hard Disk Drive | SSD Stands for Solid State Drive |
| In HDD data stores in Platters | In SSD stores in Flash memory[NAND] |
| In HDD data stores in form of Sectors / Blocks | In SSD data stores in form of Cells /Pages / Blocks |
| Form factor HDD is 3.5 inches | Form factor of SSD is 2.5 inches |
| Lifespan of JDD is more than SSD | Lifespan of SSD is less than HDD |
| HDD’s are more reliable | SSD’s are less reliable |
| HDD’s are less expansive | SSD’s are more costly |
| HDD’s is slower than SSD | SSD is faster than HDD |

**Types of NAND /SSD’s**

1. SLC [Single Level Cell]
2. MLC [Multi Level Cell]
3. TLC [Tri Level Cell]
4. QLC [Quadra Level Cell]

###### SLC:

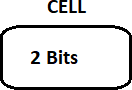
* + SLC stands for Single Level Cell
  + It can store 1 bit per cell



* + SLC Drives are costlier than MLC, TLC and QLC Drives
  + Performance of SLC drives is better than MLC, TLC and QLC Drives
  + Lifespan is also more than MLC, TLC and QLC Drives

###### MLC:

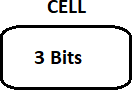
* + MLC stands for Multi Level Cell
  + It can store 2 bits per cell



* + MLC Drives are less expensive than SLC Drives and more expansive than TLC and QLC Drives
  + Performance of MLC is less than SLC and more than TLC and QLC drives
  + Lifespan of MLC is less than SLC and more than TLC and QLC drives

###### TLC:

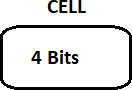
* + TLC stands for Tri Level Cell
  + It can store 3 bits per cell



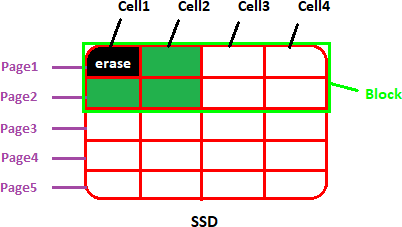
* + TLC drives are less expensive than SLC and MLC Drives and more expansive than QLC drives
  + Performance of TLC is less than SLC and MLC and more than QLC
  + Lifespan of TLC Drives is less than SLC and MLC and TLC Drives

###### QLC:

* + QLC stands for Quadra Level Cell
  + It can store 4 bits per cell



* + QLC Drives are cheaper than SLC, MLC and TLC Drives
  + Performance of QLC is less than SLC, MLC and TLC Drives
  + Lifespan of QLC is less than SLC, MLC and TLC Drives
* In HDD we can over write the data
* In SSD we need to erase the data before re writing
* In SSD write Happens in Page Level and erases happens on Block Level



##### Why Erase Happens in Block Level:: \*\*\*

In SSD Data writes in page level, If we erase individual page [Group of cells] there are high chances for other cell to get impact, To avoid the SSD controller will do erase in a block level.

##### Features of SSD:: \*\*\*

1. **Wear Leveling**
2. **Write Amplification** :: Write amplification occurs when a NAND flash-based SSD writes more data to disk than the

host submits

1. Endurance :: The amount of data you can write to the SSD before the device wears out and can no longer

Store data safely

1. P/E Cycles
2. Erase Blocks
3. **Garbage Collection** :: Garbage collection basically copies data still in use to a new block, and the

Deletes all data from then hold data

1. **Over Provisioning** :: it reserving a portion[some data] about 10% of a ssd’s total capacity which is not visible

To the user. The reserved space is used by ssd’s controller for tasks such as wear-leve

Ling, garbage collection,and maintaining performance consistency

1. LBA
2. FTL

###### Wear Leveling ::

* + If we continuously writes and erases the same cell there are high chances for that particular cell to go bad [Wear Out].
  + To avoid that SSD controller distributes writes equally among all the cells this process is called as wear leveling
  + Lifespan of SSD will increase with Wear leveling

**Factors Effects the Performance of Storage Device**

**Data transfers in form commands [ Throughput ]**

**IOPS [Input Output Per Second]**

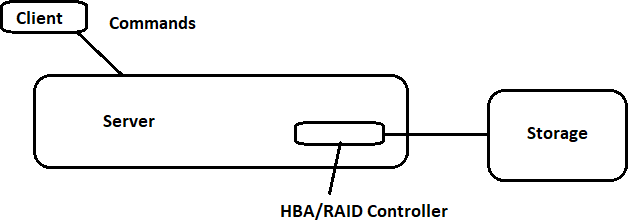
**Latency**

Queue depth **Block Size**

###### Throughput::

* + At what speed the command is travelling to the target is called as Throughput
  + It is measured in Numbers

###### Latency::

* + The time taken by a command to reach the target is called as Latency
  + It is measured in mille seconds or Micro Seconds

### \*\*\*HBA::

* + HBA Stands for Host Bus Adapter
  + It establishes the connection between server and storage
  + It takes the commands from Users and send to the storage (Target)

###### RAID Controller::

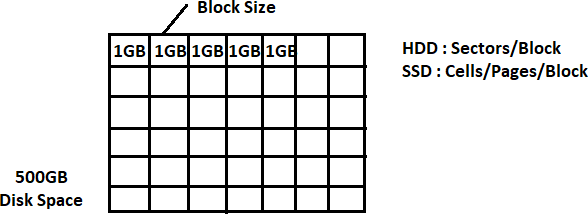
* + It works same as HBA, only difference is RAID controller comes with RAID features as default

###### Queue depth::

* + No. of outstanding commands that controller/HBA can hold at a time is called as Queue depth
  + It is measured in Numbers
  + Queue depth is depends on no of queues
  + Queue depth == commands

###### Block Size::

* + It is amount of storage space. In drives the storage space is divided into No. of small chunks of data called as Block



* + Performance of storage system will depend on the Block size Queue depth
  + We need to give Block size and Queue depth[cmds] as a input to check the performance and we will measure IOPS, Latency and Throughput

High IOPS > more performance High Latency > less performance

High Throughput > High performance

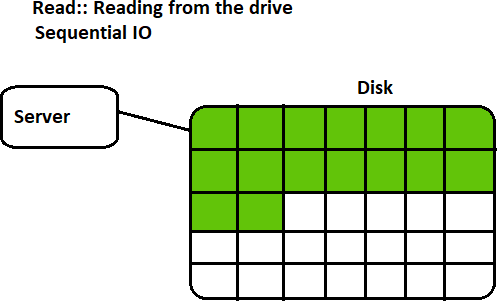
###### Tools used for measuring performance of storage device

1. IO meter :: windows
2. CDM [Crystal disk mark] :: windows
3. **FIO[Flexible IO] :: Linux [command for fio tool]**
4. [ fio --filename=/dev/sdd --name=test1 --bs=4KB --iodepth=32--runtime=5m --io engine=libaio --rw= randwrite]
5. Vdbench :: Linux

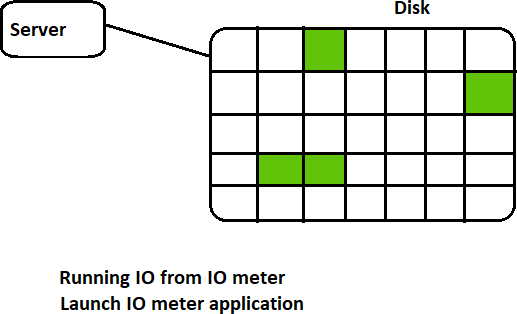
###### Types of IO [Write and Read]

1. Sequential IO
   1. Sequential write
   2. Sequential read
2. Random IO
   1. Random write
   2. Random read **Write::** Writing to the drive **Read::** Reading to the drive

**Sequential IO**



**Random IO**



**[Lab Practice steps]** Running IO from Io meter Launch IO meter application

Go to disk targets and select the drive Click on Access Specifications

Select IO pattern

###### Running IO from IO meter::

1. Launch IO meter application
2. Go to disk targets and select the drive
3. Click on Access Specification
4. Select Io pattern
5. Go to result display and select no. of hours/minutes and test name
6. Click on [Green flag]
7. Save the result

After running the test Go to result display

Observe:: IOPS, Throughput, Latency

IO pattern [type of IO, Block size and Queue depth] 512B; 100% read; 0% random

512B:: 512Bytes [Block size] 100% read:: Sequential read

0% random:: 100% Sequential lO

64KB; 50% read; 100% random Block size = 64KB

50% read 50% write Random write/read

:: with sequential IO we will measure the Throughput

:: with Random IO we will measure the IOPS

To Run IOPS from Linux::

<Leave 3 Pages space>



**Firmware::** Firmware is a software that resides in a hardware device Without firmware hardware device will not work

###### Device Driver::

Device drivers is a software that establishes the communication between software and hardware

We will get the firmware files from developers Firmware images ends with extension of .bin or .ini

We need to install the firmware on top of hardware

For firmware installation we use tools like hdparm, SUU, FUU etc…

**Syntax**::

Hdparm -| <device\_name> <firmware\_images> C:\User\Eswar\Software\SUU1286138.bin (path address)

|  |  |  |
| --- | --- | --- |
| **PARAMETER** | **DEVICE DRIVER** | **FIRMWARE** |
| Purpose | To help in the communication  between an operating system and the hardware | Defines the function ability of the hardware |

**RAID Group** :: Group of physical drives combined and called as a RAID Group

**Volume** :: The small amount of storage space that we can derive from the RAID Group

We can create upto 32 volumes per RAID Group

###### How to create volume in Linux

Volume create –name=<volume\_name> --raid-group=<raid\_group\_name> --size=<size of volume>

**LUN** :: Logical Unit Number

After creating volume, a unique number will be assigned to each volume that is called as LUN LUN’s always starts with 0

We identify the volume by its name or LUN

###### LUN Mapping

Mapping the volume [LUN] to the target is called as LUN mapping or

Presenting the Volume to the user is called as LUN mapping

**Linux**::

map voume –name=volume\_name –lun=LUN\_ID –target=<Target\_WWN\_SerialID\_IQN>

###### LUN Masking

Making the LUN [Volume] invisible to the particular user is called as LUN masking

\*\*\*\*\*\*\* **Storage Protocols** \*\*\*\*\*\*\*

###### Protocol:: Set of rules and standards

**IEEE::** International Electrical and Electronics Engineering

**NAA::** National Address Authority

**SNIA::** Storage networking industry association

###### Storage Protocols

ATA SATA

SCSI SAS

Iscsi FC FCOE

Nvme Nvme-OF

### ATA::

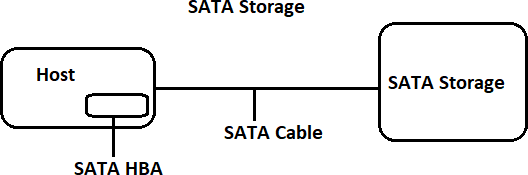
* ATA Stands of Advanced Technology Attachment
* ATA is also called as IDE (Integrated Device Electronics)
* ATA supports speed up to 300mbps (Throughput)
* ATA supports cable length up to 42 cm
* ATA supports up to 2 devices
* ATA drives are identified by serial id
* ATA is having its own command sets called as ATA commands

###### Few ATA Commands

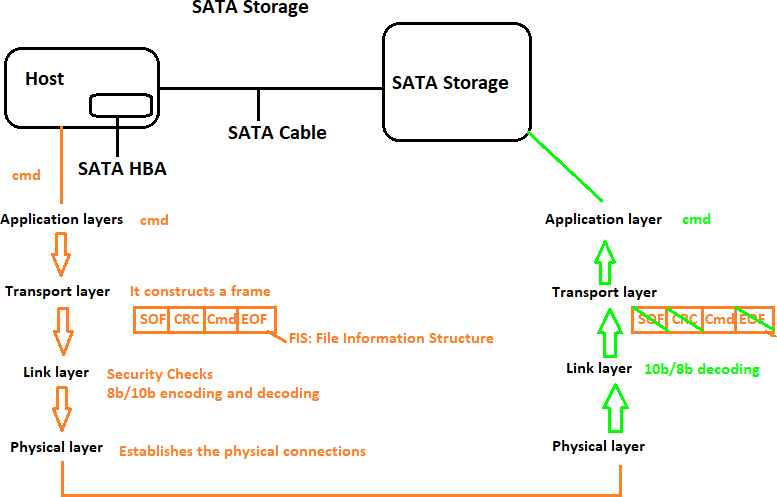
1. **Read**:: reading from the drive
2. **Write**:: Writing to the drive
3. **Identify::** Identify a specific device in ATA domain
4. **ABORT**:: To stop the previously sent command (Previous operation)
5. **TRIM**:: It gives the location of deleted data (Location of Garbage Data)

### SATA::

* + SATA stands for Serial Advanced technology Attachment
  + SATA supports up to 8 devices
  + SATA supports speeds up to 6 GBPS
  + Cable length of SATA is 1 meter
  + SATA is a serial protocol
  + If we want to implement SATA storage we need to have SATA HBA, SATA cable
  + SATA is available as both HDD and SSD’s
  + SATA Devices are identify by serial ID and RPM of SATA is 7200



**Architecture of SATA**



**SOF**:: Start of frame

**EOF**:: End of frame

**CRC**:: Cyclic Redundancy Check

###### Primitive::

* + Application layer receives the data from host and sends to the transport layer
  + Transport layer receives the data from application layer and constructs the frame and sends to the link layer
  + Link layer receives the data from transport layer and performs 8b/10 encoding and decoding and sends the frame to physical layer
  + Physical layer establishes the physical connections between host to target

### SCSI::

* + SCSI stands for small computer system interconnect/interface
  + SCSI has its own command sets called as CDB [Command Descriptor Block]
  + SCSI supports speeds up to 8 GBPS
  + Cable length of SCSI is 6 meters
  + It supports 16 devices

###### There two types of SCSI::

1. Narrow SCSI:: 0 - 7 devices
2. Wide SCSI:: 8 - 16 devices

###### SCSI Commands:

1. **Read**:: Reading from the drive
2. **Write**:: Write to the drive
3. **ABORT**:: To stop the previously executed command
4. **Write Buffer**:: To install firmware we use Write buffer
5. **Read Buffer**:: To read the firmware details
6. **Request Sense**:: To Check the Previously executed command status

We get the sense keys as response to the request sense command Some of the response codes [Sense keys]

**01h**:: No sense [No error]

**02h**:: Not ready

**04h**:: Hardware error

**05h**:: illegal Request [There is some corruption in data]

**FH**:: Completed

**Test Unit Ready**:: To check target is ready to receive the data or not

**Inquiry**:: To list all the drives [To check particular device]

**Unmap**:: It gives the location of deleted data [Garbage Data]

### \*\*\* SAS ::

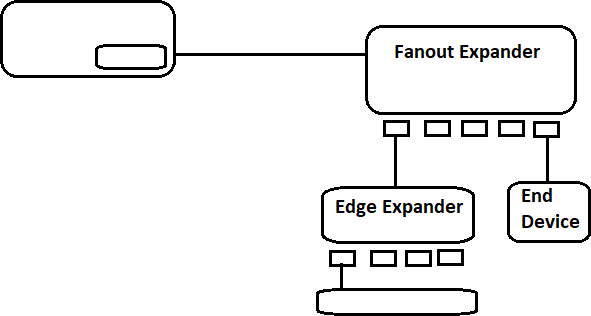
* SAS Stands for Serial Attached SCSI
* SAS uses the SCSI commands
* SAS Supports speeds upto 24 GBPS [we need host side also 24Gbps to use 24Gbps SAS]
* It supports upto 64535 devices
* It is a serial protocol [follows serial communications]
* It uses the **Expanders** to increase the no. of devices connected

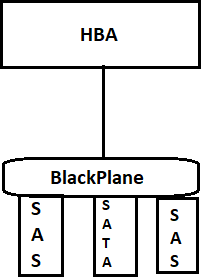
There are two types of Expanders

* 1. Fan out Expanders
  2. Edge Expanders

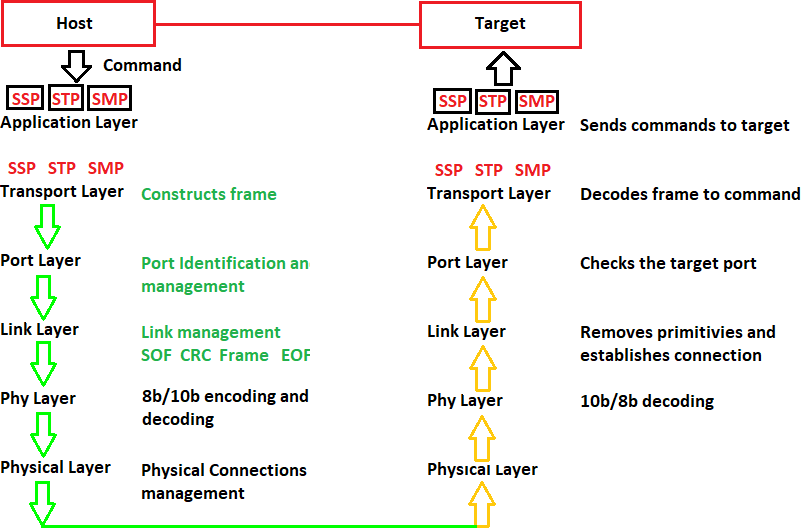


* Expander is a device which acts like a switch. It is used to extend and connect no. of devices in SAS Storage
* We can connect multiple Edge Expanders to one Fan out Expander



* SAS Devices are identified by WWN [World Wide Name]
* WWN is a 64bit Hexa decimal number
* SAS Gives Backward Compatibility to SATA Devices
* We can connect SATA Drives to the SAS HBA by using Backplane

#### Architecture of SAS or SAS Layers



**SSP** :: Serial SCSI Protocol

**STP** :: Serial Management Protocol **SMP** :: Serial ATA Tunnelling Protocol **Application Layer** ::

Application layer receives the command from host

If it is a SCSI command SSP protocol takes and transfers to Transport Layer If it is a ATA command STP protocol takes and transfers to Transport Layer

###### Transport Layer ::

Transport layer receives the command from Application layer and constructs the frame

###### Port Layer ::

Port Layer receives the frame transport layer and identifies the end port to which has to send the data

###### Link Layer ::

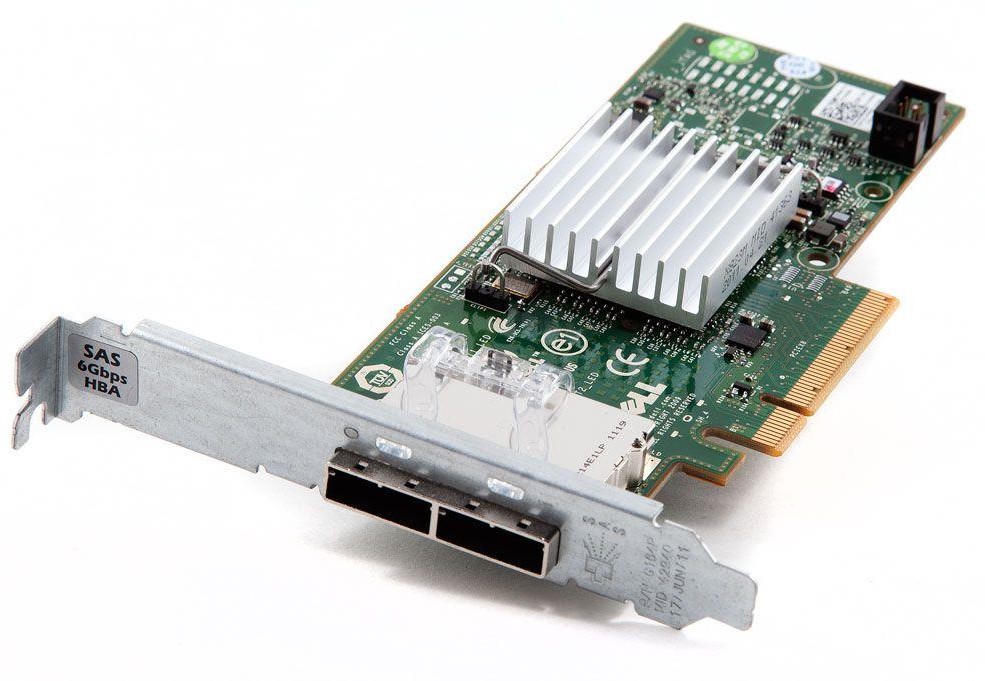
Link Layer established the connection and adds the primitives to the frame

###### Phy Layer ::

phy Layer performs 8b/10 bit encoding and decoding

###### Physical Layer ::

Physical Layer established the Physical connection between host and the target SAS HBA’s are having dual port compatibility



6GBPS

SAS Devices are available in 6GBPS and 12GBPS speeds

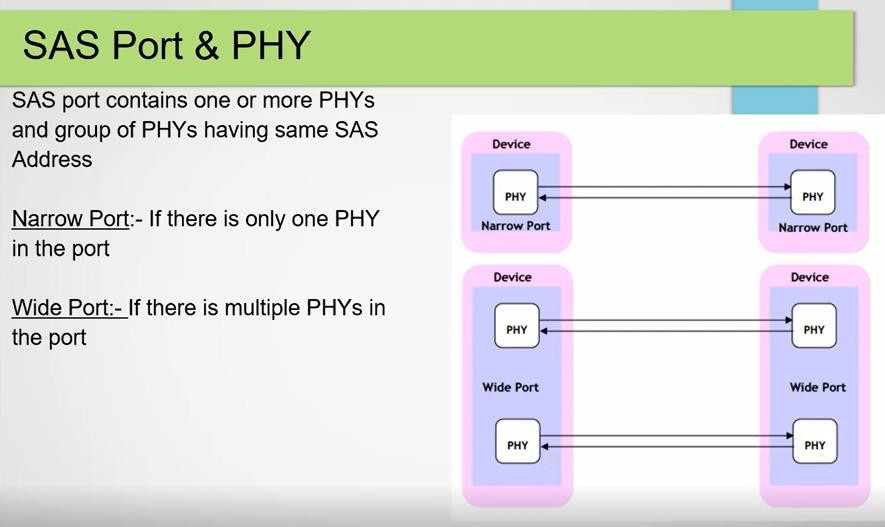


**12 GBPS ports**

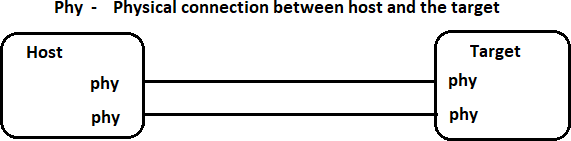


6GBPS

* In SAS connectivity happens between port to port
* Port contains one or more phy’s



* If port contains only one port we call it is narrow port
* If port contains multiple phy’s we call it as wide port
* Group of phy’s having same SAS address as port



SAS OOB Sequence [OOB – Out of band]

###### Primitives ::

COMINIT

**Host Target**

COMINT COMINT

COMSAS COMSAS [SAS Device]

COMSAS no response [SATA Device]

* If host and target exchange COMINIT then the connection is successfully established
* If host sends COMSAS and response it receives COMSAS then it is a SAS Device
* If host sends COMSAS and if it not receives COMSAS as a response then it is a SATA Device

SAS is a full duplex protocol

* SAS ports are identified by World Wide Name (WWN) is assigned by the device manufacturer. It is 64 bit hexadecimal number

4 bits assigned by NAA [National Address Authority]

24 bits assigned by IEEE [Institute of Electrical and Electronics Engineering] 36 bits assigned by Vendor

###### Types of protocols

1. Hard disk Protocol

ATA, SATA, SCSI, SAS, Nvme

1. Transmission Protocol ISCSI, FC, FCOE, Nvme-OF

**ISCSI ::** Internet SCSI

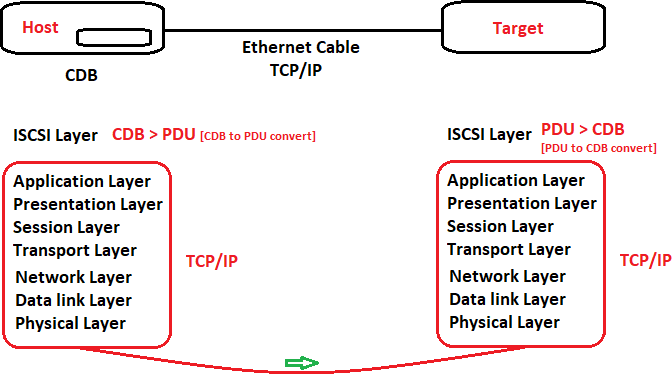
* + ISCSI uses a Transmission protocol
  + ISCSI uses SCSI commands
  + ISCSI device are identified by IQN [ISCSI Qualified name]
  + It supports speeds upto 10GBPS
  + ISCSI is widely used in Remote Storage [SAN – Storage Area Network]
  + It supports maximum cable length
  + It uses Ethernet cables for connection

###### There are two types of ISCSI

1. Software ISCSI
2. Hardware ISCSI
   * Using ISCSI initiators we can configure software ISCSI
   * We need dedicated HBA’s to implement Hardware ISCSI

ISCSI Devices are identified by IQN [ISCSI Qualified Name] IQN Format :: iqn.yyyy-mm-naming\_authority:unique\_name Example :: iqn.2012-05.com.microsoft.win32

IQN can be upto 255 characters

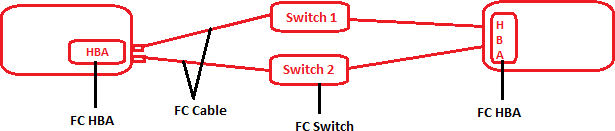


ISCSI Layer receives the command [CDB] from host and converts to PDU [Protocol Data Unit] and transfers across the TCP/IP network

At the target side ISCSI Layer converts back the PDU to CDB and sends to the target

### FC ::

* FC Stands for Fibre Channel
* FC also uses SCSI commands
* FC devices are identified by WWN
* It supports speeds upto 128GBPS [32GBPS FC]
* It is widely used in Remote Storage [SAN]



* FC Cable lengths are limited
* FC HBA’s are dual port compatible

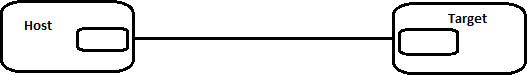
**Half – Page Gap**

#### Storage Topologies:

1. DAS [Direct Attached Storage]
2. SAN [Storage Area Network]
3. NAS [Network Attached Storage]

### DAS ::

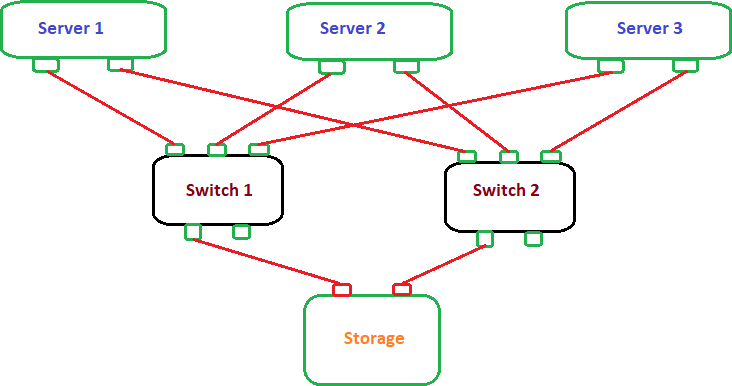
* + DAS Stands for Direct Attached Storage
  + We will connect server and storage directly with the help of Cables



* + It supports Protocols like SATA, SCSI, SAS, Nvme
  + It is a local storage
  + The implementation of DAS is simple
  + DAS is a Block Level Protocol
  + Performance of DAS is good

### SAN ::

* + SAN Stands for Storage Area Network
  + We can connect Server and Storage together with the help of Switches
  + SAN is a remote storage



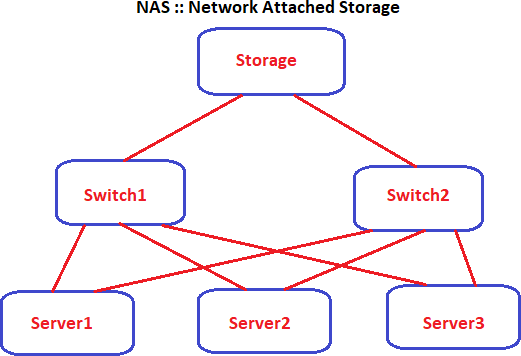
* + There will be a redundancy for the switches
  + It supports Protocols like ISCSI, FC and Nvme-OF and Nvme-OE
  + SAN is a Block Level Protocol
  + Implementation of SAN is costlier than DAS

**DAS :: Direct Attached Storage :: SAS, SATA, Nvme, SCSI SAN :: Storage Area Network :: lscsi, FC, Nvme-of**

**NAS :: Network Attached Storage**

### NAS ::

* + NAS Stands for Network Attached Storage
  + It is also widely used in Remote Storage
  + It supports protocols like NFS and CIFS
  + NFS stands for Network File System and it is used in Unix Flavoured operating systems [Linux]
  + CIFS Stands for Common internet File System and it is used in Windows operating system
  + The implementation of NAS costlier
  + NAS is a File Level Protocol / File Level IO / File Level Storage



#### \*\*\*Block Level Storage ::

* In Block Level Storage we assign Raw storage space to the user [Server] and user can format with his required file system and use it

Ex :: DAS and SAN

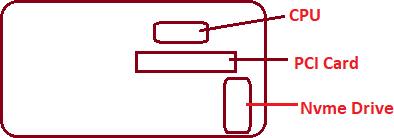
#### \*\*\* File Level Storage::

* In File Level Storage we create file system on volumes and will share storage space as a files across the network.

Ex :: NAS

#### \*\*\* Nvme ::

* + Nvme stands for Non-Volatile Memory Express
  + It uses PCI [Peripheral Component Interface] Interface \*\*\*

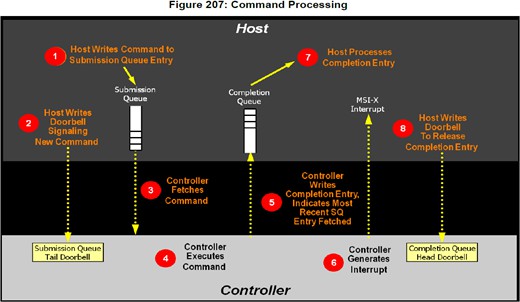


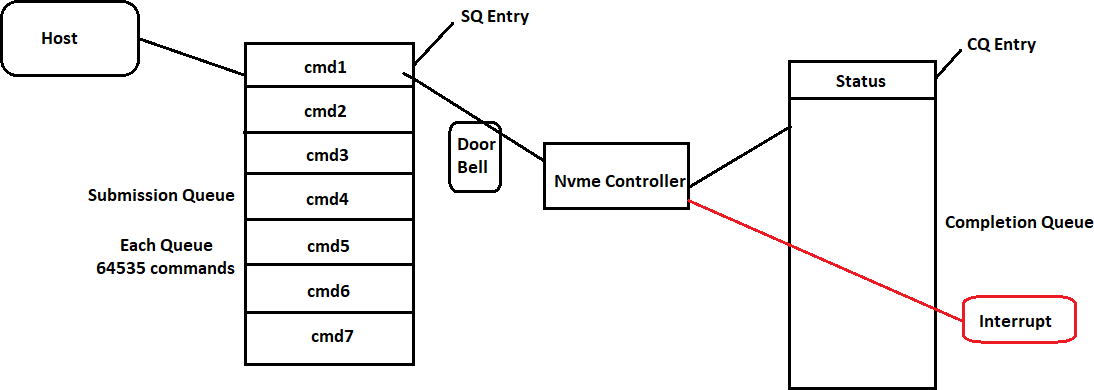
* + PCI interface makes Nvme much faster than SAS and SATA Protocol
  + Nvme drives are available as only SSD’s
  + Nvme is 10 times faster than SAS
  + Latency of Nvme devices are lessthan SAS and SATA
  + Throughput of Nvme is more than SAS and SATA
  + IOPS of Nvme is more than SAS and SATA
  + Nvme having its own command sets
  + Nvme devices are directly connected to PCI slots \*\*\*
  + There are two types of Nvme commands are there
  + To list Nvme devices we use “Nvme list”
  + The current of Nvme Spec is 2.0 [Nvme 1.4]
  + Nvme works based on Queues

There are two types of Queue’s

1. Submission Queue
2. Completion Queue

###### \*\*\*\* How Nvme Works?





**Nvme Works based on Queues HOW NVME WORKS**

1. Host writes the data in Submission Queue [SQ Entry] and rings the door bell
2. Nvme Controller Fetches the command from Submission Queue
3. Nvme Controller executes the command [Writes in NAND]
4. Nvme Controller writes the status in Completion Queue [CQ Entry] and generated the interrupt [MSI-X interrupt]
5. Host process the command status from Completion Queue and releases Completion Queue entry

* Nvme Supports up to 64535 Queues and each Queue supports up to 6435 commands
* Queue depth of Nvme is 64535 Queues
* IOPS of Nvme is better than SAS and SATA
* Throughput of Nvme is more than SAS and SATA
* Nvme is 10 times faster than SAS Protocol

There are two types of commands in Nvme

* 1. Admin Commands
     1. Admin Submission Queue
     2. Admin Completion Queue
  2. IO Command
     1. IO Submission Queue
     2. IO Completion Queue

###### Admin Commands

1. Create IO Submission Queue
2. Delete IO Submission Queue
3. Create IO Submission Queue
4. Delete IO Submission Queue
5. Get log page :: To get log pages
6. Identify :: To identify the device in Nvme domain
7. Abort :: To stop previously executed command
8. Set features :: To set the features
9. Get features :: To get the supported features
10. Firmware image download :: To download the firmware image
11. Firmware image activate :: To activate firmware
12. Firmware Commit :: To commit the changes [Before activate we need to send this command]

###### IO Commands

1. Read :: Reading from Flash memory
2. Write :: Writing to Flash memory
3. Flush ::
4. Write Zeros :: Writing zeros in flash memory
5. Compare
6. Nvme Format :: Secure erase of data

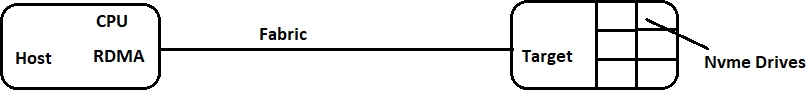
#### Namespace::

* + Namespace is a amount of storage space
  + It is a quantity of non-volatile memory that can be formatted into logical blocks
  + Similar to volume in SAS/SCSI/SATA in Nvme we call it as Namespace
  + Namespaces can be identified with Namespace ID [NSID]

1 page gap

#### Nvme-of [Nvme over fabric]

* + Nvme-of is a transmission protocol. It is used to transfer data across the network
  + Nvme-of uses Nvme commands and it uses fabrics to use Nvme in remote storage
  + It uses RDMA Protocol [Remote Direct Memory Access]
  + In RDMA memory to memory happens and thus reduces the burden on CPU



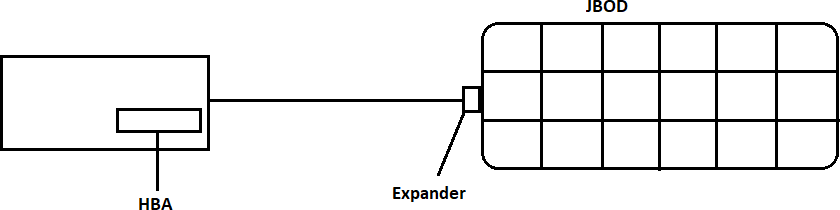
* + Nvme-of is more reliable than ISCSI and FC Protocols
  + Latency of commands in remote storage reduced by using Nvme-of Protocol

#### Types of storage Devices::

1. HDD
2. SSD
3. JBOD
4. Storage Array / EBOD [Extended bunch of Disks]

### JBOD ::

* + JBOD Stands for just bunch of disks
  + JBOD is a storage enclosure [box] where it contains group of drives

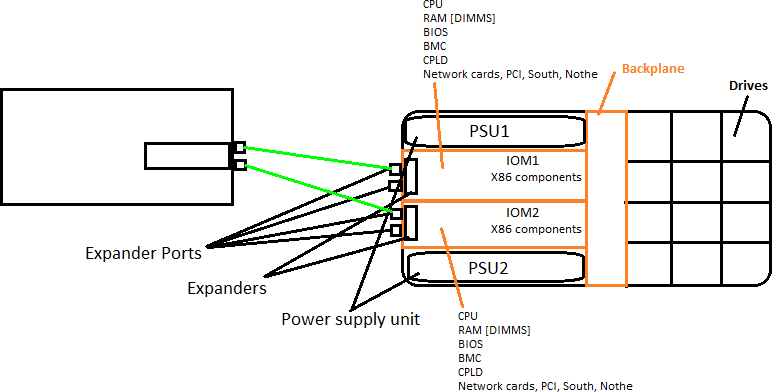




* + It provides additional storage capacity to the server
  + JBOD is not an intelligent device
  + We need to connect JBOD with HBA to make it as intelligent device
  + JBOD contains expander and drive slots

#### Storage Array ::

* + Storage array is a device and it can hold group of drives
  + Storage Array contains operating system and X86 components [CPU, RAM, BIOS, BMC, CPLD]
  + It contains no. of nodes /IOMS / controllers



#### Components of storage Array ::

1. Drives
2. Backplane
3. IOM’s / Controllers / Nodes / Modules Components of IOM / Node / Module

CPU, RAM, ROM, Expander, BIOS, BMC, CPLD, Expanders, Network ports, South bridge, North bridge, DIMMS, PCI slots.

1. PSU
2. Network ports

#### Features of Storage Array ::

1. Multipath
2. HA
3. Takeover and give back / Failover and Fail back
4. Remote management

###### Multipath ::

* + Multiple paths to the same target
  + If one path fails we can send the data from anther path

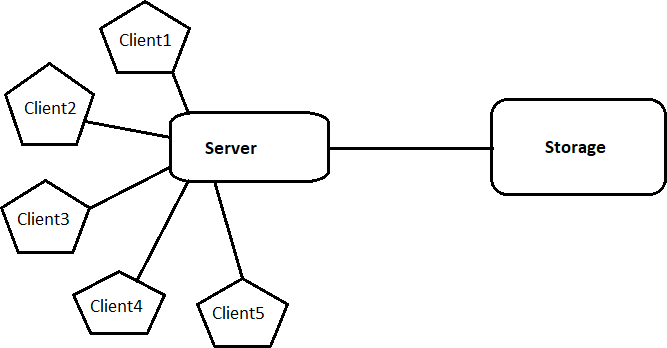
###### HA :: High Availability

* + HA Stands for High Availability
  + HA provides of continues access to storage for users
  + Multipath is a main feature for HA to work
  + Incase of one IOM goes bad the other IOM takeover the IO request that are coming from host
  + Two IOM’s will be in a sync with each other
  + They acts like a master and slave. If master IOM fails slave becomes master and takeovers the IO requests
  + This configuration we call it as Active – Passive configuration

###### Other storage configurations

Active – Active storage Controllers :: two IOMS acts a master

Active – Passive storage Controllers :: One IOM acts a master and other One acts as slave



**\*\*\* Servers \*\*\***

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

* Server is a physical device which serves the client request
* Server is nothing but a high end version of laptop
* Laptop/desktop OS is different than Server Operating system
* Server without the operating system called as **bare metal server**
* Server is used to connect and communicate with external storage devices [JBOD, Storage array] There are different types of servers

1. Rack Server
2. Blade Server
3. Tower Server

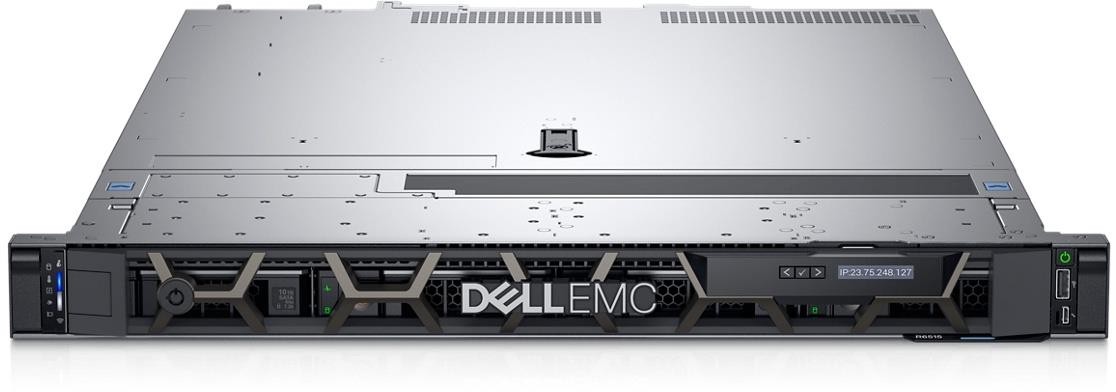
Rack Server



Tower Server



Blade Server



###### Form Factor of Server::

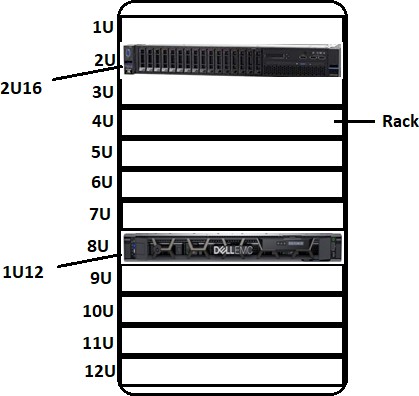
Form factor or foot print

We measure server / JBOD / Storage Array from factor in UNITS 1U == 1.4 inches

###### Examples::

**2U24** :: 2 units and 24 drive slots **5U84** :: 5 units and 84 drive slots **4U114** :: 4 units and 114 drives

Rack is a place which is used to hold storage enclosures / JBOD’s / EBODS and servers



#### Server Components or Server Architecture::

Processor [CPU] RAM [DIMM]

Drives Mother board FAN

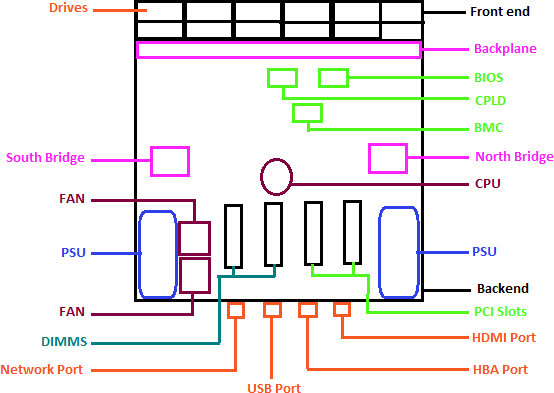
Power supply unit PCI Slot

N/W ports USB Port BIOS

BMC CPLD

Backplane South Bridge North Bridge CMOS battery

#### Server Architecture::



**Drives::** To store the data

**Backplane::** To connect / extend devices

###### CPU / Processor::

**PSU::** [Power supply unit] it is used to supply power to all the components in a server **DIMMS::** [Dual Inline Memory module] which is used to insert RAM [Random Access memory] **Southbridge::**

**Northbridge** ::The northbridge is one of the two chips, or integrated circuits (ICs), within the chipset on the motherboard. The other chip is called the Southbridge

**PCI Slot::** [Peripheral component Interface] used to connect RAID controllers or HBA cards to extend the storage capacity

### \*\*\* BIOS ::

* Basic input output system
* BIOS Stands for Basic input and output system
* Whenever we power on the sever the first control goes to BIOS
* BIOS performs POST [Power-On Self Test]
* It checks all the devices in the server are coming up or not. If any device is not coming up it will through the POST code Error
* BIOS is a intelligent device and it has its own firmware
* We can press F2 and F12 while system is booting to go the BIOS
* It loads the boot loader [MBR > Master boot record]

###### POST Codes

0000 :: Time issue 0003 :: Battery Low 000C :: RAM Failed 000E :: Drive Failed

8601 :: BMC Not responding

8102 :: USB port failed to initialize 00SD :: SMART Error

Based on POST Codes we can identify the type of issue

###### There are 2 types of BIOS

1. BIOS [Legacy BIOS]
2. UEFI [Unified Extensible Firmware Interface] UEFI

IUEFI Stands for Unified Extensible firmware Interface

It is a advanced of BIOS and acts as a interface between OS and firmware UEFI based systems boots faster than Legacy BIOS

###### \*\*\* Difference between BIOS and UEFI

|  |  |
| --- | --- |
| BIOS stands for Basic Input and Output System | UEFI Stands for Unified Extendible Firmware Interface |
| BIOS provides slower boot time | UEFI provides Faster Boot time |
| BIOS uses MBR partition Table | UEFI used GPT partition Table |
| BIOS supports Drive Size up to 2.2 TB | UEFI Supports drive size up to 9 ZB |
| Systems with BIOS are less secure as it don’t have  Secure boot feature | Systems with UEFI are more secure as it is having  “Secure boot” feature |
| BIOS store information about initialization in firmware | UEFI stores information about initialization and starts up in **.efi file** |

**CPLD:: Complex Programmable Logic Device**

* + CPLD gets power from PSU and it supplies to all the sensors in motherboard

#### \*\*\* BMC:: Baseboard Management Controller

* + BMC Stands for baseboard management controller
  + BMC is a small chip that is embedded in the motherboard

###### BMC is mainly used for remote management of a server /Storage array

* + Without login to the server /storage array we can control them by using BMC

###### Using BMC we can monitor the health and status of sensors and all the components of a server

**/storage array**

* + If CPU gets crashed also, we can login to the system via BMC
  + BMC is connected to the CPU **via 12C Bus**

#### How remote management is possible

BMC have multiple interfaces to control the system

1. IPMI [Intelligent Platform Management Interface]
   1. IPMI is a protocol that enabled the communication between CPU and BMC
   2. It provides management and monitoring capabilities independently of the CPU
   3. IPMI interface uses IPMB protocol
   4. The current version of IPMI is IPMI V2.0
   5. There are 2types of IPMI interface are available
      1. inband
      2. out of band
2. **Inband**::

If we execute the commands directly on BMC it is called as in band

###### Out of band ::

If we execute the commands via LAN it is called as out of band IPMI tool it provides access to the BMC [in band and out of band]

###### You can do the following with IPMI tool:

* Read the Sensor Data Record [SDR] repository
* Print Sensor values
* Display the contents of the system event log [SEL]
* Print field – replaceable unit [FRU] inventory information
* Read and set LAN configuration parameters
* Perform remote chassis power control

IPMItool is a tool its provides access to the BMC [Inband and Out of band]

**IPMItools Inband commands**

ipmitool chassis status :: To check enclosure / Server status

ipmitool chassis power [on|off|cycle] :: For power related operations ipmitool fru print :: To print all the components in a server

ipmitool sensor list :: It lists all the sensor related info ipmitool sdr list :: To get logs about sensors

ipmitool sel list :: To get all the system event logs

###### IPMItool out band commands

Ipmitool – H <IPaddress> - | lanplus –u <username> - p <password> chassis power [on|off|cycle] :: power operations

Ipmitool – I lanplus set ipaddress-u <username> - p <password> :: To set the IP address

1. **System Information**:
   * **ipmitool mc info**: Displays information about the BMC (Management Controller).
   * **ipmitool lan print 1**: Shows LAN configuration settings.
   * **ipmitool sdr list**: Lists sensor data records (hardware sensors).
   * **ipmitool fru print**: Displays FRU (Field Replaceable Unit) information.
2. **Power Management**:
   * **ipmitool power status**: Shows the power status of the system.
   * **ipmitool power on**: Powers on the system.
   * **ipmitool power off**: Powers off the system.
   * **ipmitool power reset**: Resets the system.
   * **ipmitool power cycle**: Power cycles (turns off and then on) the system.
3. **Sensor Monitoring**:
   * **ipmitool sensor**: Displays sensor readings (temperatures, voltages, fan speeds, etc.).
   * **ipmitool sensor thresh**: Shows sensor thresholds and settings.
   * **ipmitool sdr**: Prints sensor data repository entries.
4. **Event Log**:
   * **ipmitool sel list**: Lists entries in the System Event Log (SEL).
5. **User Management**:
   * **ipmitool user list**: Lists BMC users.
6. **Remote Console**:
   * **ipmitool sol activate**: Activates Serial Over LAN (SOL) for remote console access.
   * **ipmitool sol deactivate**: Deactivates SOL.
   * **ipmitool sol info**: Shows SOL configuration information.
7. **Firmware Updates**:
   * **ipmitool mc update image <file>**: Updates BMC firmware with the specified image file.
   * **ipmitool mc reset cold**: Performs a cold reset of the BMC.
8. **Remote Management**:
   * **ipmitool chassis status**: Retrieves the chassis status.
   * **ipmitool chassis identify [seconds]**: Identifies the chassis by blinking the front panel LED.

# \*\*\* Testing \*\*\*

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

Testing is a process of verifying any software or product is working as per expected behavior or not

**Code** :: Developer [Develops the product]

**Testing** :: Testers of QA Engineers or Test engineers, Storage QA

**Support** :: Technical Support Engineers

1. Developers develop the software
2. Testers tests the software or product
3. Technical support engineers will support the customers

#### Ways of Testing ::

1. Manual Testing
2. Automation Testing

###### Manual Testing::

It is a process of verifying product or software manually to check it is working as per expected behavior or not

Or

Executing the test cases without depend on any tools

###### Advantages ::

1. For testers no need of programming language
2. Best way to cover manual intervention testcases

###### Disadvantages ::

1. It takes more time
2. Less Efficiency

###### Automation Testing ::

It is a process of verifying the product or software by using scripts or tools to check it is working as per expected behavior or not

Or

Executing the testcases with the help of a tool or script

###### Advantages ::

1. It is more efficient
2. It takes less time [Save time]
3. Less manual intervention

###### Disadvantages ::

1. Implementation cost will be high
2. 100% test coverage won’t be code

#### Why we do Testing?

* + To find the issues in developers code or product / software functionality
  + To increase the product / Software Quality

#### Methods of Testing

1. White Box Testing
2. Black Box Testing
3. Grey Box Testing

#### White Box Testing::

* + It is a process of testing to find the issues or bugs in software code or developers code
  + Testers must have knowledge on developers code

###### Advantages::

* + It helps in optimizing developer code

###### Disadvantages::

* + Due to the requirement of skilled people [Skilled Testers] the implementation costs will be increased

#### Black Box Testing::

* + It is a process of testing to find the issues or bugs in software product functionality

###### Advantages:

* + Testers need not be having knowledge on Developers code
  + Developer code is not required for the testers

###### Disadvantages:

* + Inefficient testing will be done

#### Grey Box Testing::

It is a method of testing to find the issues in both developer’s code and product functionality Testers should have knowledge on the both developers code and product functionality

###### Advantages:

* + It has a combined benefit of Whitebox and Blackbox testing
  + Efficient testing will be done [Test coverage will be more]

###### Disadvantage:

* + Skilled engineers are required
  + It take more time to complete

#### Testcase::

It is a step by step approach of verifying any software or product is working as per expected behavior or not.

###### Fields of a testcase / Components of a testcase

1. S.No
2. Testcase ID
3. Summary :: Verifying google chrome functionality
4. Steps
5. Expected Results
6. Actual Results

Example of Testcase / Testcase Template

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S.No** | **Testcase ID** | **Summary** | **Steps** | **Expected Results** | **Actual Results** |
| 1 | TDS-123 | Test to Verify Google chrome home page | Turn on the PC | PC should power on |  |
|  |  |  | Navigate to Google chrome | Google chrome icon should be visible |  |
|  |  |  | Click on Google Chrome | Google chrome should open |  |
|  |  |  | Type google.com | Should display search bar |  |

* + If all the steps are working as per expected behaviour, we will pass the testcase
  + If at least one step is not working as a expected behaviour we need to fail the testcase and raise the bug

#### Defect or BUG::

* BUG :: It is a deviation from Expected behaviour
* Testers will raise bugs and developer will fix the testers raised defects/bugs

#### How to write testcases::

###### Functionality verification document or product verification document ::

* This document contains the in detail explanation of each functionality of a product or software or application
* Tester will write the testcases by referring this document

#### Test Scenarios::

* We will write the testcases from test scenario
* Test scenarios can be derived from functionality verification documents
* Test scenarios are created to make sure that every single feature covered on the testing and converted to testcases

###### Example::

Test Scenario:: Verifying Google Chrome Functionality

**Testcase::** Check refresh button functionality Check Home button functionality

#### Test Suite:: [Group of Testcases]

* Test suite is a collection of testcases those are used to test the software or product

**6 lines Gap**

#### SDLC [Software Development Life Cycle]::

SDLC Stands for Software Development Life Cycle

1. Idea



1. Design



1. Coding



1. Testing



1. Release



1. Support



1. End of Life

##### Idea::

* + Any software product or application should start with idea

##### Design::

* + Once the idea is approved architects will design the product

##### Coding::

* + In coding phase developers will develop the code

##### Testing::

* + In testing phase tester will do the end to end testing to make sure that product is working fine and it has zero % defects

##### Release::

* + Once the testing is done marketing team or owners will release the product into market

##### Support::

* + Once the product is released into the market support team will resolve the customer issues

##### End of life [EOL]::

* + After some years product will reach end of life and no support will be provided

#### \*\*\* STLC [Software Testing Life Cycle] \*\*\*

STLC Stands for Software Testing Life Cycle

1. Requirement Analysis



1. Test Planning



1. Test Design



1. Test Environment Setup



1. Test Execution



1. Test Reporting

#### Requirement Analysis:

* + In the Requirement Analysis phase testers analysis the customers and work with the developers to get the requirements which are testable

#### Test Planning:

In the planning phase all the planning about testing will be done

1. What to test and what not to test
2. What test strategy to be followed
3. What will be the test Environment
4. Resource required for testing’’
5. Test schedule / Test Estimation

##### Test Design::

* + In the design phase we will writhe the testcases either manual or automation testcase

##### Test Environment Setup::

* + In the phase we will build the testbed or test configuration for executing testcase

##### Test Execution::

* + In the phase we will execute the testcase / Scripts on the testbeds
  + We will get pass/Fail result after executing the testcase

##### Fail::

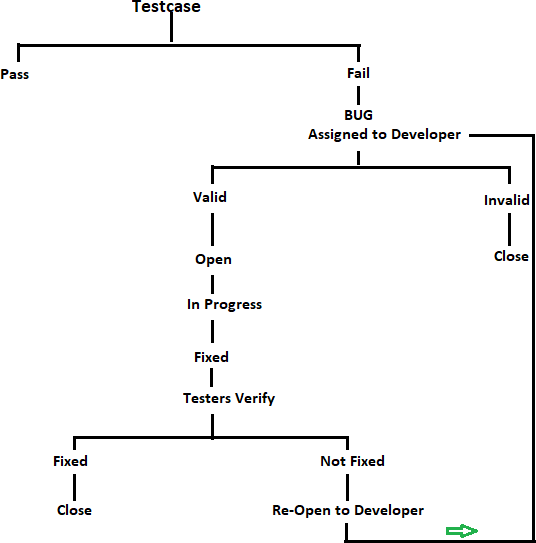
* + If the testcase is failed we will fail the test and raise a bug

##### Test Reporting::

* + We will report the test results in this phase
  + We will prepare test exit report and send to the customers / stake holders
  + We will prepare bug report [How many bugs/defects are raised and how many are fixed]

### \*\*\*\*\* BUG LIFE CYCLE::

**BUG::** Bug is a deviation from expected behaviour



1. After executing the testcase if it is failed testers will raise a bug
2. As soon as tester raise a bug it will be assigned to the developer and the state of the bug will be **new**
3. Once developer starts working on it, He analyses it is valid or invalid bug, If it is valid bug he will keep the state as **open**
4. If it is invalid he will reject the bug and closes
5. if it is valid and once developer starts working on it, he will keep the state as **inprogress**
6. After developer fixes the issue, they will change the state of the bug to “**fixed**” and release the build (code) to testers
7. Testers will take the fix and verify that defect is really fixed or not
8. If it is working as per expected behaviour testers will close the defect
9. If it is not working as per expected behaviour testers will re-open the defect and re assign it to the developer

#### States of BUG::

NEW OPEN

IN Progress Reject Close

Fixed Resolve Duplicate Won’t Fix Third Party

##### Duplicate::

If the same bug is already submitted by other engineer, developer keeps the state of the bug as duplicate

##### Won’t Fix::

If developer fixes the bug other major functionalities may not work

##### Third party::

If the issue is with third party application / software

#### BUG Tracking Tools:

* 1. JIRA
  2. BUGZILLA
  3. IMS
  4. BURT [Netapp]
  5. QC [HP]

##### Steps to file/raise a BUG

1. Log in to JIRA
2. Select Project
3. Click on Create
   1. Select Project [Our Project Name]
   2. Issue type [BUG]
   3. Summary [one liner information about bug]
   4. Reporter [Test engineer name whoever reports the BUG]
   5. Description [Whatever the steps followed to hit this issue] [issue reproduct]
   6. Assignee :: [Developer] [Leave it to auto assignee]
   7. Environment :: Testbed details [System configurations]
   8. Attachment :: We need to attach the logs [Proof for the bug]. Screenshots and system logs
   9. Priority ::
   10. Severity ::

We need to click on submit

**Once we submit the bug/defect one BUG/Defect ID will be generated**

##### Priority::

How fast the developer needs to fix the issue

1. Blocker
2. Critical
3. Major
4. Minor
5. Trivial

##### Severity::

How severely the bug is impacting the functionality of a product

1.1 :: Highest

2.2

3.3

4.4

5.5 :: Lowest

###### Examples::

High Priority and high Severity Low Priority and Low Severity High Priority and Low Severity Low Priority and high Severity

#### Test Tracking Tools::

1. Teststrack
2. Qtest
3. JIRA
4. HP ALM

* Test tracking tool is used to manage the testcases in one place
* We need to put the testcase in In progress state once we start it
* We need to update FAIL / PASS after the execution of testcase
* We can create testplans, assign it to engineers, update the results, fetch the previously executed testcases

#### \*\*\* Types of Testing::

1. Functional Testing
2. Non-Functional Testing
3. Functional Verification Testing
4. Regression Testing
5. Re-Testing
6. System Testing
7. Unit Testing
8. Sanity Testing
9. Smoke Testing
10. Performance Testing
    1. Load Testing
    2. Stress Testing
11. Adhoc Testing
12. Compatibility Testing
13. Acceptance Testing
14. Alpha Testing
15. Beta Testing
16. Monkey Testing

#### Functional Testing::

* Functional Testing is a type of software Testing in which we will test the product / software against functional requirements are satisfied as per the customer requirements or not

###### Example::

FVT, Unit testing, Smoke Testing, Regression Testing etc…..

##### Non-Functional Testing::

* Non-Functional Testing is a type of software testing in which we will test the behaviour of product

/ software under different workloads / circumstances

* Here we will test on the aspects that are not covered in functional Testing

###### Example::

Performance Testing, Load Testing, Stress Testing

##### Functional Testing or FVT [Functional Verification Testing]::

* In FVT we will verify all the functionalities of a product are working as per expected behaviour or not
* To make sure that all the functionalities are covered as per the requirements specifications we perform functional testing

or

* Testing every possible option in the product is working as fine or not is called as FVT

\*\*\* **BUILD::**

* It is a working software that developer will give to the testers
* Testers need to install the build before starting testing
* Once the build installation is successful testers will start the testing
* Whenever there is a change in code [BUG Fixes] developer releases the build to testers

##### \*\*\*\* Regression Testing::

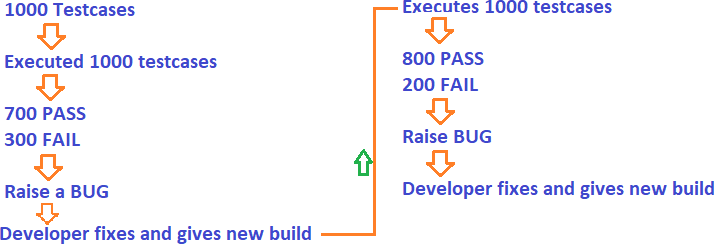
Executing the passed testcases with new build is called as Regression Testing

###### Why ::

To make sure that other working functionalities are not affected due to the code changes

##### When we will regression Testing::

Whenever there are code changes [New Build] we will do regression Testing



##### Re Testing::

* Executing the failed testcases with new build to make sure that BUG’s are fixed is called as re- testing
* Retesting is also called as defect/BUG/JIRA verification
* **Device Under Test or System under test [DUT or SUT]**
* The product which we are testing is called as device under test or system under test

##### System Testing::

* It is a level of software testing in which we will test the “end to end” features of a product as whole with respect to the Customer use cases
* How the customer user out product we will execute those kind of scenarios

##### Unit Testing::

* Unit testing is also called as **code level testing**
* In which developers will divide the source code into units and tests that is working or not before releasing the code [Build] to the testers
* It will be done by developers
* Once Unit testing passed only developers will release the build testers

##### Integration Testing::

* It is a level of software testing which individual units are combined together and tested as a group
* Integration starts with unit testing
* Either developers or testers can perform integration testing

#### \*\*\*Performance Testing::

* It is a level of software testing in which we will test the performance of a device under test [DUT] with different workloads

There are two types of performance Testing

1. Load Testing
2. Stress Testing

##### Load Testing::

* + It is one of the method of performance testing in which we will keep the load **within the supported level** and check how the product/system/software/application is performing

##### Stress Testing::

* + It is one of the type of performance testing in which we will keep the load **above the supported level** and check how the product/ system/software/application is performing

##### Sanity Testing::

* + Whenever there is a less time we will pick the important testcases and execute. This process is called as sanity testing

Sanity testing will be done by testers 100 Testcases

21 days

We will do complete Regression testing

100 Testcases

10 Days

We pick some important testcases and perform sanity testing

##### Smoke Testing::

* + It is a level of software testing in which we will check the stability of a software / product [Built] in initial stage
  + It will be done be Testers
  + It is also called as build verification testing
  + Here we will check build installation is successful or not
  + It will help us to find if any issues in the build in initial stage only
  + As soon as build is released into the testers the first testing they do is “Smoke Testing”

##### Compatibility Testing::

* + In compatibility testing we will test our product/software is working fine or not with different OEM’S [Original Equipment Manufacturers]

##### Acceptance Testing Or

**Basic Acceptance Testing [BAT] or User Acceptance Testing [UAT]::**

* + It is a level of testing in which we will test the product/software for acceptance
  + Customer checks all their requirements are satisfied or not

##### Alpha Testing::

* + Customer come to our environment and executes their testcases

##### Beta Testing::

* + Customer takes the product to their environment and executes their testcases

##### Adhoc Testing::

* + If we complete the assigned testcases early then we will do adhoc testing

Ex:: if we assign with 100 testcases to complete in 10 days but if we complete it in 9 days we will do adhoc testing

* + Adhoc testing is also called as un planed testing
  + When testing is performed without any proper planning and documentation is called as adhoc testing
  + After completing the formal testing on the product testers will do adhoc testing

##### Monkey Testing::

* + If the testing is performed without any knowledge on product is called as monkey testing

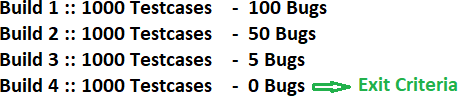
**Entry Criteria and Exit Criteria::**

##### Entry Criteria::

* + Entry Criteria tells us when to start the testing
  + Build installation should be successful to start with the test execution
  + **Build installation success is as an entry criteria for our testing**

##### Exit Criteria::

* + Exit Criteria tells us when to stop the testing
  + Once the product is bug free we can stop the testing



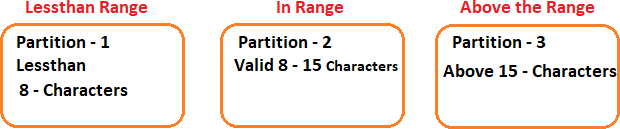
* + Once all the bugs are fixed, Our product has met exit criteria

#### Testcase Writing Techniques::

1. Equivalence partition Technique
2. Boundary Value Analysis
3. Error Guessing

##### Equivalence Partition Technique:

* + In equivalence partition technique we will divide the input into equivalent partition and derive the testcases

Ex: Gmail Password Minimum Characters :: 8 Maximum Characters :: 15

##### Boundary Value Analysis::

* + In Boundary Value Analysis we will take the boundary value of supported inputs and we will derive the testcases

**Ex:**

Password :: Min 8 characters

Max 15 characters



##### Error Guessing::

* + In this method based on the prior experience on testing, testers will guess the errors and write the testcases

###### Ex:

* If the tester is already worked on RAID features in their previous company / Project they can guess that R5 creation will fail with minimum of 2 drives and write the testcases accordingly

##### Positive Testing::

* + It is a type of testing in which we will provide the valid inputs and checks how the product is working

##### Negative Testing::

* + It is a type of testing in which we will provide invalid inputs and checks how the product is working

**The purpose of negative testing is our product [DUT] should not fail/crash and remains stable with the invalid inputs**

##### Traceability Matrix::

* + This is also called as requirement traceability Matrix [RTM]
  + The relation between the testcases and the requirements are shown in a document is called as Traceability Matrix
  + If testcase fails this document will help us to find the corresponding functionality easily
  + It also helps us to ensure that all the requirements are covered which are needed

**Example::**

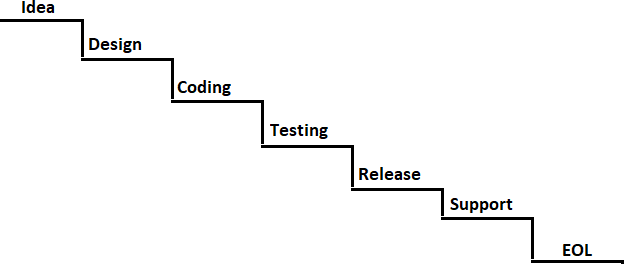
|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Requirement Description** | **Testcase ID** | **Status** |
| 1 | Test Scenario 1 | TC1, TC2, TC3, TC4 | TC1 Pass, TC2 Fail, TC3 Pass, TC4 Fail |

|  |  |  |  |
| --- | --- | --- | --- |
| 2 | Test Scenario 2 | TC5, TC6, TC7, TC8, TC9 | TC5 Pass, TC6 Fail, TC7, TC8, TC9 Not run |
| 3 | Test Scenario 3 | TC10, TC11, TC12 | TC10 Pass, TC11 Pass, TC12  Fail |
| 4 | Test Scenario 4 | TC13, TC14, TC15 | TC13 Pass, TC14 Fail, TC15  Not run |

#### \*\*\* Models of Testing::

1. Water Fall Model
2. V-Model
3. Agile Model

##### Water Fall Model:



* + It is a step by step approach of following software development life cycle [SDLC]
  + After one step completes only, we can proceed to next step.

###### Advantages::

1. Easy to understand
2. Each step is well planned
3. Proper handover and takeover happen
4. Well suited for small projects

###### Disadvantages::

1. Not suitable for big projects
2. We can’t change the design once it is moved to next phase
3. It consumes more time

##### 3. Agile Model::

* The software testing that follow the principles of agile software development is called as Agile Model
* Agile is an interactive development where requirements involve through collaboration between the customers and organizations [Who develops the product or software]

\*\*\* Here coding and testing happens parallelly

###### Components or features of Agile:

1. Sprint
2. Scrum
3. User story
4. Subtask
5. Backlog
6. Story points
7. Scrum Master
8. Sprint Review Meeting
9. Scrum Meeting [Daily stand-up meeting]
10. Retrospective Meeting

##### Sprint::

* + Sprint is a duration of time to complete assigned task
  + Generally, sprint document will be 2 to 3 weeks
  + A project contains multiple sprints one by one can done.

##### Scrum::

* + It is frame work [Set of standards] in agile for developing, managing and delivering the complex project.
  + The project will follows scrum in agile is known scrum project. In this they contains multiple sprint

##### Scrum Master::

* + Is a person within the team who is responsible for creating sprints and conducting scrum meeting [Daily stand-up meeting] and discussing with the stake holders [Customers or high-level employees]?
  + He/she also responsible for creating user stories and assign it to all the engineers

##### Scrum Meeting::

* + It is also called as daily stand-up meeting
  + In which we will discuss about the status of assigned work and if any issues/blockers that are

stopping our test execution

* + Developers and testers will have separate scrum meetings
  + Scrum master conducts the scrum meeting
  + All the team members will be part of meeting [Including manager, team lead, Scrum master]

##### Sprint Review Meeting or Sprint Planning::

* + This meeting happens before starting the sprint in which scrum master, product owner/ team lead manager will join and discuss about the sprint plan [What work to be assigned to each engineer]

##### User Story::

* + It is a template where it contains the type of work that is assign to team
  + In a sprint if scrum master assigns the same work they will create a story for each type of work

###### Ex::

User story 1 :: execute 20 Testcases

User story 2 :: Write automation scripts for 10 Testcases

User story 3 :: Go through the product verification documents

##### Subtask::

* + Subtask is a work that is part of user story, Each engineer will create subtask for their assigned work.

**Ex::** In a 2 weeks sprint [80 hours]

Subtask 1 :: 30 hours executing testcases Subtask 2 :: 30 hours writing automation scripts Subtask 3 :: 10 hours meetings

Subtask 4 :: 10 hours going through the documents

##### Story Points::

* + For each user story scrum master / manager assigns the story points depends on the priority / importance of the particular work

##### Backlog::

* + Backlog is a remaining work from the current sprint that can be carried to next sprint

##### Retrospective Meeting::

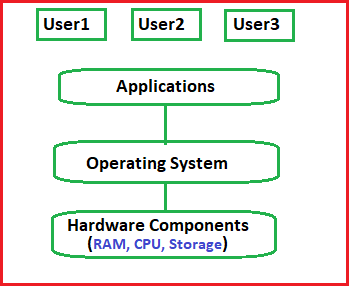
* + This meeting happens after the sprint [Last date of sprint]
  + In which we will be discuss how the previous sprint has happened [What went right and What went wrong]
  + And if there are any improvements are needed for planning the next sprint

# \*\*\* Linux \*\*\*

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

##### OPERATING SYSTEM ::

Operating System is an interface between user and the computer hardware



##### Type of Operating System::

There are two types of Operating Systems

1. Single User

Only one user can login to the system at a time Ex: Windows

1. Multi User

Multiple users can login to the system at a time Ex: Linux

###### LINUX ::

LINUX is a flavor of UNIX Operating System

Red hat Linux [RHEL] Cent OS

Ubuntu SUSE Linux

Scientific Linux

Debian Fedora HP-UX

Solaris

GUI [Graphical User Interface] CLI [Command Line Interface]

>>>> Difference between GUI and CLI [3 Points] Linux is available in both GUI and CLI **Difference between Windows and Linux**

|  |  |
| --- | --- |
| **Windows** | **Linux** |
| Windows is a single user OS | Linux is a multi user OS |
| Windows is a available only in GUI | Linux is a available in both GUI and CLI |
| Windows is not open source Operating System | Linux is an open source Operating System |
| Need to install Anti Virus | No Need to install any Anti Virus software’s |
| Windows is less secured than LINUX | Linux is more secured than Windows |
| Windows is Case Insensitive | Linux is Case Sensitive |

Minimum Requirements for Installing Linux OS Processer :: Inter / AMD Dual Core

RAM :: 8 GB

Storage :: 50 GB **Basic Commands ::** File Operations

To create a file

**touch** :: used to create a file Syntax :: **touch <file\_name>**

Ex: touch file1

touch file1 file2 file3 file4

Listing the files “**ls**” >> To list files Syntax ::

**ls ::** lists all the files Example ::

ls file 1

###### Delete/remove a file

‘”**rm**” >> delete a file

Syntax :: rm <file\_name>

###### Rename a file::

“**mv**”

Syntax :: **mv <file\_name> <new \_file\_name>**

Example :: mv file1 to file10

###### >> How to write data into file or list content in a file

To list content of a file cat < file\_name> Example:: cat file1

###### To write data into a file

1. **cat >** file\_name [overwrite]

**cat >>** file\_name [Appending the data] with **“>”** it overwrites the data

with **“>>”** it appends the new data at the end of a file [After the previous data] If file does not exist, it will create a new file

1. “**vi editor**” or “**vim editor**”

To edit and create files in Linux Syntax::

###### vi <file\_name>

Example::

vi file1

To open:: vi file1

To edit [Press “**a**” or “**l**” or “**insert**”]

To save changes >> **“esc”:w!**

To save and exit >> **“esc”:wq!**

To exit without changes >> **“esc”:q!**

To go to the end of line >> **“press end”**

To go to the end of file >> **“press ctrl + end”**

To search for particular string >> **“/string”**

To search and replace >> **:%s/string1/string2 >> to change one occurrence**

###### :%s/string1/string2/g >> to change globally [ All the occurrence]

To delete line >> **[ Press “esc” + dd] [two times “d”]**

To undo changes >> **press “u”**

###### Directory

Directory is a group of files

Windows folder is nothing but Linux directory

Directory Handling To create a directory Syntax::

###### mkdir <dir\_name>

Example:: mkdir dir1

###### To rename a directory

“**mv” <dir\_name> <new\_dir\_name>**

Example::

mv dir1 dir2

To delete a directory Syntax::

**rmdir <dir\_name>** [To remove empty directory]

**rm –rf <dir\_name>** [To remove if files are there forcefully]

Change directory syntax::

###### cd <dir\_name>

Example ::

cd dir2

**pwd** :: Present working directory

“**pwd**”

**cd** :: Change Directory

###### mkdir dir1

**cd dir1 ::** To go the dir1

**cd..** :: to come one step backward

**cd -** :: to go the previously working directory

**cd ~** :: to go to the home directory

**cd /** :: to go to the root directory

##### Copying files and directories

**“cp” ::** used to copy files and directories from one place to another place

To copy files Syntax ::

###### cp <source\_file> <destination\_directory>

Example ::

**cp** file1 dir1 :: to copy file1 to dir1

To copy directory :

**cp –r** <source\_directory> <destination\_directory> Ex ::

**cp -r** dir1 dir2 :: to copy to dir2

**ls** :: To list the files and directories

**ls -l** :: prints long list about the files

root@rhel:~#ls -l

total 0

drwxr-xr-x. 2 root root 20 Jun 24 15:38 copy\_dir

drwxr-xr-x. 3 root root 18 Jun 24 15:23 copy\_dir

**ls -a** :: To list the hidden files

root@rhel:~#ls -a

. .bash\_history .bash\_profile .config .cshrc dir12 file1 file11 .ssh .vimrc

.. .bash\_logout .bashrc copy\_dir dir1 dir11 dir13 file10 new\_dir .tcshrc

**ls -s** :: To list the files with size

root@rhel:~#ls -s

total 4

0 copy\_dir 0 dir1 0 dir10 0 dir11 0 dir12 0 dir13 4 file1 0 file10 0 file11 0 new\_dir

**ls -h** :: To list files in human readable format

root@rhel:~#ls -sh

total 4.0K

0 copy\_dir 0 dir1 0 dir10 0 dir11 0 dir12 0 dir13 4.0K file1 0 file10 0 file11 0 new\_dir

**ls -t** :: To list files by modified time

root@rhel:~#ls -lt

total 8

-rw-r--r--. 1 root root 20 Jun 24 15:59 file10

-rw-r--r--. 1 root root 0 Jun 24 15:59 file111

-rw-r--r--. 1 root root 0 Jun 24 15:59 file100

-rw-r--r--. 1 root root 96 Jun 24 15:55 file1

###### ls -alrt ::

**ls -r** :: To print in reverse order

**ls \*** :: to print all the files and directories in a current working directory

**ls \*.py** :: to list all the files ending with .py

**ls test\*** :: all the files starting with test

**rm -rf \*** :: to remove all the files forcefully

**rm -rf \*.py** :: to remove all the files ending with .py

**cp \*.py dir1** :: to copy all the files ending with .py to dir1

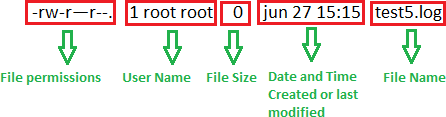
##### ls -alrt command explanation

root@rhel:~/dir1# ls –alhrt total 4.0k

-rw-r--r--. 1 root root 0 jun 27 15:15 test5.log

drwxr-xr-x. 2 root root 6 jun 27 15:21 dir1

If the starting letter is “d” then it is a directory



###### “help” command and man pages

To get help or manual pages for any command Syntax::

<command> --help

###### Man <command>

Examples ::

ls -help Man cp

**head and tail command [Filter commands] head** :: to print first **“n”** lines

syntax ::

###### head -n file\_name

Example ::

root@rhel:~/dir1# head -2file1 line1

line2

###### tail

To print last **“n”** lines

Syntax::

###### Tail -n ‘file\_name”

Example ::

root@rhel:~/dir1# tail -2file1 line 14

line15

###### tail -f file1

It prints last “n” lines and tells the last content [lines] of a file in real time usage of file

**sort** ::

To sort the file content in ascending order syntax ::

###### sort <file\_name>

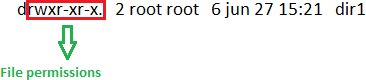
Example :: sort file2

###### File Permissions

root@rhel:~/dir1# ls –alrt

-rw-r—r--. 1 root root 0 jun 27 15:15 test5.log

drwxr-xr-x. 2 root root 6 jun 27 15:21 dir1



## \*\*\* File Permissions

There are three types of permissions will be there for file in linux Access modes

**Read :: r :: 4**

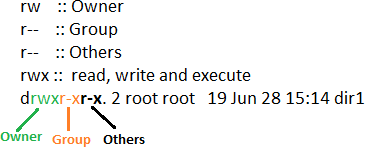
**Write :: w :: 2**

**Execute :: x :: 1**

1. Owner
2. Group
3. Others

root@rhel:~/dir1#ls –alrt file1

-**rw-r--r--.** 1 root root 28 Jun 28 15:14 file1



**How to change the file permissions “chmod” ::** To change the file permissions Syntax:: **chmod <permissions> file\_name** Chmod <owner,group,others> file\_name

Example::

Chmod 471file\_name Chmod 444 file\_name

Chmod 777 file\_name :: to give complete permissions to all the users Chmod 333 file\_name

Chmod 111 file\_name :: To give executable permissions to all the users

root@rhel:~dir1#ls -alrt

-rw-r--r--. 1 root root 28 Jun 28 15:14 file1

root@rhel:~/dir#chmod 400 file1 root@rhel:~/dir1# ls -alrt

-r -. 1 root root 28 Jun 28 15:14 file1

##### How to change the owner for particular file?

“**chown**” to change the owner name for file

Syntax::

###### Chown <new\_user\_name> <file\_name>

Chown user1 file1

To list all the users in linux :: **users**

To add the user :: **useradd <user\_name>**

To remove user :: **userdel <user\_name>**

Examples::

**useradd test1** :: to add new user “test1”

**userdel test1** :: to delete user “test1”

###### Inode

Inode is a data structure in linux Operating System which stores the information about files and directories, such as ownership, access modes [read, write, execute] permissions and type of file

OS can recognize the file/directory by Inode number

Whenever we create a file Inode number will automatically generated

**How to get the Inode** :: **ls -i <file\_name>** Example::

ls -i file1

root @rhel:~/dir1# ls -i file\_inode 50601924 file\_inode

30-06-22

Soft Link and Hard Link [Symbolic Links] Windows shortcut files are linux links

###### Soft Link::

Soft link for a file is equal to the shortcut file of windows

If Original file gets deleted soft link will be broken and data will be lost Inode of soft link and Original file are different

###### How to create a Soft Link ::

Syntax::

###### ln -s <Original\_file> <soft\_link\_file>

Example:

**ln -s file1 file2** file1 :: Original file file2 :: Soft Link file

###### Hard Link::

Hard Link is a backup copy of original file

If we delete original file hard link file will be there only Inode of hard link and original file are same

###### How create Hard Link::

Syntax::

###### ln <Original\_file> <Hard\_link\_file>

Example::

**ln file3 file4 file3**:: Original file

**file4**:: Hard link file

##### Difference between soft Link and Hard Link

|  |  |
| --- | --- |
| **Soft Link** | **Hard Link** |
| Short cut file | Backup copy |
| If Original file gets deleted soft link will not be accessible | If Original file gets deleted hard link will be there only |
| Inode will be different for original file and soft link  file | Inode will be same for original file and hard link file |
| Soft Link works for both directories and files | Hard Link works for only files |

###### Search and Replace of a word in a file

* 1. vi editor
  2. sed

###### vi

Syntax::

Esc

###### :%s/originalword/replaceble word/

Example::

###### :%s/python/linux/g

1. **sed**

sed stands for stream editor

with sed, without opening a file we can search and replace the particular word Syntax::

###### sed ‘s/original\_word/replace word/g’ file\_name

Example::

###### sed ‘s/python/linux/g’ file1

**User Management::**

Linux is a multi-user operating system

* 1. To know how many users are registered to login with linux machine

###### users

* 1. To know the current logged in user name

###### whoami

* 1. To add a new user

###### useradd <new\_user\_name>

Example::

###### useradd user1

* 1. To delete the user

Syntax :: **useradel <user\_name>**

Example:: **userdel user1**

* 1. To change [Switch] the user

**su** [Switch User]

Syntax :: **su <user\_name>**

Example:: **su user3**

* 1. To change the password

**passwd**

## IO Redirection::

01-07-2022

It means storing the output of command into a file

1. By using “>>”

syntax::

###### cmd>> “file10”

example::

ls >> file10

1. “**tee**”

“**tee**” is used to redirect the output from one command to a file “**|**” >> PIPE

Pipe will take output of first command [left side command] and sends as a input to the next command [Right side command]

syntax::

###### command | tee file\_name

Example ::

ls -alrt | tee file10

###### \*\*\* grep :: [pattern matching]

grep searches al file for a particular pattern of characters or keyword and displays the lines that contains the pattern/keyword

syntax::

###### command | grep <pattern>

Example::

###### cat file1 | grep “python”

**grep -i** :: “-i” is for ignore case [Makes case insensitive]

Example::

###### cat file1 | grep -i “python”

It lists both lower case and upper case matched patterns

###### find command ::

find command is used to find the files or directories path

syntax::

###### find <directory> -name file or dir name

**<directory>** :: in which directory we need to search to files or directories

**“.”** :: search in current working directory

**“/”** :: search in root directory

Examples::

**find/root -name file1** :: it prints all the paths of file1 in /root

**find dir2 -name file2** :: it prints all the paths of file1 in dir2

**find. -name file3** :: it prints all the paths of file3 in current working directory

**find dir3 -type f** :: it prints all the paths of a files in dir3

**find / -type d** :: it prints all the directories path in / directory **find / root -type l** :: all the paths of a link files in / root directory **find / root -inum** :: using inode number we can get path of a file

##### 04-07-2022

**File System::** Storage Space: 500 GB Drive

It tells us how to store the data in drive

Without file system we can’t use the storage space

Supported File System

1. Windows [NTFS and FAT]
   1. NTFS :: New Techonology File System
   2. FAT ::
2. Linux [EXIT] [ext2, ext3, ext4]
   1. Ext :: extended File System
3. Vmware [ESXI]
   1. VMFS [VM File system] Format a storage space with File System **Windows::**
4. Go to Disk Management
5. Right click on Unallocated space and select New simple volume
6. Next
7. Select Size
8. Select File System [NTFS and FAT]
9. Finish

###### Linux ::

**mkfs** :: make file system syntax ::

###### mkfs.<ext> <drive\_name>

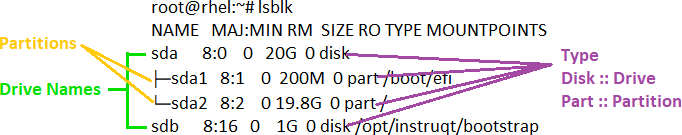
Example::

mkfs.ext4 /dev/sda

###### Partition Management::

**Partition** :: Dividing storage space into multiple small units

\*\*\* To List Drives in Linux :: **lsblk**



**Partition Management**

Partition is a part of drive space treated as individual / independent disk

**Partition** :: Dividing storage space into multiple small units

###### Different types of Partitions

1. Primary Partition
   1. We can create up to 4 partitions
2. Extended Partition
   1. We can create more than 4

###### Creating Partition in Windows

1. Go to disk Management
2. Right Click on Unallocated area / disk space
3. Next
4. Select Partition size
5. Select Drive Letter
6. Finish

###### Linux::

Syntax ::

###### fdisk <drive\_name>

Example::

###### fdisk /dev /sdb

Command (m for help): n

Select Partition Type [Primary and Extended] :: p Select partition number :: 1

First Sector :: 2048

Last Sector :: 1M [K,M,G,T] Command (m for help): w

**Note ::** if we want to leave it default we just need to press enter

###### To list the Partitions::

syntax ::

fdisk -l <drive\_name> Example::

fdisk -l /dev/sdb

To delete Partition in linux

1. fdisk <drive\_name>
2. Command (m help): d
3. Partition number (1,2 default 2): 1 [Select the partition that we need to delete]

##### 07-07-2022

**Process Management**

Whenever we launch any application / command there will be a process created for that particular application and running in the background [Task Manager in Windows]

Process is identified by a process ID [PID]

\*\*\*To list all Process **“ps”**

\*\*\*To list all the running processs :: **ps -eaf To kill the process::**

Syntax

###### kill -9 <Process\_id>

kill -9 PID Example::

###### kill -9 1866

To kill the process with process name syntax::

###### pkill <process\_name>

**Generic ls commands**

**lsblk ::** To list all the block devices [Drives]

**lsusb ::** To list all the usb device

**lscpu ::** To list all the CPU’s

**lsscsi ::** To list all the SCSI Devices

**lspci ::** To list all the PCI Devices

**lshw ::** To list all the hardware devices

**Questions**

## Difference between 32 bit and 64 bit Operating system?

* + A 32-bit system can access 232 memory addresses, i.e., 4 GB of RAM or physical memory; ideally, it can also access more than 4 GB of RAM.
  + A 64-bit system can access 264 memory addresses, i.e., actually 18-Quintillion bytes of RAM. In short, any amount of memory greater than 4 GB can be easily handled by it.

A 64-bit processor is more capable than a 32-bit processor because it can handle more data at once. A 64-bit processor can store more computational values, including memory addresses, which means it can access over 4 billion times the physical memory of a 32-bit processor.

|  |  |  |
| --- | --- | --- |
| **Parameters** | **32-bit Processors** | **64-bit Processors** |
| Handling of  Data and Storage | As its name suggests, the 32 bit OS can store and handle lesser data than the 64 bit OS. More specifically, it addresses a maximum of 4,294,967,296 bytes (4 GB) of RAM. | The 64 bit OS, on the other hand, can handle more data than the 32 bit OS. It means that it can address a total of 264 memory addresses, which is 18- Quintillion GB of RAM. |
| Architecture | The 32-bit system has general computing, including IBM System/360 and IBM System/370, the DEC VAX, the Motorola 68000 Family, the Intel IA-32, and the 32-bit version of x86 architecture different versions. These are architectures that are used for embedded computing and include 68000 families. | The registers are divided into different groups like integer, floating, control and often for addresses of various uses and names like address, index or base registers. The size of these registers is dependent on the amount of addressable memory. |
| Compatibility of System | A 32-bit processor system could properly run a 32-bit OS, but it cannot run the 64-bit OS at its full capability. | A 64-bit processor system can run either a 32-bit or 64-bit version of an installed operating system (OS). |
| Performance | The factor of performance in a 32-bit processor is less efficient than the 64-bit processor. | It exhibits a higher performance than the 32-bit processor. |
| Application Support | The 64-bit programs and applications won't work. | The 32-bit programs and applications will work with no hassle. |
| Addressable Space | It has an addressable space of 4 GB. | These have an addressable space of 16 GB. |
| Calculation per second | 32-bit systems have dual-core and quad-core versions available. | 64bit systems can come with dual- core, quad-core, six-core, and eight- core versions. Having these multiple cores available has increased its speed of calculations per second. |

|  |  |  |
| --- | --- | --- |
| Multitasking Support | The 32-bit system is not an ideal option for multitasking and stress-testing. | For multitasking and stress testing, the 64-bit processor is better. It also works well for the execution of other heavy applications. |
| OS Support | It needs a 32-bit operating system. | This one can run on both 32-bit and the 64-bit operating system. |
| OS and CPU Requirements | The 32-bit applications and operating systems require 32-bit CPUs. | The 64-bit operating system needs a 64-bit CPU, and the 64-bit applications require a 64-bit CPU and OS. |
| Systems Available | These support Windows 7, Windows XP, Windows Vista, Windows 8, and Linux. | These support Windows XP Professional, Windows 7, Windows 8, Windows 10, Windows Vista, Linux, and Mac OS X. |
| Limits in  Memory | A 32-bit system has a limit of 32 bit Windows  3.2 GB of RAM. The limit in its addressable space doesn't allow you to use the entire physical memory space of 4GB. | A 64-bit system enables its users to s |

## Difference between desktop OS and Server OS?

Both of these are types of OS, but there is a fundamental difference between Server OS and Client OS. In this article, we will discuss the same in a tabular form. But before we do that, let us know a bit more about an OS. The term OS is an abbreviation for the operating system. The OS is basically system software. It performs the management of software resources and computer hardware. It also provides some common services for various computer programs. Now let us take a look at the difference between client and server.

What is a Client OS?

A client OS is a type of OS that operates with the desktop (available device). We use it to obtain various services from a server, and it runs on various client devices, such as computers, laptops, etc., and it is a fairly simple type of operating system.

What is a Server OS?

A server OS is a type of OS that is designed in a way that we can use it on an available server. We use a server OS to provide a variety of services to many of the clients. It is capable of serving multiple numbers of clients at any given time, and it is a comparatively much more advanced type of operating system.

Difference Between Server OS and Client OS

Let us talk about the differences present between Server OS and Client OS.

|  |  |  |
| --- | --- | --- |
| **Parameters** | **Server OS** | **Client OS** |
| Basics | We use a Server OS for providing various services to multiple numbers of clients. | We use a Client OS for obtaining various services from any given server. |

|  |  |  |
| --- | --- | --- |
| Number of Users/ Clients | A Server OS is capable of serving multiple clients at any given time. | The Client OS is capable of serving just a single client at any given time. |
| Complexity | It is a complex type of OS. | It is a fairly simple type of OS. |
| Medium | A server OS basically runs on a given server. | A client OS basically runs on various client devices, such as computers, laptops, etc. |
| Operations | This type of OS is designed in a way that it operates on any server. | This type of OS is designed in a way that it operates within a desktop. |
| Security | It is comparatively more secure. | It is comparatively much less secure. |
| Processing Power | The processing power is fairly high in Server OS. | The processing power is much lower in Client OS. |
| Efficiency | The Server OS is highly efficient. | The Client OS is comparatively less efficient. |
| Stability | Server OS is much more stable. | Client OS is much less stable. |
| Examples | Linux, Red Hat, etc., are a few examples of the Server OS. | Android, Windows, etc., are a few examples of the Client OS. |

## What is IP address?

* + An IP address is a unique address that identifies a device on the internet or a local network. IP stands for "Internet Protocol," which is the set of rules governing the format of data sent via the internet or local network.
  + In essence, IP addresses are the identifier that allows information to be sent between devices on a network: they contain location information and make devices accessible for communication. The internet needs a way to differentiate between different computers, routers, and websites. IP addresses provide a way of doing so and form an essential part of how the internet works.

An IP address is a string of numbers separated by periods. IP addresses are expressed as a set of four numbers — an example address might be 192.158.1.38. Each number in the set can range from 0 to 255. So, the full IP addressing range goes from 0.0.0.0 to 255.255.255.255.

IP addresses are not random. They are mathematically produced and allocated by the [Internet Assigned](https://www.iana.org/) [Numbers Authority](https://www.iana.org/) (IANA), a division of the [Internet Corporation for Assigned Names and Numbers](https://www.icann.org/) (ICANN). ICANN is a non-profit organization that was established in the United States in 1998 to help maintain the security of the internet and allow it to be usable by all. Each time anyone registers a domain on the internet, they go through a domain name registrar, who pays a small fee to ICANN to register the domain.

Watch this video to learn what IP address is, why IP address is important and how to protect it from hackers

## What are different types of IP address

**What is an IP Address?**

An IP address is a numerical label assigned to the devices connected to a computer network that uses the IP for communication. IP address act as an identifier for a specific machine on a particular network. It also helps you to develop a virtual connection between a destination and a source.

**IP Full Form**: The IP address stands for Internet Protocol address is also called IP number or internet address. It helps you to specify the technical format of the addressing and packets scheme. Most networks combine TCP with IP

**Types of IP address**

There are mainly four types of IP addresses:

* + Public,
  + Private,
  + Static
  + Dynamic.

Among them, public and private addresses are based on their location of the network private, which should be used inside a network while the public IP is used outside of a network.

Let us see all these types of IP address in detail.

**Public IP Addresses**

A public IP address is an address where one primary address is associated with your whole network. In this type of IP address, each of the connected devices has the same IP address.

This type of public IP address is provided to your router by your ISP.

**Private IP Addresses**

A private IP address is a unique IP number assigned to every device that connects to your home internet network, which includes devices like computers, tablets, smartphones, which is used in your household.

It also likely includes all types of Bluetooth devices you use, like printers or printers, smart devices like TV, etc. With a rising industry of internet of things (IoT) products, the number of private IP addresses you are likely to have in your own home is growing.

**Dynamic IP address:**

Dynamic IP addresses always keep changing. It is temporary and are allocated to a device every time it connects to the web. Dynamic IPs can trace their origin to a collection of IP addresses that are shared across many computers.

Dynamic IP addresses are another important type of internet protocol addresses. It is active for a specific amount of time; after that, it will expire.

**Static IP Addresses**

A static IP address is an IP address that cannot be changed. In contrast, a dynamic IP address will be assigned by a Dynamic Host Configuration Protocol (DHCP) server, which is subject to change. Static IP address never changes, but it can be altered as part of routine network administration.

Static IP addresses are consistent, which is assigned once, that stays the same over the years. This type of IP also helps you procure a lot of information about a device

## What is gateway and default gateway?

**The default gateway is used as the destination of all traffic that is not on the same subnet**. The gateway is a layer 3 device such as a router or multi-layer switch that is used to route traffic on a hop-by-hop basis

## What is DNS?

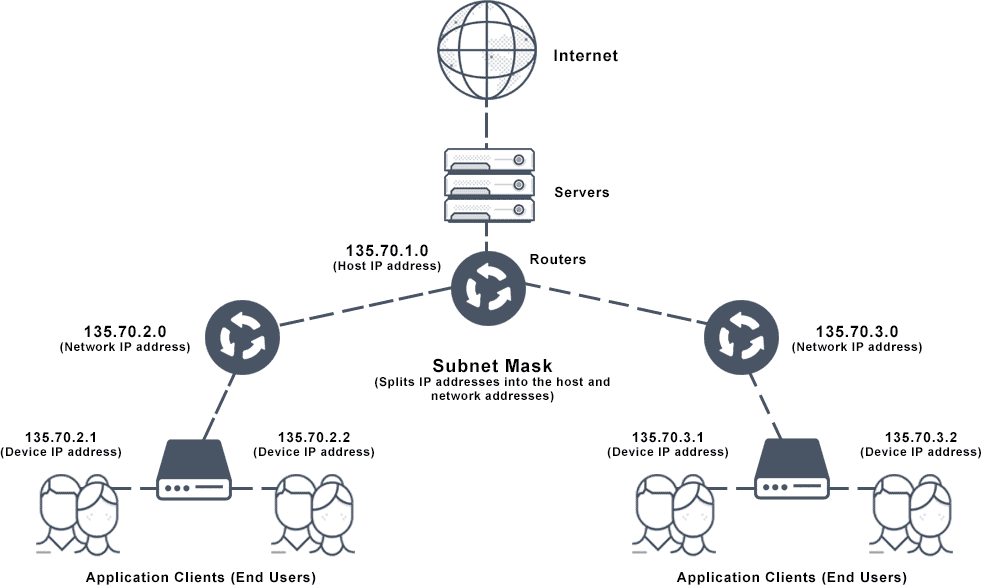
The domain name system (DNS) is **a naming database in which internet domain names are located and translated into Internet Protocol (IP) addresses**. The domain name system maps the name people use to locate a website to the IP address that a computer uses to locate that website.

## What is DHCP?

Dynamic Host Configuration Protocol (DHCP) is **a network protocol that is used to configure network devices to communicate on an IP network**. A DHCP client uses the DHCP protocol to acquire configuration information, such as an IP address, a default route, and one or more DNS server addresses from a DHCP server.

## What is subnet mask?

A subnet mask is **a 32-bit number created by setting host bits to all 0s and setting network bits to all 1s**. In this way, the subnet mask separates the IP address into the network and host addresses. The “255” address is always assigned to a broadcast address, and the “0” address is always assigned to a network address



## Difference between IPV4 and IPV6 with examples of IP address?

We can also define an IP address as **a numeric address assigned to each device on a network**. An IP address is assigned to each device so that the device on a network can be identified uniquely.

Differences between IPv4 and IPv6.

|  |  |  |
| --- | --- | --- |
|  | **Ipv4** | **Ipv6** |
| Address length | IPv4 is a 32-bit address. | IPv6 is a 128-bit address. |

## What is MAC address, difference between MAC and IP?

The major difference between MAC address and IP address is that IP addresses are applied to uniquely find the connection of the network with that connecting device. While the MAC Address is used to provide the physical address of the device/computer. This address uniquely identifies the devices on a network.

## Mounting ::

To mount storage space / partition

1.Create mount directory under /mnt

“**mount**”

syntax::

###### mount <partition> <mount\_direcory>

Example::

###### mount /dev/sda1 /mnt/dir1

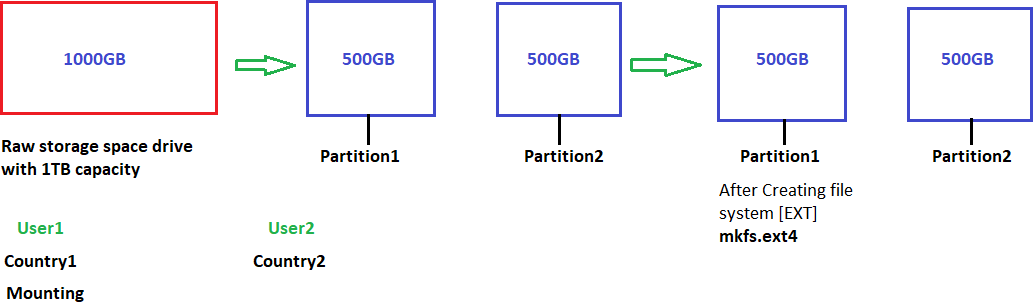
Above one is an example for local mounting

To mount it to a remote user syntax::

###### mount -o <patition> -t nfs username@ipaddress:<mount\_directory>

Example::

mount -o /dev/sda1 -t nfs root@10.102.1.1:/mnt/dir1



##### \*\*\* Package Management

There are different ways of installing packages in linux

1. RPM [Redhat Package Manager]
2. YUM [Yellow dog updater, Modifier]
3. apt-get

###### \*\*\* RPM::

RPM Stands for Redhat Package Manager Rpm files having an extension **.rpm** Example::

**python3.orge.12912.rpm** To install rpm package **rpm -i <package\_name>**

Example::

rpm -i python3.orge.12912.rpm

To list rpm package

###### rpm -qa <rpm\_package\_name>

Example::

rpm -qa python3.orge.12912.rpm

To erase / remove rpm file syntax::

###### rpm -e <rpm\_package\_name>

Example::

rpm -e python3.orge.12912.rpm

###### \*\*\* YUM ::

YUM Stands for yellow dog updater, modifier To install package

syntax::

###### yum install <package\_name>

Example::

yum install python

To uninstall package syntax::

###### yum remove <package\_name>

Example::

yum remove fio

To Upgrade syntax::

###### yum upgrade <package\_name>

Example::

yum upgrade python

To list all the installed packages

**yum list**

**Linux questions**

**1.What are diffrent stages in Linux Booting process**

**BIOS or UEFI Firmware ::**

The process begins with the computer's Basic Input/Output System (BIOS) or Unified Extensible Firmware Interface (UEFI). This firmware initializes hardware components, performs a Power-On Self-Test (POST), and searches for a boot device.

**Boot Loader ::**

After the firmware initializes the hardware, it looks for a boot loader on the boot device (usually a hard drive or an SSD). Popular boot loaders for Linux include GRUB (GRand Unified Bootloader) and LILO (LInux LOader). The boot loader loads the Linux kernel into memory and passes control to it.

**Kernel Initialization ::**

The Linux kernel is loaded into memory by the boot loader. It then initializes essential hardware components, such as the CPU, memory, and devices required for booting.

**Init Process ::**

Once the kernel is initialized, it starts the init process. Traditionally, this process is managed by the SysVinit system, but many modern Linux distributions have shifted to systemd. The init process is responsible for starting system services and processes defined in the system's runlevel or systemd targets.

**User Space Initialization ::**

After the init process has completed its tasks, it starts other system services and daemons required for the user space environment. These services may include network management, logging, graphical user interface (if applicable), and others.

**Login Prompt or Graphical Display Manager ::**

Once the user space is initialized, the system presents a login prompt in text mode or launches a graphical display manager (such as GDM, LightDM, or SDDM) if configured to start a graphical session.

**User Login ::**

Finally, users can log in to the system either through the text-mode login prompt or the graphical display manager. After successful authentication, users gain access to the system and can start using it.

**2.what are diffrent run levels in linux**

**Runlevel 0**: Halt or shutdown mode. The system is brought down to a state where it is safe to power off.

**Runlevel 1:** Single-user mode. Also known as maintenance mode, this runlevel is used for system maintenance tasks. Typically, only a minimal set of services are started, and the system boots to a command-line interface as the root user.

**Runlevel 2:** Multi-user mode with networking but without graphical interface. In this runlevel, the system boots to a command-line interface, and networking services are available.

**Runlevel 3:** Multi-user mode with networking and text-based login. Similar to runlevel 2, but may include additional services.

**Runlevel 4:** Unused. Historically, this runlevel was left unused in many distributions and could be customized by the user.

**Runlevel 6:** Reboot mode. The system is rebooted.

**note :: It's important to note that the exact behavior of runlevels or targets can vary between different Linux distributions**

3.what is the path to set run level

**/etc/initrd**

4.what is the path of GRUB configuration path ?

5.command to create 5 files in same directory

**touch f1 f2 f3 f4 f5**

6.command to list all hidden files ?

**ls -a**

7.command list directories?

**ls -ld**

8.command to list files in human readable format >

**ls -h**

9.command to print files with sizes >

**ls -s**

10.what is the use of "cd ~" , "cd /" , "cd -" ?

**cd - :: one step backword**

**cd ~ :: to go root directory**

**cd / :: to go back dir**

11.command to list content in a file

**cat filename**

12.command to overwrite & replace a file content

**cat>>filename**

13.command to print first 10 lines in a file

**head -10 filename**

14.command to print last 10 lines in a file

**tail -10 filename**

15.command to copy files from one dir to another dir

**cp file\* dir1 dir2**

16.command to copy directory to another directory

**cp -r dir1 dir2**

17.how to remove files forcefully in linux?

**rm -f files\***

18.how to rename a file ? file1 to myfile.txt ?

**mv file1 myfile.txt**

19.how to find IP address of linux machine ?

**ifconfig**

20.how to change file permissions in Linux ?

**by useing chmod**

21.Diffrent modes in file permissions ?

**read 4, write 2, execute 1**

**by useing chmod**

22.how to change hostname ?

**hostname ::** to known the current hostname

**sudo hostname new\_hostname ::** to change hostname temporarily

**sudo nano /etc/hostname ::** to change hostname permanently

23.how to add user?

**useradd**

24.how to change password ?

**passwd**

25.how to find current date ?

**date**

26.command to print system information

**lshw**

27.command to print count no.of occurences of given string in linux using grep

**grep -o "search\_string" file\_name | wc -l**

**wc ==** word count

The -o option tells grep to only print the matching parts of the line, rather than the entire line

28.command to search pattern inside all directories ?

**find /path/to/search -type f -exec grep -H 'pattern' {} +**

**find /path/to/search**: This command starts a search in the specified directory (/path/to/search) and its subdirectories.

**-type f:** This option tells find to only consider regular files (not directories or other types of files).

**-exec grep -H 'pattern' {} +:** For each file found by find, this command executes grep to search for the specified pattern.

**-H** option tells grep to print the file name along with the matched line.

**'pattern'** is the pattern you want to search for.

**{}** is a placeholder that find replaces with the name of the file being processed.

**+** tells find to pass as many file names as possible to each invocation of grep, which is more efficient than invoking grep once for each file.

29.how to find test.py file in /etc

**find /etc -name test.py if we give -iname then find incase sensitive**

30.command to print all real time information about performace of system

**top, htop, iotop, nmon, atop this all commands can show the information but in different format**

31.how to get all running process in linux

**ps**

32.how to kill process forcefully?

**kill -9 PID**

33.how to check path of any package

**which python**

34.command to create and extract a zip file

**zip archive\_name.zip file1 file2 directory1**

**unzip archive\_name.zip**

35.command to create and extract archive file

**tar -cvf archive\_name.tar file1 file2 directory1**

**tar -xvf archive\_name.tar**

36.command to get IPaddress if we know hostname

**host hostname**

37.how to change IP address ?

**sudo ifconfig eth0 <new\_ip\_address> netmask <netmask>**

**For example:**

**sudo ifconfig eth0 192.168.1.100 netmask 255.255.255.0**

38.command to change user from root to user

**sudo -u username -i**

39.command to list all hardware information ?

**lshw**

40.command to list all modules?

**lsmod**

41.what are the different ways of installing packages with syntax ?

**Advanced Package Tool (APT) - Debian/Ubuntu:**

**sudo apt install package\_name**

**Example:**

**sudo apt install firefox**

**Yellowdog Updater, Modified (YUM) - Red Hat-based (CentOS, Fedora):**

**sudo yum install package\_name**

**Example:**

**sudo yum install httpd**

42.how to start,stop, restart a service in linux

**To start a service:**

**sudo service service\_name start**

**or**

**sudo systemctl start service\_name**

**To stop a service:**

**sudo service service\_name stop**

**or**

**sudo systemctl stop service\_name**

**To restart a service:**

**sudo service service\_name restart**

**or**

**sudo systemctl restart service\_name**

43.command to list all services in linux

**systemctl list-units --type=service --all**

**44.what are diffrent root directories in linux**

**/root**

**/lib**

**/bin**

**/dev**

**/etc**

45**.what is the command used to create partitions in linux**

**fdisk /dev/{'drive\_name'}**

46.what is filesystem ?

**the linux os which can formated with .ext filesystem**

**by creating this filesystem on drive only we can boot the os or we can store data**

**the term "filesystem" refers to the organization and management of files and directories on a storage device such as a hard disk drive, solid-state drive**

**47.what is the command used to create filesystem**

**mkfs.ext4**

**mkfs.ext4 /dev/sda1**

**48.commands to get following info**

**dmidecode tool is used to see informataion of hardware components**

**49.CPU info**

**dmidecode -t proccesor**

**cpuinfo**

**50.hardware info**

**lshw to list hardware components**

**dmidecode**

**51.block devices**

**lsblk**

**52.SCSI devices**

**lsscsi**

**53.RAM info**

**dmidecode -t memory**

**meminfo**

**54.Disk space utilization**

**du -h**

**df -h**

**55.what are all the info /proc contains**

**process information**

**cmdline: Command-line arguments used to start the process.**

**cwd: Current working directory of the process.**

**exe: Symbolic link to the executable file of the process.**

**status: Various status information about the process, including memory usage, process state, etc.**

**System information**

**cpuinfo: Information about the CPU(s) in the system, such as model name, number of cores, etc.**

**meminfo: Information about system memory, including total memory, free memory, etc.**

**uptime: System uptime and idle time.**

**version: Kernel version and build information.**

**interrupts: Information about interrupts handled by the kernel.**

**loadavg: System load average.**

**56.command used to mount a volume with syntax**

**sudo mount /dev/sda1 /mnt/data**

**57.command to get all empty files and delete**

**find . -type f -empty -delete**

**58.what is swap file and location**

**A swap file is a special type of file used by operating systems, including Linux, as a form of virtual memory.When a system's physical RAM (Random Access Memory) is fully utilized, and additional memory is needed, the operating system can use a swap file to temporarily store data that would typically be stored in RAM. This allows the system to continue running smoothly even when physical memory is exhausted.**

**Swap files are typically located on the filesystem, just like regular files, but they serve a different purpose. The location of the swap file can vary depending on the distribution and configuration of the Linux system.**

**59.syntax to copy files & directories over 2 machines**

**scp -r /local/path/to/files user@remote\_host:/remote/path**

**60.syntax to connect remote machine**

**import paramiko**

**client=paramiko.SSHClient**

**client.set\_missing\_host\_key\_policy(paramiko.AutoAddPolicy)**

**client.connect(ip\_adress,username=username,password=password)**

**\_stdin,\_stdout,\_stderr=client.exec\_command(" ")**

**print(\_stdout.read().decode())**

**61.what is inode**

**after creating every file os can create a unique inode number for that file**

**os can detect a file by useig the inode number**

**62.how to find inode number**

**ls -i**

**63.diffrence b/w GPT and MBR partitions ?**

**64.command to get linux version > cat /os**

**uname -a**

**65.path to see system logs in linux**

**cat /var/log/syslog**

**66.command to create hardlink and softlink**

**ln -s file1 file2 :: softlink**

**ln file1 file2 :: hardlink**

**67.filesystem used in linux ?**

**.ext filesystem**

**68.file system used in windows ?**

**NFTS filesystem**

**69.command to get present working directoy**

**pwd**

**70.syntax to give full permission to file.txt**

**chmod 777 file.txt**

**71.command to check CPU utilization**

**top**

**72.diffrence b/w hardlink and softlink**

**hardlink :: to used to retrive the data[used for backup file]**

**softlink :: used to ceate the shortcut file**

**hl == ln file1 file2**

**sl == ln -s file1 file2**

**ls -i :: to list inode numbe of all files anaad dirs**

**ln file file1 :: to create**

**73.command to list all files ending with .py and starting with python ?**

**ls \*.py**

**ls \*python**

**74.command to copy .py files from present working directory to /etc/var/log?**

**cp \*.py /etc/var/log**

**75.What is the maximum length for any file name under LINUX? 255**

**255 characters in length.**

**76.comamnd to save output of command in file**

**ls -l > file\_list.txt**

**77.use of netstat**

**The netstat command is a network utility tool**

**netstat -tulpn :: Shows all TCP and UDP connections.**

**netstat -at :: Displays all active TCP or UDP connections.**

**netstat -lnt :: Displays all listening TCP or UDP ports.**

**netstat -r :: Prints the kernel routing table.**

**netstat -i :: Shows statistics for all network interfaces.**

**netstat -s :: Prints various network statistics.**

**netstat -l :: Shows all listening sockets.**

**netstat -g :: Shows multicast group memberships for IPv4 and IPv6.**

**netstat -M :: Shows masqueraded connections for Network Address.**

**78.command to replace a string in file from command prompt**

**sed -i 's/old\_string/new\_string/g' filename**

**79.how to create,delete and rename directory**

**mkdir :: to create a dir**

**rm -r | rmdir :: to delete a dir**

**mv dir1 dir2 :: dir1 is original dir and dir2 rename dir**

**80. diffrence b/w du and df**

**du[disk usage] :: du -h**

**df[disk free] :: df -h**

**81. How do you get the full path of a file in Linux?**

**pwd**

**82. How do you find who is logged in?**

**who -a**

**83. Port numbers of SSH,FTP,HTTP,Iscsi**

**SSH (Secure Shell):**

**Port Number: 22**

**FTP (File Transfer Protocol):**

**Port Number: 21**

**HTTP (Hypertext Transfer Protocol):**

**Port Number: 80**

**iSCSI (Internet Small Computer System Interface):**

**Port Number: 3260**

**84. what is the first process started by kernal and what is its process ID**

**the first process while started kernal is [init]**

**cmd :: ps -p 1**

**This command will display information about the process with PID 1, which should be either "init" (in SysVinit systems) or "systemd" (in systemd-based systems).**

**85. command used to get no.of characters in a file**

**wc[word count]**

**wc -m<filename> :: m=number of charactaers**

**86. What is the minimum number of partitions that you need to install Linux?**

**one partition is enough to hold entire linux file system**

**more information ::**

**Root Partition (/) :: This partition contains the core Linux filesystem and system files.**

**Swap Partition :: This partition serves as virtual memory space and can be used to support hibernation or provide additional memory when physical RAM is exhausted.**

**/home Partition : This partition stores user data and preferences, providing a separate space for personal files and settings. Separating /home from the root partition can simplify backups and system upgrades.**

**87. use of curl command**

**The curl command is a powerful tool used for transferring data to or from a server. It supports various protocols such as HTTP, HTTPS, FTP, FTPS, SCP, SFTP, TFTP, LDAP, and more.**

**Downloading Files: You can use curl to download files from a remote server.**

**For example :: curl -O https://example.com/file.txt**

**This command will download the file file.txt from the specified URL and save it with the same name in the current directory.**

**HTTP Requests: curl allows you to make HTTP requests to web servers. You can specify various HTTP methods such as GET, POST, PUT, DELETE, etc.**

**For example :: curl -X GET https://api.example.com/data**

**This command sends a GET request to retrieve data from the specified API endpoint.**

**Sending Data: You can use curl to send data to a server, such as form data or JSON payloads.**

**For example :: curl -X POST -d "username=admin&password=123456" https://example.com/login**

**This command sends a POST request with form data to the specified URL for user authentication.**

**Uploading Files: curl can also be used to upload files to a server.**

**For example::curl -F "file=@localfile.txt" https://example.com/upload**

**This command uploads the local file localfile.txt to the specified server using a multipart/form-data POST request.**

**Follow Redirects: By default, curl follows redirects automatically. You can use the -L option to instruct curl to follow redirects.**

**For example :: curl -L https://example.com**

**This command retrieves the content of the URL after following any redirects.**

**Sending Headers: You can include custom HTTP headers in your requests using the -H option.**

**For example :: curl -H "Authorization: Bearer token123" https://api.example.com/data**

**This command sends an HTTP request to the specified endpoint with an Authorization header containing a bearer token.**

**Downloading Multiple Files :: You can use curl in conjunction with the -O option to download multiple files from a list of URLs stored in a file.**

**For example :: curl -O -L https://example.com/files.txt**

**This command downloads multiple files listed in the files.txt file.**

**...........................................................**

**88. command to check system wakeup time**

**uptime**

**89. What are all the default directories/partitions gets created during linux installation**

**/bin, /boot, /lib, /etc, /dev, /media**

**/bin :: it contains binary commands which are required for booting process**

**/boot :: it contaians required system booting process**

**/lib :: it contains libraries which required to execute /bin..**

**90. what is the use of SSH , Winscp and RDP ::**

**SSH is primarily used for secure remote access and administration of Unix-based systems**

**WinSCP is used for secure file transfer between Windows and Unix-based systems**

**RDP is used for remote desktop access and control of Windows-based systems. Each tool serves its own specific purpose in remote access and file**