UNIT I

CLOUD ARCHITECTURE MODELS AND INFRASTRUCTURE

Cloud Architecture: System Models for Distributed and Cloud Computing – NIST Cloud Computing Reference Architecture – Cloud deployment models – Cloud service models.

CLOUD ARCHITETURE:

1. Explain in detail about Cloud Computing Architecture (Or) Explain the Various Layered Cloud Architectural Development Design for Effective Cloud Computing Environment. (Nov/Dec 2020)

Cloud Architecture Design

The cloud architecture design is the important aspect while designing a cloud. The simplicity in cloud services attract cloud users to use it which makes positive business impact.

Therefore, to design such a simple and user - friendly services, the cloud architecture design plays an important role to develop that.

Every cloud platform is intended to provide four essential design goals like scalability, reliability, efficiency and virtualization.

To achieve this goal, certain requirements have to be considered. The basic requirements for cloud architecture design are given as follows:

- The cloud architecture design must provide automated delivery of cloud services along with automated management.
- It must support latest web standards like Web 2.0 or higher and REST or RESTful APIs.
- It must support very large - scale HPC infrastructure with both physical and virtual machines.
- The architecture of cloud must be loosely coupled.
- It should provide easy access to cloud services through a self - service web portal.
- Cloud management software must be efficient to receive the user request, finds the correct resources and then calls the provisioning services which invoke the resources in the cloud.
- It must provide enhanced security for shared access to the resources from data centers.
- It must use cluster architecture for getting the system scalability.
- The cloud architecture design must be reliable and flexible.
- It must provide efficient performance and faster speed of access.

Today's clouds are built to support lots of tenants (cloud devices) over the resource pools and large data volumes. So, the hardware and software plays an important role to achieve that.

The rapid development in multicore CPUs, memory chips, and disk arrays in the hardware field has made it possible to create data centers with large volumes of storage space instantly. While development in software standards like web 2.0 and SOA have immensely helped to developed a cloud services.

The Service Oriented Architecture (SOA) is also a crucial component which is used in the delivery of SaaS.

The web service software detects the status of the joining and leaving of each node server and performs appropriate tasks accordingly. The virtualization of infrastructure allows for quick cloud delivery and recovery from disasters. In recent cloud platforms, resources are built into the data centers which are typically owned and operated by a third - party provider.

Layered Cloud Architecture Design

The layered architecture of a cloud is composed of three basic layers called infrastructure, platform and application. These three levels of architecture are implemented with virtualization and standardization of cloud - provided hardware and software resources. This architectural design facilitates public, private and hybrid cloud services that are conveyed to users through networking support over the internet and the intranets. The layered cloud architecture design is shown in Fig. 1.1

In layered architecture, the foundation layer is infrastructure which is responsible for providing different Infrastructure as a Service (IaaS) components and related services.

It is the first layer to be deployed before platform and application to get IaaS services and to run other two layers.

- The infrastructure layer consists of virtualized services for computing, storage and networking. It is responsible for provisioning infrastructure components like compute (CPU and memory), storage, network and IO resources to run virtual machines or virtual servers along with virtual storages.
- The abstraction of these hardware resources is intended to provide the flexibility to the users. Internally, virtualization performs automated resource provisioning and optimizes the process of managing resources.

The infrastructure layer act as a foundation for building the second layer called platform layer for supporting PaaS services.

The platform layer is responsible for providing readily available development and deployment platform for web applications to the cloud users without needing them to install in a local device. This layer provides an environment for users to create their applications, test operation flows, track the performance and monitor execution results.

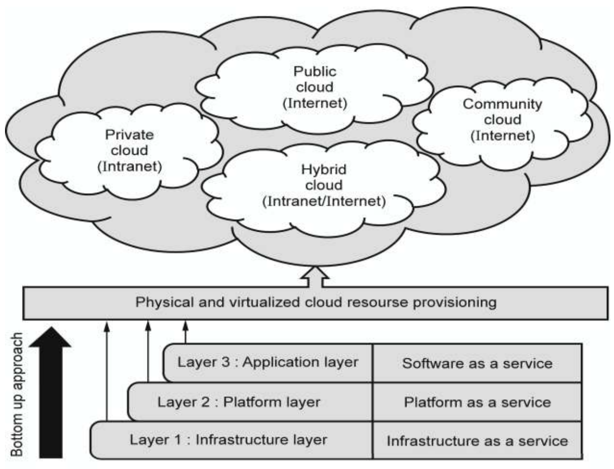

Public cloud (Internet)
Community cloud (Internet)
Private cloud (Intranet)
Hybrid cloud (Intranet/Internet)
Physical and virtualized cloud resourse provisioning
Bottom up approach
Layer 3 : Application layer
Software as a service
Layer 2 : Platform layer
Platform as a service
Layer 1 : Infrastructure layer
Infrastructure as a service

Fig. 1.1 Layered cloud architecture design

The platform must be ensuring to provide scalability, reliability and security. In this layer, virtualized cloud platform, acts as an "application middleware" between the cloud infrastructure and application layer of cloud. The platform layer is the foundation for application layer.

A collection of all software modules required for SaaS applications forms the application layer. This layer is mainly responsible for making on demand application delivery.

In this layer, software applications include day-to-day office management software's used for information collection, document processing, calendar and authentication.

Enterprises also use the application layer extensively in business marketing, sales, Customer Relationship Management (CRM), financial transactions and Supply Chain Management (SCM). It is important to remember that not all cloud services are limited to a single layer.

Many applications can require mixed - layers resources. After all, with a relation of dependency, the three layers are constructed from the bottom-up approach. From the perspective of the user, the services at various levels need specific amounts of vendor support and resource management for functionality.

In general, SaaS needs the provider to do much more work, PaaS is in the middle and IaaS requests the least. The best example of application layer is the Salesforce.com's CRM service where not only the hardware at the bottom layer and the software at the top layer is supplied by the vendor, but also the platform and software tools for user application development and monitoring.

System Models for Distributed and Cloud Computing

2. Explain in detail about system models for distributed Cloud Computing.

Distributed and cloud computing systems are built over a large number of autonomous computer nodes. These node machines are interconnected by SANs, LANs, or WANs in a hierarchical manner.

With today's networking technology, a few LAN switches can easily connect hundreds of machines as a working cluster. A WAN can connect many local clusters to form a very large cluster of clusters. Massive systems are considered highly scalable, and can reach web-scale connectivity, either physically or logically.

Massive systems are classified into four groups:

1. Clusters of Cooperative Computers

A computing cluster consists of interconnected stand-alone computers which work cooperatively as a single integrated computing resource. In the past, clustered computer systems have demonstrated impressive results in handling heavy workloads with large data sets.

1.1 Cluster Architecture

In Figure 1.15 shows the architecture of a typical server cluster built around a low-latency, high-bandwidth interconnection network. This network can be as simple as a SAN (e.g., Myrinet) or a LAN (e.g., Ethernet). To build a larger cluster with more nodes, the interconnection network can be built with multiple levels of Gigabit Ethernet, Myrinet, or InfiniBand switches. Through hierarchical construction using a SAN, LAN, or WAN, one can build scalable clusters with an increasing number of nodes. The cluster is connected to the Internet via a virtual private network (VPN) gateway.

All resources of a server node are managed by their own OS. Thus, most clusters have multiple system images as a result of having many autonomous nodes under different OS control.

1.2 Single-System Image :

Cluster designers desire a cluster operating system or some middle-ware to support SSI at various levels, including the sharing of CPUs, memory, and I/O across all cluster nodes. An SSI is an illusion created by software or hardware that presents a collection of resources as one integrated, powerful resource. SSI makes the cluster appear like a single machine to the user.

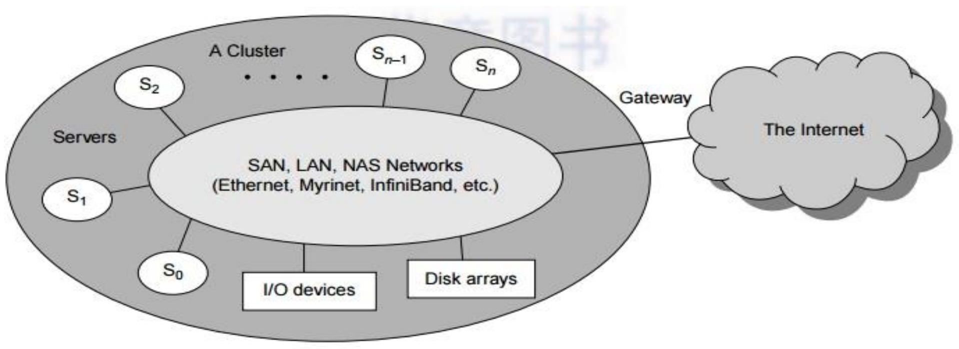


FIGURE 1.15
A cluster of servers interconnected by a high-bandwidth SAN or LAN with shared I/O devices and disk arrays; the cluster acts as a single computer attached to the Internet.

A cluster with multiple system images is nothing but a collection of independent computers.

1.3 Hardware, Software, and Middleware Support

Clusters exploring massive parallelism are commonly known as MPPs. Almost all HPC clusters in the Top 500 list are also MPPs. The building blocks are computer nodes (PCs, workstations, servers, or SMP), special communication software such as PVM or MPI, and a network interface card in each computer node. Most clusters run under the Linux OS. The computer nodes are interconnected by a high-bandwidth network (such as Gigabit Ethernet, Myrinet, InfiniBand, etc.

Special cluster middleware supports are needed to create SSI or high availability (HA). Both sequential and parallel applications can run on the cluster, and special parallel environments are needed to facilitate use of the cluster resources. For example, distributed memory has multiple images. Users may want all distributed memory to be shared by all servers by forming distributed shared memory (DSM). Many SSI features are expensive or difficult to achieve at various cluster operational levels. Instead of achieving SSI, many clusters are loosely coupled machines. Using virtualization, one can build many virtual clusters dynamically, upon user demand.

1.4 Major Cluster Design Issues

Unfortunately, a cluster-wide OS for complete resource sharing is not available yet. Middleware or OS extensions were developed at the user space to achieve SSI at selected functional levels. Without this middleware, cluster nodes cannot work together effectively to achieve cooperative computing. The software environments and applications must rely on the middleware to achieve high performance. The cluster benefits come from scalable performance, efficient message passing, high system availability, seamless fault tolerance, and cluster-wide job management, as summarized in Table 1.3.

2. Grid Computing Infrastructures

In the past 30 years, users have experienced a natural growth path from Internet to web and grid computing services. Internet services such as the Telnet command enables a local computer to connect to a remote computer. A web service such as HTTP enables remote access of remote web pages. Grid computing is envisioned to allow close interaction among applications running on distant computers simultaneously. Forbes Magazine has projected the global growth of the IT-based economy from $1 trillion in 2001 to $20 trillion by 2015. The evolution from Internet to web and grid services is certainly playing a major role in this growth.

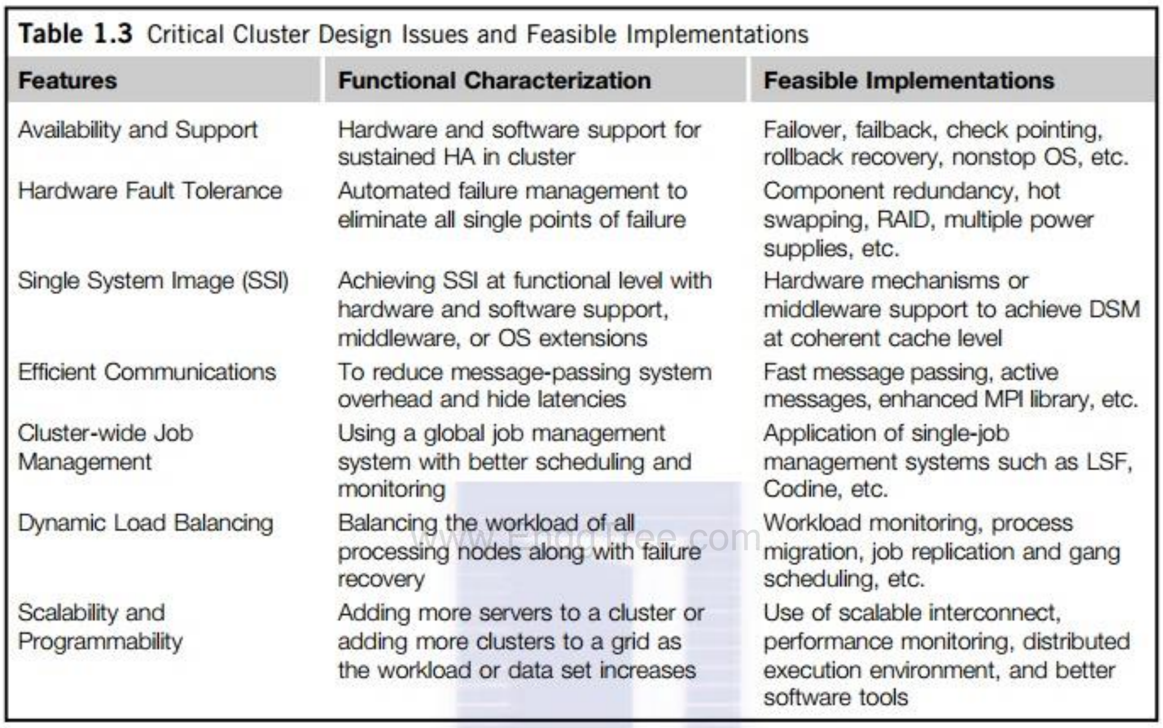


Table 1.3 Critical Cluster Design Issues and Feasible Implementations

Features	Functional Characterization	Feasible Implementations
Availability and Support	Hardware and software support for sustained HA in cluster	Failover, failback, check pointing, rollback recovery, nonstop OS, etc.
Hardware Fault Tolerance	Automated failure management to eliminate all single points of failure	Component redundancy, hot swapping, RAID, multiple power supplies, etc.
Single System Image (SSI)	Achieving SSI at functional level with hardware and software support, middleware, or OS extensions	Hardware mechanisms or middleware support to achieve DSM at coherent cache level
Efficient Communications	To reduce message-passing system overhead and hide latencies	Fast message passing, active messages, enhanced MPI library, etc.
Cluster-wide Job Management	Using a global job management system with better scheduling and monitoring	Application of single-job management systems such as LSF, Codine, etc.
Dynamic Load Balancing	Balancing the workload of all processing nodes along with failure recovery	Workload monitoring, process migration, job replication and gang scheduling, etc.
Scalability and Programmability	Adding more servers to a cluster or adding more clusters to a grid as the workload or data set increases	Use of scalable interconnect, performance monitoring, distributed execution environment, and better software tools

2.1 Computational Grids

Like an electric utility power grid, a computing grid offers an infrastructure that couples computers, software/middleware, special instruments, and people and sensors together. The grid is often con-structed across LAN, WAN, or Internet backbone networks at a regional, national, or global scale. Enterprises or organizations present grids as integrated computing resources. They can also be viewed as virtual platforms to support virtual organizations. The computers used in a grid are primarily workstations, servers, clusters, and supercomputers. Personal computers, laptops, and PDAs can be used as access devices to a grid system.

In Figure 1.16 shows an example computational grid built over multiple resource sites owned by different organizations. The resource sites offer complementary computing resources, including workstations, large servers, a mesh of processors, and Linux clusters to satisfy a chain of computational needs. The grid is built across various IP broadband networks including LANs and WANs already used by enterprises or organizations over the Internet. The grid is presented to users as an integrated resource pool as shown in the upper half of the figure.

2.2 Grid Families

Grid technology demands new distributed computing models, software/middleware support, network protocols, and hardware infrastructures. National grid projects are followed by industrial grid plat-form development by IBM, Microsoft, Sun, HP, Dell, Cisco, EMC, Platform Computing, and others. New grid service providers (GSPs) and new grid applications have emerged rapidly, similar to the growth of Internet and web services in the past two decades.

In Table 1.4, grid systems are classified in essentially two categories: computational or data grids and P2P grids.

lessons learned.

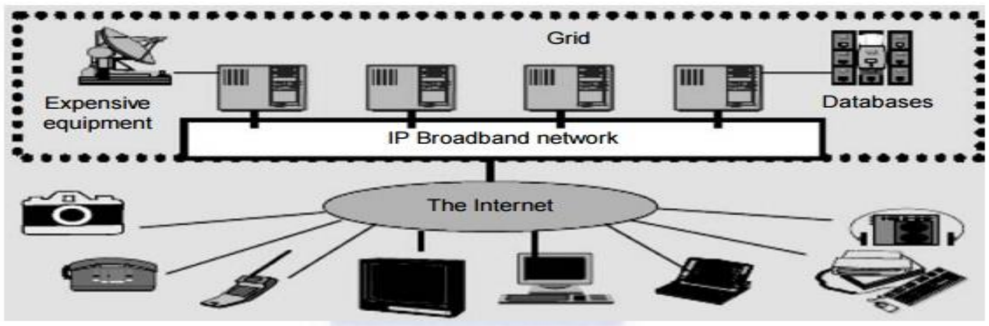


FIGURE 1.16

Computational grid or data grid providing computing utility, data, and information services through resource sharing and cooperation among participating organizations.

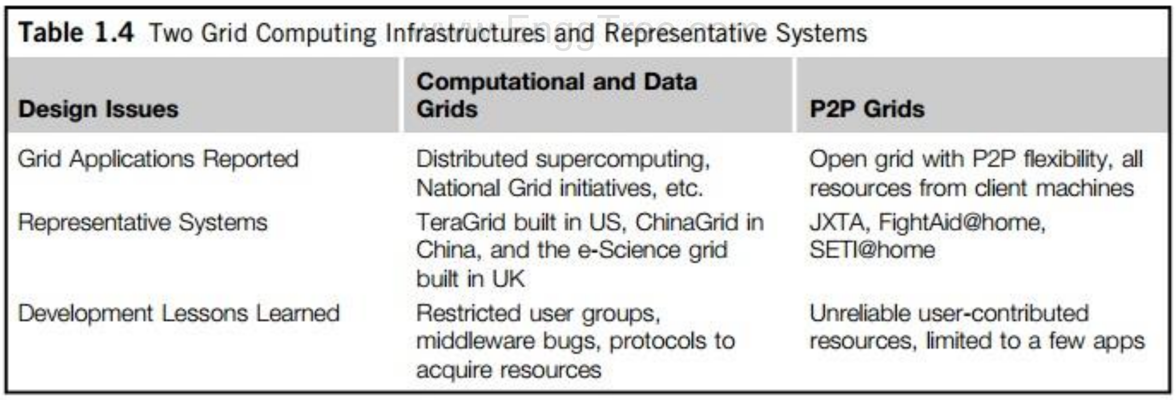


Table 1.4 Two Grid Computing Infrastructures and Representative Systems

Design Issues	Computational and Data Grids	P2P Grids
Grid Applications Reported	Distributed supercomputing, National Grid initiatives, etc.	Open grid with P2P flexibility, all resources from client machines
Representative Systems	TeraGrid built in US, ChinaGrid in China, and the e-Science grid built in UK	JXTA, FightAid@home, SETI@home
Development Lessons Learned	Restricted user groups, middleware bugs, protocols to acquire resources	Unreliable user-contributed resources, limited to a few apps

3. Peer-to-Peer Network Families

An example of a well-established distributed system is the client-server architecture. In this scenario, client machines (PCs and workstations) are connected to a central server for compute, e-mail, file access, and database applications. The P2P architecture offers a distributed model of networked systems. First, a P2P network is client-oriented instead of server-oriented. In this section, P2P systems are introduced at the physical level and overlay networks at the logical level.

3.1 P2P Systems

In a P2P system, every node acts as both a client and a server, providing part of the system resources. Peer machines are simply client computers connected to the Internet. All client machines act autonomously to join or leave the system freely. This implies that no master-slave relationship exists among the peers. No central coordination or central database is needed. In other words, no peer machine has a global view of the entire P2P system. The system is self-organizing with distributed control.

Figure 1.17 shows the architecture of a P2P network at two abstraction levels. Initially, the peers are totally unrelated. Each peer machine joins or leaves the P2P network voluntarily. Only the participating peers form the physical network at any time. Unlike the cluster or grid, a P2P network does not use a dedicated interconnection network. The physical network is simply an ad hoc network formed at various Internet domains randomly using the TCP/IP and NAI protocols. Thus, the physical network varies in size and topology dynamically due to the free membership in the P2P network.

3.2 Overlay Networks

Data items or files are distributed in the participating peers. Based on communication or file-sharing needs, the peer IDs form an overlay network at the logical level. This overlay is a virtual network

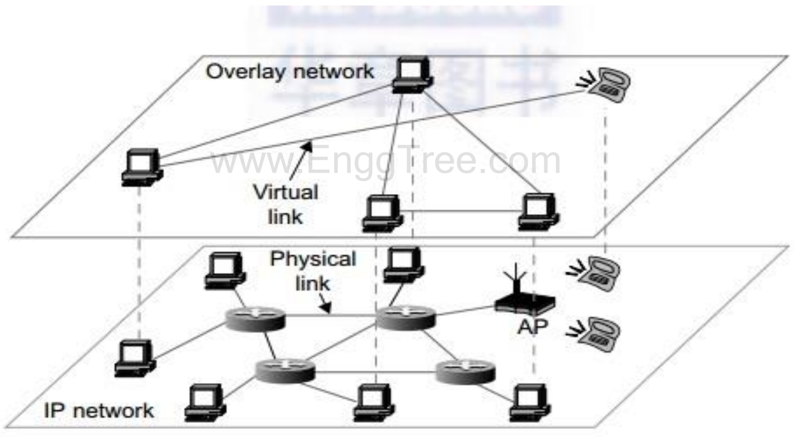


FIGURE 1.17

The structure of a P2P system by mapping a physical IP network to an overlay network built with virtual links.

formed by mapping each physical machine with its ID, logically, through a virtual mapping as shown in Figure 1.17. When a new peer joins the system, its peer ID is added as a node in the overlay network. When an existing peer leaves the system, its peer ID is removed from the overlay network automatically. Therefore, it is the P2P overlay network that characterizes the logical connectivity among the peers.

There are two types of overlay networks: unstructured and structured. An unstructured overlay network is characterized by a random graph. There is no fixed route to send messages or files among the nodes. Often, flooding is applied to send a query to all nodes in an unstructured overlay, thus resulting in heavy network traffic and nondeterministic search results. Structured overlay net-works follow certain connectivity topology and rules for inserting and removing nodes (peer IDs) from the overlay graph. Routing mechanisms are developed to take advantage of the structured overlays.

3.3 P2P Application Families

Based on application, P2P networks are classified into four groups, as shown in Table 1.5. The first family is for distributed file sharing of digital contents (music, videos, etc.) on the P2P network. This includes many popular P2P networks such as Gnutella, Napster, and BitTorrent, among others. Collaboration P2P networks include MSN or Skype chatting, instant messaging, and collaborative design, among others.

3.4 P2P Computing Challenges

P2P computing faces three types of heterogeneity problems in hardware, software, and network requirements. There are too many hardware models and architectures to select from; incompatibility exists between software and the OS; and different network connections and protocols

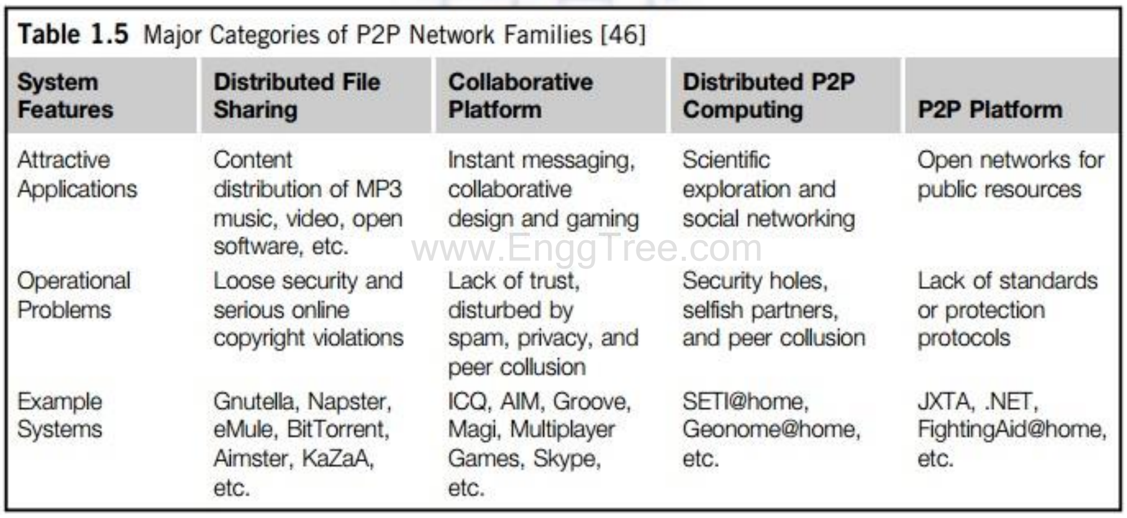


Table 1.5 Major Categories of P2P Network Families [46]

System Features	Distributed File Sharing	Collaborative Platform	Distributed P2P Computing	P2P Platform
Attractive Applications	Content distribution of MP3 music, video, open software, etc.	Instant messaging, collaborative design and gaming	Scientific exploration and social networking	Open networks for public resources
Operational Problems	Loose security and serious online copyright violations	Lack of trust, disturbed by spam, privacy, and peer collusion	Security holes, selfish partners, and peer collusion	Lack of standards or protection protocols
Example Systems	Gnutella, Napster, eMule, BitTorrent, Aimster, KaZaA, etc.	ICQ, AIM, Groove, Magi, Multiplayer Games, Skype, etc.	SETI@home, Geonome@home, etc.	JXTA, .NET, FightingAid@home, etc.

make it too complex to apply in real applications. We need system scalability as the workload increases. System scaling is directly related to performance and bandwidth. P2P networks do have these properties. Data location is also important to affect collective performance. Data locality, network proximity, and interoperability are three design objectives in distributed P2P applications.

3. **Internet clouds :**The idea is to move desktop computing to a service-oriented platform using server clusters and huge databases at data centers. Cloud computing leverages its low cost and simplicity to benefit both users and providers. Machine virtualization has enabled such cost-effectiveness. Cloud computing intends to satisfy many user Virtualized resources from data centers to form an Internet cloud, provisioned with hardware, software, storage, network, and services for paid users to run their applications.

NIST Cloud Computing Reference Architecture

4. Explain about the NIST Cloud Computing reference architecture. (May-2022)

The reference architecture model given by the National Institute of Standards and Technology (NIST). The model offers approaches for secure cloud adoption while contributing to cloud computing guidelines and standards.

The NIST team works closely with leading IT vendors, developers of standards, industries and other governmental agencies and industries at a global level to support effective cloud computing security standards and their further development. It is important to note that this NIST cloud reference architecture does not belong to any specific vendor products, services or some reference implementation, nor does it prevent further innovation in cloud technology.

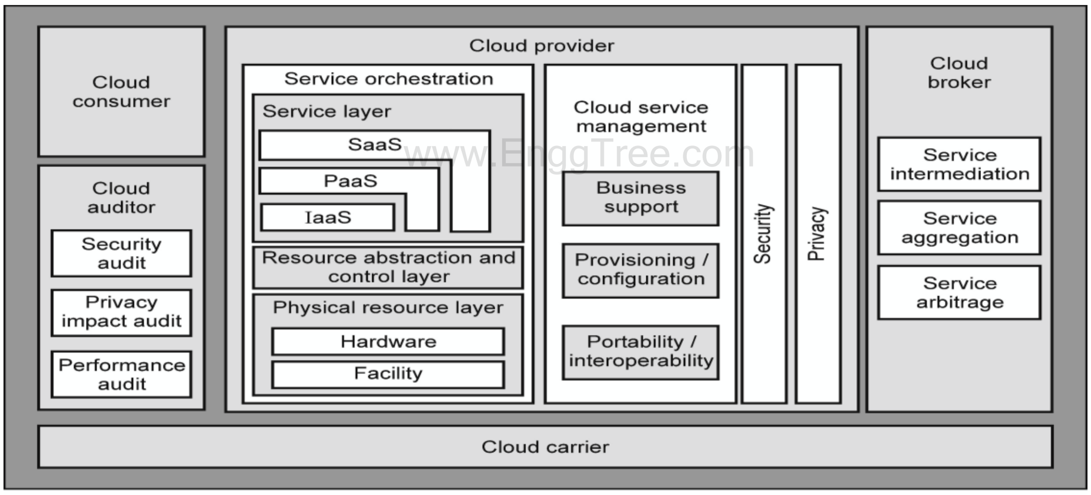


Fig. 1.2 : Conceptual cloud reference model showing different actors and entities

From Fig. 3.2.1, note that the cloud reference architecture includes five major actors :

- Cloud consumer
- Cloud provider
- Cloud auditor
- Cloud broker
- Cloud carrier

Each actor is an organization or entity plays an important role in a transaction or a process, or performs some important task in cloud computing. The interactions between these actors are illustrated in Fig. 1.3.

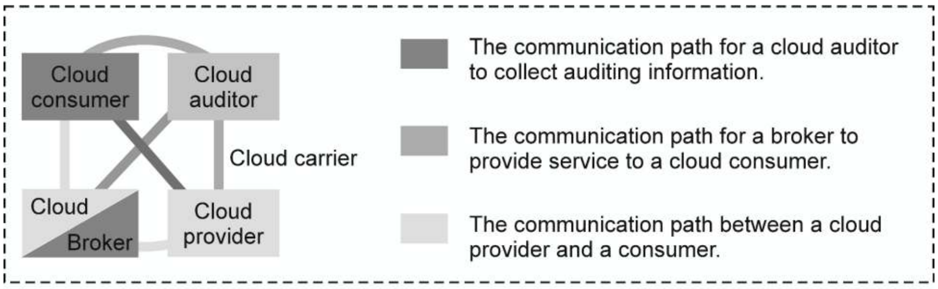

Cloud consumer
Cloud auditor
Cloud carrier
Cloud Broker
Cloud provider
The communication path for a cloud auditor to collect auditing information.
The communication path for a broker to provide service to a cloud consumer.
The communication path between a cloud provider and a consumer.

Fig. 1.3: Interactions between different actors in a cloud

Now, understand that a cloud consumer can request cloud services directly from a CSP or from a cloud broker. The cloud auditor independently audits and then contacts other actors to gather information. We will now discuss the role of each actor in detail.

Cloud Consumer

A cloud consumer is the most important stakeholder. The cloud service is built to support a cloud consumer. The cloud consumer uses the services from a CSP or person or asks an organization that maintains a business relationship. The consumer then verifies the service catalogue from the cloud provider and requests an appropriate service or sets up service contracts for using the service. The cloud consumer is billed for the service used.

Some typical usage scenarios include :

Example 1 : Cloud consumer requests the service from the broker instead of directly contacting the CSP. The cloud broker can then create a new service by combining multiple services or by enhancing an existing service. Here, the actual cloud provider is not visible to the cloud consumer. The consumer only interacts with the broker. This is illustrated in Fig. 1.4.

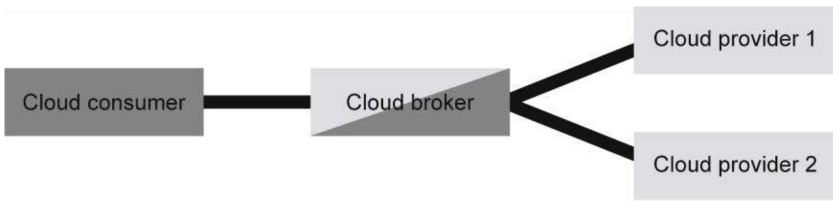


Fig. 1.4 : Cloud broker interacting with cloud consumer

Example 2 : In this scenario, the cloud carrier provides for connectivity and transports cloud services to consumers. This is illustrated in Fig. 1.5.

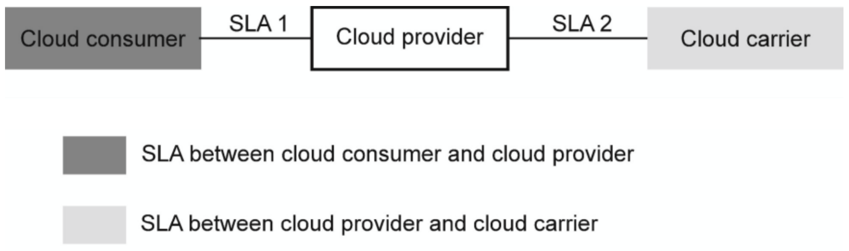

Cloud consumer
SLA 1
Cloud provider
SLA 2
Cloud carrier
SLA between cloud consumer and cloud provider
SLA between cloud provider and cloud carrier

Fig. 1.5 : Scenario for cloud carrier

In Fig. 1.2.4, the cloud provider participates by arranging two SLAs. One SLA is with the cloud provider (SLA2) and the second SLA is with the consumer (SLA1). Here, the cloud provider will have an arrangement (SLA) with the cloud carrier to have secured, encrypted connections. This ensures that the services are available for the consumer at a consistent level to fulfil service requests. Here, the provider can specify the requirements, such as flexibility, capability and functionalities in SLA2 to fulfil essential service requirements in SLA1.

Example 3 : In this usage scenario, the cloud auditor conducts independent evaluations for a cloud service. The evaluations will relate to operations and security of cloud service implementation. Here the cloud auditor interacts with both the cloud provider and consumer, as shown in Fig. 1.6.

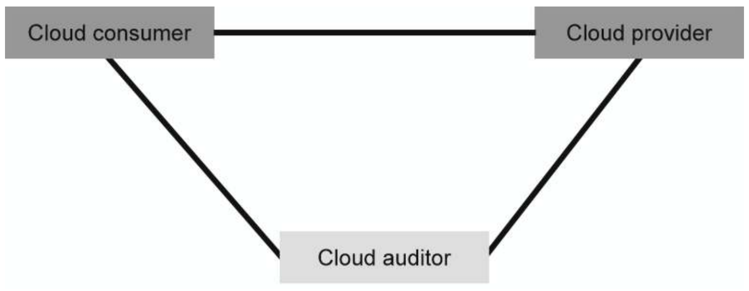


Fig. 1.6 : Usage scenario involving a cloud auditor

In all the given scenarios, the cloud consumer plays the most important role. Based on the service request, the activities of other players and usage scenarios can differ for other cloud consumers. Fig. 1.7 shows an example of available cloud services types.

In Fig. 1.7 note that SaaS applications are available over a network to all consumers. These consumers may be organisations with access to software applications, end users, app developers or administrators. Billing is based on the number of end users, the time of use, network bandwidth consumed and for the amount or volume of data stored.

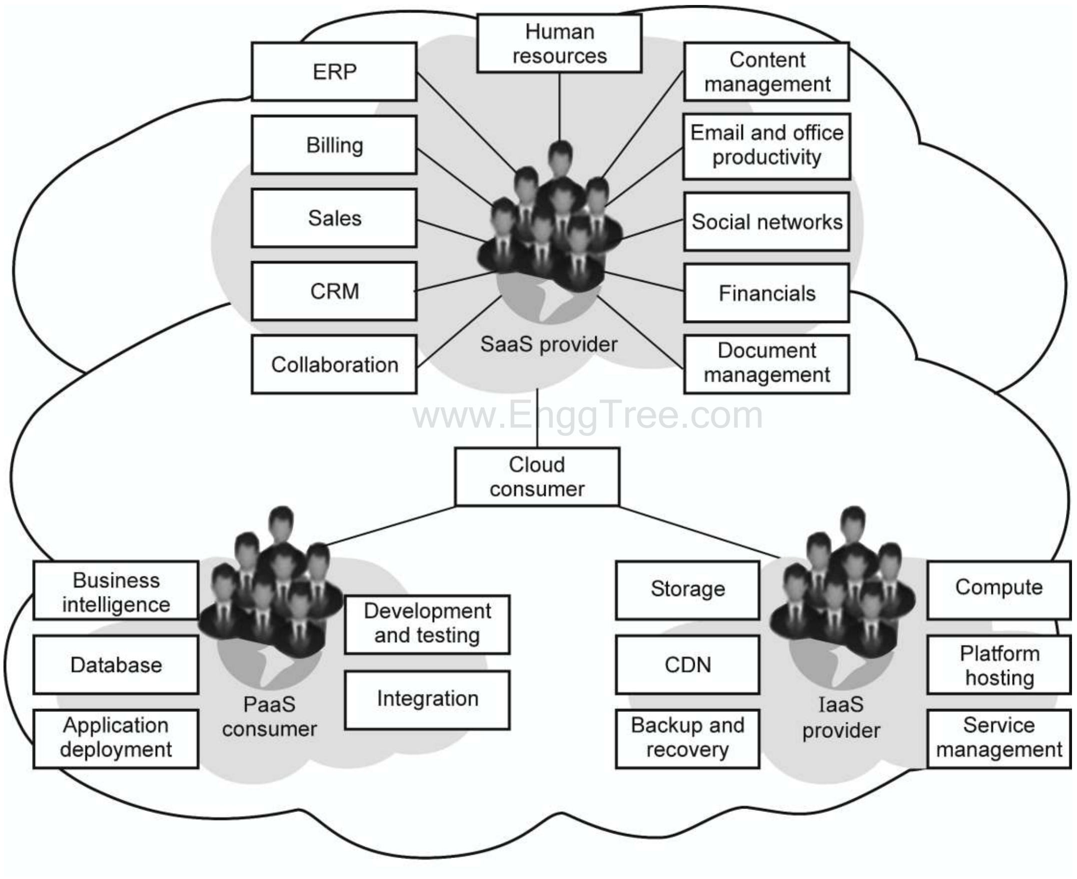


Fig. 1.7: Example of cloud services available to cloud consumers

PaaS consumers can utilize tools, execution resources, development IDEs made available by cloud providers. Using these resources, they can test, develop, manage, deploy and configure many applications that are hosted on a cloud. PaaS consumers are billed based on processing, database, storage, network resources consumed and for the duration of the platform used.

On the other hand, IaaS consumers can access virtual computers, network - attached storage, network components, processor resources and other computing resources that are deployed and run arbitrary software. IaaS consumers are billed based on the amount and duration of hardware resources consumed, number of IP addresses, volume of data stored, network bandwidth, and CPU hours used for a certain duration.

Cloud Provider

Cloud provider is an entity that offers cloud services to interested parties. A cloud provider manages the infrastructure needed for providing cloud services. The CSP also runs the software to provide services and organizes the service delivery to cloud consumers through networks.

SaaS providers then deploy, configure, maintain and update all operations of the software application on the cloud infrastructure, in order to ensure that services are provisioned and to fulfil cloud consumer service requests. SaaS providers assume most of the responsibilities associated with managing and controlling applications deployed on the infrastructure. On the other hand, SaaS consumers have no or limited administrative controls.

PaaS cloud providers manage the computing infrastructure and ensure that the platform runs the cloud software and implements databases, appropriate runtime software execution stack and other required middleware elements. They support development, deployment and the management of PaaS consumers by providing them with necessary tools such as IDEs, SDKs and others. PaaS providers have complete control of applications, settings of the hosting environment, but have lesser control over the infrastructure lying under the platform, network, servers, OS and storage.

Now, the IaaS CSP aggregates physical cloud resources such as networks, servers, storage and network hosting infrastructure. The provider operates the cloud software and makes all compute resources available to IaaS cloud consumer via a set of service interfaces, such as VMs and virtual network interfaces. The IaaS cloud provider will have control over the physical hardware and cloud software to enable provisioning and possible infrastructure services.

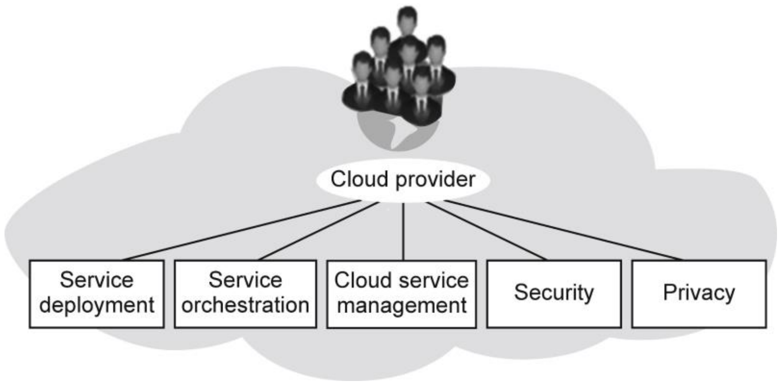


Fig. 1.8 : Major activities of a cloud provider

The major activities of a cloud provider include :

- **Service deployment:** Service deployment refers to provisioning private, public, hybrid and community cloud models.
- **Service orchestration:** Service orchestration implies the coordination, management of cloud infrastructure and arrangement to offer optimized capabilities of cloud services. The capabilities must be cost-effective in managing IT resources and must be determined by strategic business needs.
- **Cloud services management:** This activity involves all service-related functions needed to manage and operate the services requested or proposed by cloud consumers.
- **Security:** Security, which is a critical function in cloud computing, spans all layers in the reference architecture. Security must be enforced end-to-end. It has a wide range from physical to application security. CSPs must take care of security.
- **Privacy:** Privacy in cloud must be ensured at different levels, such as user privacy, data privacy, authorization and authentication and it must also have adequate assurance levels. Since clouds allow resources to be shared, privacy challenges are a big concern for consumers using clouds.

Cloud Auditor

The cloud auditor performs the task of independently evaluating cloud service controls to provide an honest opinion when requested. Cloud audits are done to validate standards conformance by reviewing the objective evidence. The auditor will examine services provided by the cloud provider for its security controls, privacy, performance, and so on.

Cloud Broker

The cloud broker collects service requests from cloud consumers and manages the use, performance, and delivery of cloud services. The cloud broker will also negotiate and manage the relationship between cloud providers and consumers. A cloud broker may provide services that fall into one of the following categories :

- **Service intermediation :** Here the cloud broker will improve some specific capabilities, and provide value added services to cloud consumers.
- **Service aggregation :** The cloud broker links and integrates different services into one or more new services.
- **Service Arbitrage :** This is similar to aggregation, except for the fact that services that are aggregated are not fixed. In service arbitrage, the broker has the liberty to choose services from different agencies.

Cloud Carrier

The cloud carrier tries to establish connectivity and transports cloud services between a cloud consumer and a cloud provider. Cloud carriers offer network access for consumers, by providing telecommunication links for accessing resources using other devices (laptops, computers, tablets, smartphones, etc.). Usually, a transport agent is an entity offering telecommunication carriers to a business organization to access resources. The cloud provider will set up SLAs with cloud carrier to ensure carrier transport is consistent with the level of SLA provided by the consumers. Cloud carriers provide secure and dedicated high - speed links with cloud providers and between different cloud
entities.

Actor	Definition
Cloud Consumer	A person or organization that maintains a business relationship with, and uses service from, Cloud Providers.
Cloud Provider	A person, organization, or entity responsible for making a service available to interested parties.
Cloud Carrier	An intermediary that provides connectivity and transport of cloud services from Cloud Providers to Cloud Consumers.

Cloud Auditor	A party that can conduct independent assessment of cloud services, information system operations, performance and security of the cloud implementation.
Cloud Broker	An entity that manages the use, performance and delivery of cloud services, and negotiates relationships between Cloud Providers and Cloud

3.Explain in detail about Cloud Computing Deployment Model with a neat diagram. Nov/Dec 2021(Nov/Dec 2022)

Cloud Deployment Models

A cloud deployment models are defined according to where the computing infrastructure resides and who controls the infrastructure. The NIST have classified cloud deployment models into four categories namely,

- **Public cloud**
- **Private cloud**
- **Hybrid cloud**
- **Community cloud**

They describe the way in which users can access the cloud services. Each cloud deployment model fits different organizational needs, so it's important that you pick a model that will suit your organization's needs. The four deployment models are characterized based on the functionality and accessibility of cloud services. The four deployment models of cloud computing are shown in Fig. 1.9**.**

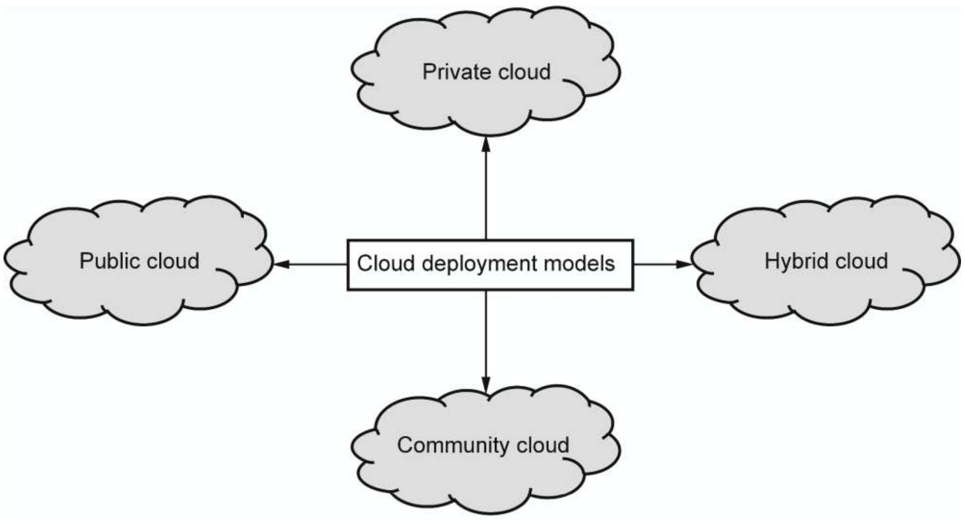


Fig. 1.9 : Four deployment models of cloud computing

Public Cloud

The public cloud services are runs over the internet. Therefore, the users who want cloud services have to have internet connection in their local device like thin client, thick client, mobile, laptop or desktop etc. The public cloud services are managed and maintained by the Cloud Service Providers (CSPs) or the Cloud Service Brokers (CSBs). The public cloud services are often offered on utility base pricing like subscription or pay- per-use model. The public cloud services are provided through internet and APIs. This model allows users to easily access the services without purchasing any specialize hardware or software. Any device which has web browser and internet connectivity can be a public cloud client. The popular public cloud service providers are Amazon web services, Microsoft azure and Google app engine, Salesforce etc.

Advantages of public cloud

1. It saves capital cost behind purchasing the server hardware's, operating systems and application software licenses.
2. There is no need of server administrators to take care of servers as they are kept at CSPs data center and managed by them.
3. No training is required to use or access the cloud services.
4. There is no upfront or setup cost is required.
5. A user gets easy access to multiple services under a single self - service portal.
6. Users have a choice to compare and select between the providers.
7. It is cheaper than in house cloud implementation because user have to pay for that they have used.
8. The resources are easily scalable.

Disadvantages of public cloud

1. There is lack of data security as data is stored on public data center and managed by third party data center vendors therefore there may be compromise of user's confidential data.
2. Expensive recovery of backup data.

3. User never comes to know where (at which location) their data gets stored, how that can be recovered and how many replicas of data have been created.

Private Cloud

The private cloud services are used by the organizations internally. Most of the times it run over the intranet connection. They are designed for a single organization therefore anyone within the organization can get access to data, services and web applications easily through local servers and local network but users outside the organizations cannot access them. This type of cloud services are hosted on intranet therefore users who are connected to that intranet get access to the services. The infrastructure for private cloud is fully managed and maintained by the organization itself.

It is much more secure than public cloud as it gives freedom to local administrators to write their own security policies for user's access. It also provides good level trust and privacy to the users. Private clouds are more expensive than public clouds due to the capital expenditure involved in acquiring and maintaining them. The well-known private cloud platforms are Openstack, Open nebula, Eucalyptus, VMware private cloud etc.

Advantages of private cloud

1. Speed of access is very high as services are provided through local servers over local network.
2. It is more secure than public cloud as security of cloud services are handled by local administrator.
3. It can be customized as per organizations need.
4. It does not require internet connection for access.
5. It is easy to manage than public cloud.

Disadvantages of private cloud

1. Implementation cost is very high as setup involves purchasing and installing servers, Hypervisors, Operating systems.
2. It requires administrators for managing and maintaining servers.
3. The scope of scalability is very limited.

Hybrid Cloud

The hybrid cloud services are composed of two or more clouds that offers the benefits of multiple deployment models. It mostly comprises on premise private cloud and off- premise public cloud to leverage benefits of both and allow users inside and outside to have access to it. The Hybrid cloud provides flexibility such that users can migrate their applications and services from private cloud to public cloud and vice versa. It becomes most favored in IT industry because of its eminent features like mobility, customized security, high throughput, scalability, disaster recovery, easy backup and replication across clouds, high availability and cost efficient etc. The popular hybrid clouds are AWS with eucalyptus, AWS with VMware cloud, Google cloud with Nutanix etc.

The limitations of hybrid cloud are compatibility of deployment models, vendor-lock in solutions, requires a common cloud management software and management of separate cloud platforms etc.

Community Cloud

The community cloud is basically the combination of one or more public, private or hybrid clouds, which are shared by many organizations for a single cause. The community cloud is setup between multiple organizations whose objective is same. The Infrastructure for community cloud is to be shared by several organizations within specific community with common security, compliance objectives which is managed by third party organizations or managed internally. The well-known community clouds are Salesforce, Google community cloud etc.

Comparison between various Cloud Deployment Models

The comparison between different deployment models of cloud computing are given in Table 1.3.1.

S	Feature	Public Cloud	Private Cloud	Hybrid Cloud	Community Cloud
1	**Scalability**	Very High	Limited	Very High	Limited
2	**Security**	Less Secure	Most Secure	Very Secure	Less Secure
3	**Performance**	Low to Medium	Good	Good	Medium
4	**Reliability**	Medium	High	Medium to High	Medium

5	**Upfront Cost**	Low	Very High	Medium	Medium
6	**Quality of Service**	Low	High	Medium	Medium
7	**Network**	Internet	Intranet	Intranet and Internet	Internet
8	**Availability**	For general public	Organizations internal staff	For general public and organizations internal Staff	For Community members
9	**Example**	Windows Azure, AWS etc.	Openstack, VMware cloud, CloudStack, Eucalyptus etc.	Combination of Openstack and AWS	salesforce community

Table 1.3.1 : Comparison between various Cloud Deployment Models

5. **Explain in detail about the three cloud models at different service levels of the cloud. (Or) Give the importance of cloud computing and elaborate the different types of services offered by it.)Nov/Dec 2021(Nov/Dec 2022).**

Cloud Service Models

A Cloud computing is meant to provide variety of services and applications for users over the internet or intranet.

The most widespread services of cloud computing are categorised into three service classes which are called cloud service models or cloud reference models or working models of cloud computing.

They are based on the abstraction level of the offered capabilities and the service model of the CSPs. The various service models are :

- **Infrastructure as a Service (IaaS)**
- **Platform as a Service (PaaS)**
- **Software as a Service (SaaS)**

The three service models of cloud computing and their functions are shown in Fig. 1.10.

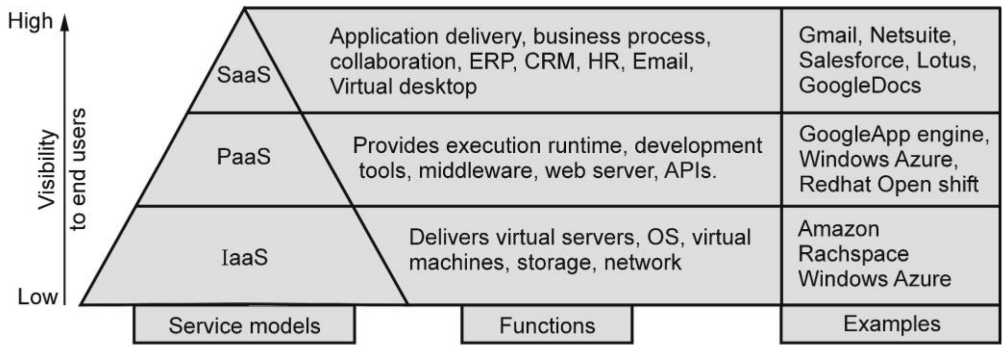

High
Low
Visibility to end users
SaaS
PaaS
IaaS
Application delivery, business process, collaboration, ERP, CRM, HR, Email, Virtual desktop
Provides execution runtime, development tools, middleware, web server, APIs.
Delivers virtual servers, OS, virtual machines, storage, network
Gmail, Netsuite, Salesforce, Lotus, GoogleDocs
GoogleApp engine, Windows Azure, Redhat Open shift
Amazon Rachspace Windows Azure
Service models
Functions
Examples

Fig. 1.10 : Cloud service models

From Fig. 1.10, we can see that the Infrastructure as a Service (IaaS) is the bottommost layer in the model and Software as a Service (SaaS) lies at the top.

The IaaS has lower level of abstraction and visibility, while SaaS has highest level of visibility.

The Fig. 1.11 represents the cloud stack organization from physical infrastructure to applications.

In this layered architecture, the abstraction levels are seen where higher layer services include the services of the underlying layer.

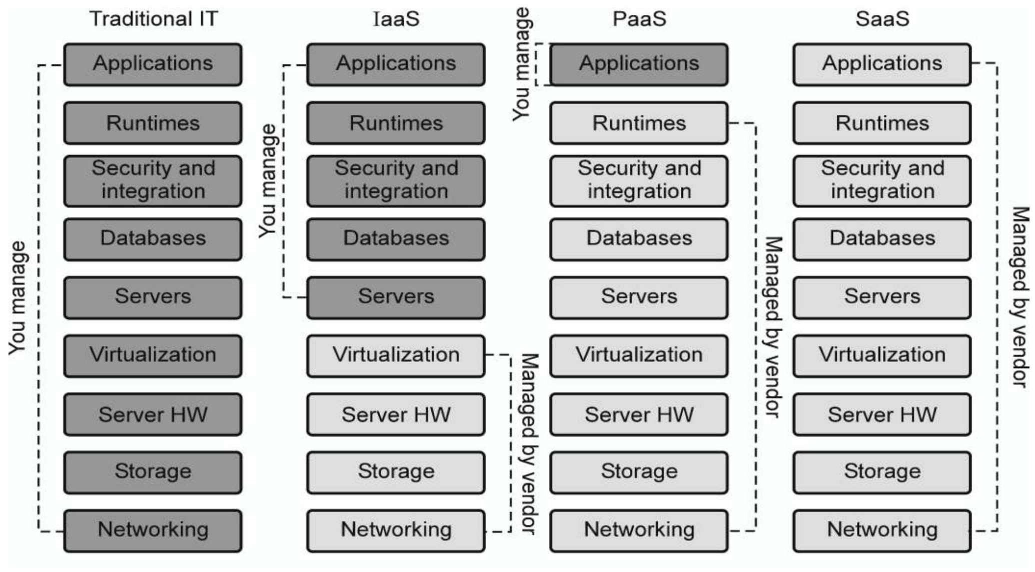


Fig. 1.11 : The cloud computing stack

As you can see in Fig. 1.4.2, the three services, IaaS, PaaS and SaaS, can exist independent of one another or may combine with one another at some layers. Different layers in every cloud computing model are either managed by the user or by the vendor (provider).

In case of the traditional IT model, all the layers or levels are managed by the user because he or she is solely responsible for managing and hosting the applications.

In case of IaaS, the top five layers are managed by the user, while the four lower layers (virtualisation, server hardware, storage and networking) are managed by vendors or providers. So, here, the user will be accountable for managing the operating system via applications and managing databases and security of applications.

The core middleware manages the physical resources and the VMs are deployed on top of them. This deployment will provide the features of pay-per-use services and multi-tenancy. Infrastructure services support cloud development environments and provide capabilities for application development and implementation.

It provides different libraries, models for programming, APIs, editors and so on to support application development. When this deployment is ready for the cloud, they can be used by end-users/ organisations. With this idea, let us further explore the different service models.

Infrastructure as a Service (IaaS)

- Infrastructure-as-a-Service (IaaS) can be defined as the use of servers, storage, computing power, network and virtualization to form utility like services for users.
- It is a cloud service model that provides hardware resources virtualized in the cloud. It provides virtual computing resources to the users through resource pool.
- In IaaS, the CSP owns all equipment, such as servers, storage disks, and network infrastructure.

- Developers use the IaaS service model to create virtual hardware on which the applications and/ or services are developed.
- Developers can create virtual private storage, virtual private servers, and virtual private networks by using IaaS.
- The private virtual systems contain software applications to complete the IaaS solution. The infrastructure of IaaS consists of communication networks, physical compute nodes, storage solutions and the pool of virtualized computing resources managed by a service provider.
- IaaS provides users with a web-based service that can be used to create, destroy and manage virtual machines and storage.

- Instead of purchasing extra servers, softwares, datacenter space or network equipment, IaaS enables on-demand provisioning of computational resources in the form of virtual machines in cloud data center. Some key providers of IaaS are Amazon Web Services (AWS), Microsoft Azure, GoGrid, Joyent, Rackspace etc. and some of the private cloud softwares through which IaaS can be setup are Openstack, Apache Cloud Stack, Eucalyptus, and VMware VSphere etc.

- In IaaS service delivery, workload is the fundamental component of the virtualised client. It simulates the capacity of a physical server to perform work. Hence, the work done is equal to the total number of Transaction Per Minute (TPM).
- In the case of hosted applications, the client runs on a dedicated server inside a server rack. It may also run on a standalone server.
- The user reserves an equivalent machine required to run workloads. The IaaS infrastructure runs the instances of the server in the data centre offering the service.

The resources for this server instance are drawn from a mix of virtualised systems, RAID disks, network and interface capacity. These are physical systems partitioned into logical

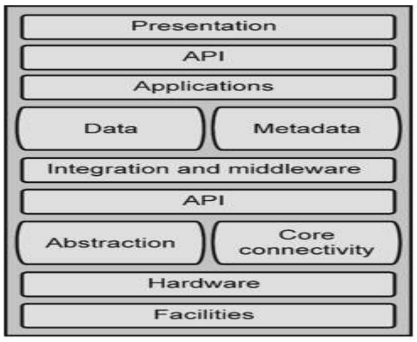


Fig. 1.12 : Components in IaaS service model (cloud security alliance)

The client in IaaS is allocated with its own private network. For example, Amazon EC2 enables this service to behave such that each server has its own separate network unless the user creates a virtual private cloud. If the EC2 deployment is scaled by adding additional networks on the infrastructure, it is easy to logically scale, but this can create an overhead as traffic gets routed between logical networks.

In IaaS, the customer has controls over the OS, storage and installed applications, but has limited control over network components. The user cannot control the underlying cloud infrastructure. Services offered by IaaS include web servers, server hosting, computer hardware, OS, virtual instances, load balancing, web servers and bandwidth provisioning. These services are useful during volatile demands and when there is a computing resource need for a new business launch or when the company may not want to buy hardware or if the organisation wants to expand.

Platform as a Service

- The Platform as a Service can be defined as a computing platform that allows the user to create web applications quickly and easily and without worrying about buying and maintaining the software and infrastructure.
- Platform-as-a-Service provides tools for development, deployment and testing the softwares, middleware solutions, databases, programming languages and APIs for developers to develop custom applications; without installing or configuring the development environment.
- The PaaS provides a platform to run web applications without installing them in a local machine i.e. the applications written by the users can be directly run on the PaaS cloud. It is built on the top of IaaS layer.

- The PaaS realizes many of the unique benefits like utility computing, hardware virtualization, dynamic resource allocation, low investment costs and pre-configured development environment. It has all the application typically required by the client deployed on it. Some key providers of PaaS clouds are Google App Engine, Microsoft Azure, NetSuite, Red hat Open shift etc.

- The PaaS realizes many of the unique benefits like utility computing, hardware virtualization, dynamic resource allocation, low investment costs and pre-configured development environment. It has all the application typically required by the client deployed on it. Some key providers of PaaS clouds are Google App Engine, Microsoft Azure, NetSuite, Red hat Open shift etc.

- The PaaS model includes the software environment where the developer can create custom solutions using development tools available with the PaaS platform. The components of a PaaS platform are shown in Fig. 1.13. Platforms can support specific development languages, frameworks for applications and other constructs. Also, PaaS provides tools and development environments to design applications. Usually, a fully Integrated

- Development Environment (IDE) is available as a PaaS service. For PaaS to be a cloud computing service, the platform supports user interface development. It also has many standards such as HTML, JavaScript, rich media and so on.
- In this model, users interact with the software and append and retrieve data, perform an action, obtain results from a process task and perform other actions allowed by the PaaS vendor.
- In this service model, the customer does not own any responsibility to maintain the hardware and software and the development environment.
- The applications created are the only interactions between the customer and the PaaS platform. The PaaS cloud provider owns responsibility for all the operational aspects, such as maintenance, updates, management of resources and product lifecycle.

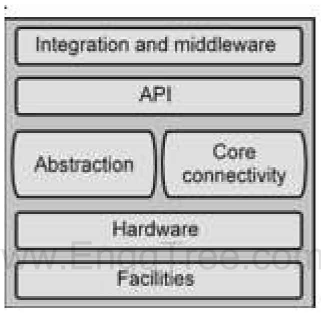


Fig. 1.13 : Components of PaaS

A PaaS customer can control services such as device integration, session management, content management, sandbox, and so on. In addition to these services, customer controls are also possible in Universal Description Discovery and Integration (UDDI), and platform independent Extensible Mark-up Language (XML) registry that allows registration and identification of web service apps.

Let us consider an example of Google app engine.

The platform allows developers to program apps using Google's published APIs. In this platform, Google defines the tools to be used within the development framework, the file system structure and data stores. A similar PaaS offering is given by Force.com, another vendor that is based on the Salesforce.com development platform for the latter's SaaS offerings.Force.com provides an add - on development environment.

In PaaS, note that developers can build an app with Python and Google API. Here, the PaaS vendor is the developer who offers a complete solution to the user. For instance, Google acts as a PaaS vendor and offers web service apps to users. Other examples are : Google Earth, Google Maps, Gmail, etc.

PaaS has a few disadvantages. It locks the developer and the PaaS platform in a solution specific to a platform vendor. For example, an application developed in Python using Google API on Google App Engine might work only in that environment.

PaaS is also useful in the following situations :

- When the application must be portable.
- When proprietary programming languages are used.
- When there is a need for custom hardware and software.

Major PaaS applications include software development projects where developers and users collaborate to develop applications and automate testing services.

Part-A

1. **List out the major functionalities of cloud computing. (or) Mention the characteristic features of cloud. (Apr/May'17)(May-2022)**
   - The cloud will free users to focus on user application development and create business value by outsourcing job execution to cloud providers. The computations (programs) are sent to where the data is located, rather than copying the data to millions of desktops as in the traditional approach.
   - Cloud computing avoids large data movement, resulting in much better network bandwidth utilization.
   - Furthermore, machine virtualization has enhanced resource utilization, increased application flexibility, and reduced the total cost of using virtualized data-center resources.
   - The cloud offers significant benefit to IT companies by freeing them from the low-level task of setting up the hardware (servers) and managing the system software.
2. **Write short notes on Research Compute Cloud (RC2) / Why do we need hybrid cloud? [NOV / DEC'16](Nov/Dec 2021)**

   Research Compute Cloud (RC2) shown below is a private cloud, built by IBM, that interconnects the computing and IT resources at eight IBM Research Centers scattered throughout the United States, Europe, and Asia. A hybrid cloud provides access to clients, the partner network, and third parties. Public clouds promote standardization, preserve capital investment, and offer application flexibility. Private clouds attempt to achieve customization and offer higher efficiency, resiliency, security, and privacy. Hybrid clouds operate in the middle, with many compromises in terms of resource sharing.

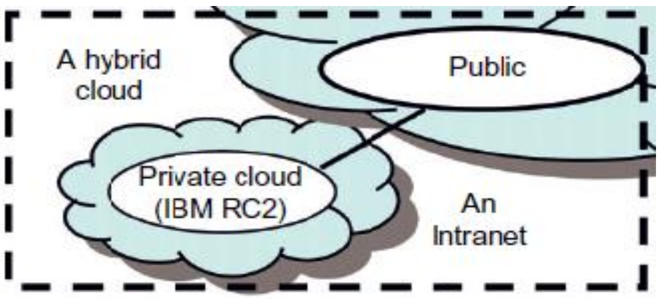


Fig: Private cloud

3. Define Platform as a Service (PaaS) Nov/Dec 2020

Platform as a service (PaaS) is a category of cloud computing services that provides a platform allowing customers to develop, run, and manage applications without the complexity of building and maintaining the infrastructure typically associated with developing and launching an application. PaaS can be delivered in two ways: as a public cloud service from a provider, where the consumer controls software deployment with minimal configuration options, and the provider provides the networks, servers, storage, OS, 'middleware'. as software deployed on a public infrastructure as a service.

4. Highlight the cloud platform services offered byPaaS services.

Cloud Name	**Languages and Developer Tools**	**Programming Models Supported by Provider**	**Target Applications and Storage Option**
Google App Engine	Python, Java, and Eclipse-based IDE	MapReduce, web programming on demand	Web applications and BigTable storage
Salesforce.com's Force.com	Apex, Eclipse-based IDE, web-based Wizard	Workflow, Excel-like formula, Web programming on demand	Business applications such as CRM
Microsoft Azure	NET, Azure tools for MS Visual Studio	Unrestricted model	Enterprise and web applications
Amazon Elastic MapReduce	Hive, Pig, Cascading, Java, Ruby, Perl, Python, PHP, R, C++	MapReduce	Data processing and e-commerce

Aneka.	.NET, stand-alone SDK	Threads, task, MapReduce	NET enterprise applications, HPC

<u>Pros and Cons of cloud computing</u>

5. Mention the six main challenges in cloud architecture development.(May-2022)

- Challenge 1—Service Availability and Data Lock-in Problem
- Challenge 2—Data Privacy and Security Concerns
- Challenge 3—Unpredictable Performance and Bottlenecks
- Challenge 4—Distributed Storage and Widespread Software Bugs
- Challenge 5—Cloud Scalability, Interoperability, and Standardization
- Challenge 6—Software Licensing and Reputation Sharing

6. Outline the key challenges associated in the process of storing images in cloud.(Nov/Dec 2021).

As cloud grows in popularity, it has become common to deploy applications in the cloud and provide them to end users. At the same time, the trend of using serverless architecture means that an unspecified number of end users can seamlessly access resources in the cloud.

Getting started building an image upload feature Before diving into the GCP components needed to implement this service, let's define our requirements:

- Use managed services as much as possible
- Enable only authenticated users to upload files
- Validate/filter the content uploaded by users.

7. How Hybrid Clouds are Formed?(May-2023)

Hybrid cloud refers to a mixed computing, storage, and services environment made up of on-premises infrastructure, private cloud services, and a public cloud—such as Amazon Web Services (AWS) or Microsoft Azure—with orchestration among the various platforms.

8. Mention the design goals of cloud platform.?(May-2023)

Cloud Platform Design Goals: The major goals of a cloud computing platform are scalability, efficiency, VZ, and reliability. A cloud platform manager receives the user requests, finds the resources, and calls the provisioning services to allocate the appropriate amount of resources for the job.

9. **Differentiate Public Cloud and Private Cloud .(Nov-2022)**

Public Cloud	Private Cloud
Hosted at Service provider Site	Hosted at Enterprise or Service provider Site
Supports connectivity over Internet	Supports connectivity over Internet/Private network (WAN)
Suitable for information not very sensitive	Suitable for very sensitive information
Cheaper than private cloud	Costlier than public cloud
Utilizes shared infrastructure	Doesn't utilize shared infrastructure
Supports multiple customers	Supports one customer
Requires higher level of security	Requires medium level of security.
May use vlans , access list , VRF lite , MPLS etc. to logically segregate different customer compute environment	Doesn't have to be too sensitive on logical segregation of customer data since only one customer dedicated environment.
Shared servers	Dedicated servers
No Dedicated proactive monitoring	Dedicated proactive monitoring
Fixed cost	Variable TCO
Multitenant architecture	Dedicated customer architecture.

10. what are the different layers?

The different layers of cloud computing are:

- **SaaS:** Software as a Service (SaaS), it provides users access directly to the cloud application without installing anything on the system.

- **IaaS:** Infrastructure as a service, it provides the infrastructure in terms of hardware like memory, processor speed etc.
- **PaaS:** Platform as a service, it provides cloud application platform for the developers

11. What are the different modes of software as a service (SaaS)?

- **Simple multi-tenancy:** In this each user has independent resources and are different from other users, it is an efficient mode.
- **Fine grain multi-tenancy:** In this type, the resources can be shared by many but the functionality remains the same.

12. How important is the platform as a service?

Platform as a service or PAAS is an important layer in cloud computing. It provides application platform for providers. It is responsible for providing complete virtualization of the infrastructure layer and makes it work like a single server.

13. What is a cloud service?

Cloud service is used to build cloud applications using the server in a network through internet. It provides the facility of using the cloud application without installing it on the computer. It also reduces the maintenance and support of the application which are developed using cloud service.

14. List down the three basic clouds in cloud computing?

- Professional cloud
- Personal cloud
- Performance cloud

15. What are the resources that are provided by it?

IAAS (Infrastructure As A Service) provides virtual and physical resources that are used to build a cloud. It deals with the complexities of deploying and maintaining of the services provided by this layer. Here the infrastructure is the servers, storage and other hardware systems.

16. What are the business benefits involved in cloud architecture?

The benefits involved in cloud architecture is

- Zero infrastructure investment
- Just in time infrastructure
- More efficient resource utilization

17.What are the characteristics of cloud architecture that separates it from traditional one?

The characteristics that make cloud architecture above traditional architecture is According to the demand cloud architecture provides the hardware requirement Cloud architecture is capable of scaling the resource on demand Cloud architecture is capable of managing and handling dynamic workloads without failure

18.What are the building blocks?

- Reference architecture
- Technical architecture
- Deployment operation architecture

UNIT II

VIRTUALIZATION BASICS

Virtual Machine Basics – Taxonomy of Virtual Machines – Hypervisor – Key Concepts – Virtualization structure – Implementation levels of virtualization – Virtualization Types: Full Virtualization – Para Virtualization – Hardware Virtualization – Virtualization of CPU, Memory and I/O devices.

Virtual Machine Basics:

1. **Explain in detail about virtual Machine and its types**

Virtual Machine:

Virtual Machine can be defined as an **emulation of the computer systems** in computing. Virtual Machine is based on computer architectures. It also gives the functionality of physical computers. The implementation of VM may consider specialized software, hardware, or a combination of both.

Virtual Machine Basics To understand what a virtual machine is, we must first discuss what is meant by machine, and, as pointed out earlier, the meaning of "machine" is a matter of perspective. From the perspective of a process executing a user program, the machine consists of a logical memory address space that has been assigned to the process, along with user-level registers and instructions that allow the execution of code belonging to the process.

The I/O part of the machine is visible only through the operating system, and the only way the process can interact with the I/O system is via operating system calls, often through libraries that execute as part of the process. Processes are usually transient in nature (although not always). They are created, execute for a period of time, perhaps spawn other processes along the way, and eventually terminate.

To summarize, the machine, from the prospective of a process, is a combination of the operating system and the underlying user-level hardware. The ABI provides the interface between the process and the machine.

A system is a full execution environment that can simultaneously support a number of processes potentially belonging to different users. All the processes share a file system and other I/O resources. The system environment persists over time (with occasional reboots) as processes come and go. The system allocates physical memory and I/O resources to the processes and allows the processes to interact with their resources via an OS that is part of the system. Hence, the machine, from the perspective of a system, is implemented by the underlying hardware alone, and the ISA provides the interface between the system and the machine.

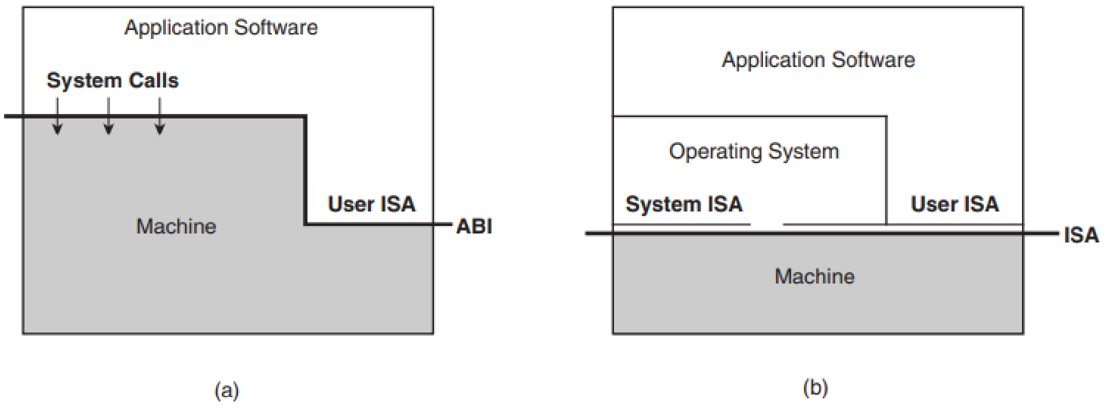


Fig:2.1 Virtual Machines

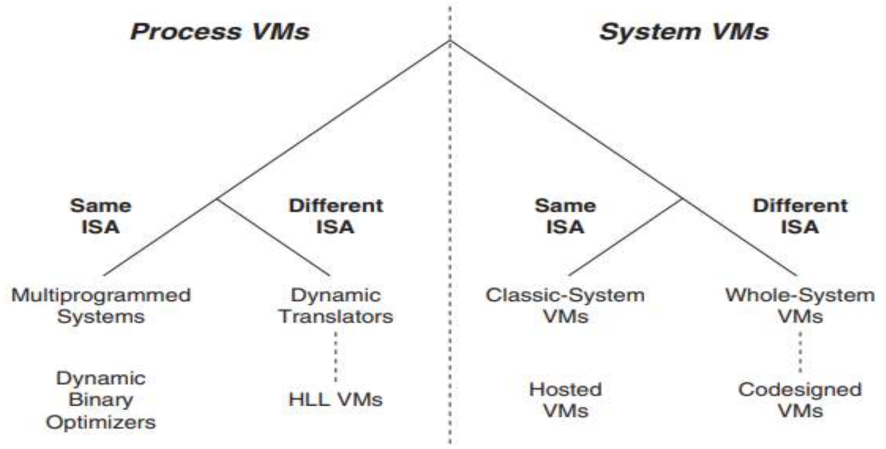
Taxonomy of Virtual Machines:

A Taxonomy We have just described a rather broad array of VMs, with different goals and implementations. To put them in perspective and organize the common implementation issues, we introduce a taxonomy illustrated in Figure 2.2.

First, VMs are divided into the two major types: process VMs and system VMs. In the first type, the VM supports an ABI — user instructions plus system calls; in the second, the VM supports a complete ISA — both user and system instructions. Finer divisions in the taxonomy are based on whether the guest and host use the same ISA.

On the left-hand side of Figures are process VMs. These include VMs where the host and guest instruction sets are the same. In the figure, we identify two examples. The first is multiprogrammed systems, as already supported on most of today's systems. The second is same-ISA dynamic binary optimizers, which transform guest instructions only by optimizing them and then execute them natively. For process VMs where the guest and host ISAs are different, we also give two examples. These are dynamic translators and HLL VMs. HLL VMs are connected to the VM taxonomy via a "dotted line" because their process-level interface is at a different, higher level than the other process VMs. On the right-hand side of the figure are system VMs. If the guest and host use the same ISA, examples include "classic" system VMs and hosted VMs. In these VMs, the objective is providing replicated, isolated system environments.

The primary difference between classic and hosted VMs is the VMM implementation rather than the function provided to the user. Examples of system VMs where the guest and host ISAs are different include whole-system VMs and code signed VMs. With whole-system VMs, performance is often of secondary importance compared to accurate functionality, while with code signed VMs, performance (and power efficiency) are often major goals. In the figure, code signed VMs are connected using dotted lines because their interface is typically at a lower level than other system VMs.

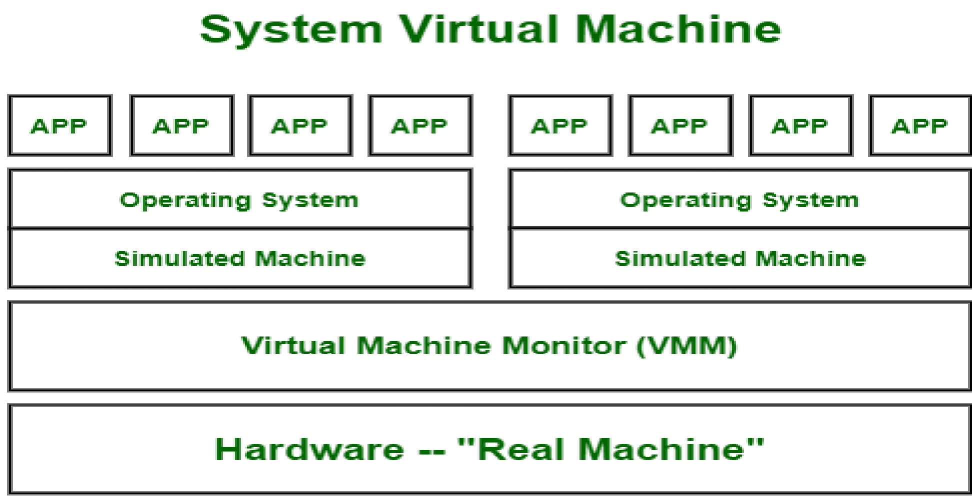


3 A Taxonomy of Virtual Machines.

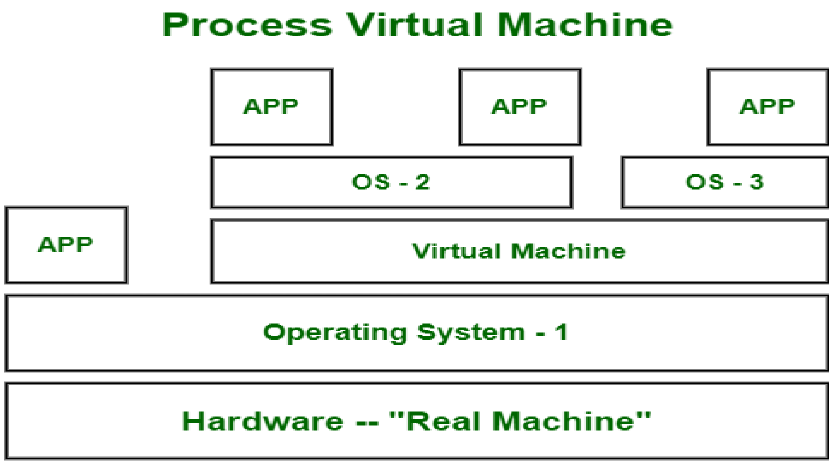
Fig 2.2 A Taxonomy of Virtual Machines

Types of Virtual Machines : You can classify virtual machines into two types:

1. System Virtual Machine: These types of virtual machines gives us complete system platform and gives the execution of the complete virtual operating system. Just like virtual box, system virtual machine is providing an environment for an OS to be installed completely. We can see in below image that our hardware of Real Machine is being distributed between two simulated operating systems by Virtual machine monitor. And then some programs, processes are going on in that distributed hardware of simulated machines separately.



2. Process Virtual Machine : While process virtual machines, unlike system virtual machine, does not provide us with the facility to install the virtual operating system completely. Rather it creates virtual environment of that OS while using some app or program and this environment will be destroyed as soon as we exit from that app. Like in below image, there are some apps running on main OS as well some virtual machines are created to run other apps. This shows that as those programs required different OS, process virtual machine provided them with that for the time being those programs are running. **Example –** Wine software in Linux helps to run Windows applications.



Virtual Machine Language : It's type of language which can be understood by different operating systems. It is platform-independent. Just like to run any programming language (C, python, or java) we need specific compiler that actually converts that code into system understandable code (also known as byte code). The same virtual machine language works. If we want to use code that can be executed on different types of operating systems like (Windows, Linux, etc) then virtual machine language will be helpful.

HYPERVISOR:

2. Explain in detail about Hardware based Virtualization.(or)Give the Virtualization Structure and Explain the various types of Virtualization.(May-2023)

Each instance of operating system called Virtual Machine (VM) and operating system runs inside virtual machine is called guest operating system. Depending on the position of the virtualization layer, there are two classes of VM architectures, namely the hypervisor architectures like bare-metal or host- based. The hypervisor is the software used for doing virtualization also known as the VMM (Virtual Machine Monitor). The hypervisor software provides two different structures of Virtualization namely Hosted structure (also called Type

2 Virtualization) and Bare-Metal structure (also called Type 1 Virtualization) .

Hosted Structure (Type II)(Hypervisor)

In hosted structure, the guest OS and applications run on the top of base or host OS with the help of VMM (called Hypervisor). The VMM stays between the base OS and guest OS. This approach provides better compatibility of hardware because the base OS is responsible for providing hardware drivers to guest OS instead of the VMM. In this type, hypervisor has to rely on host OS for pass through permissions to access hardware. In many cases, hosted hypervisor needs emulator, which lies between guest OS and VMM to translate the instructions in native format. The hosted structure is shown in Fig. 2.2.1.

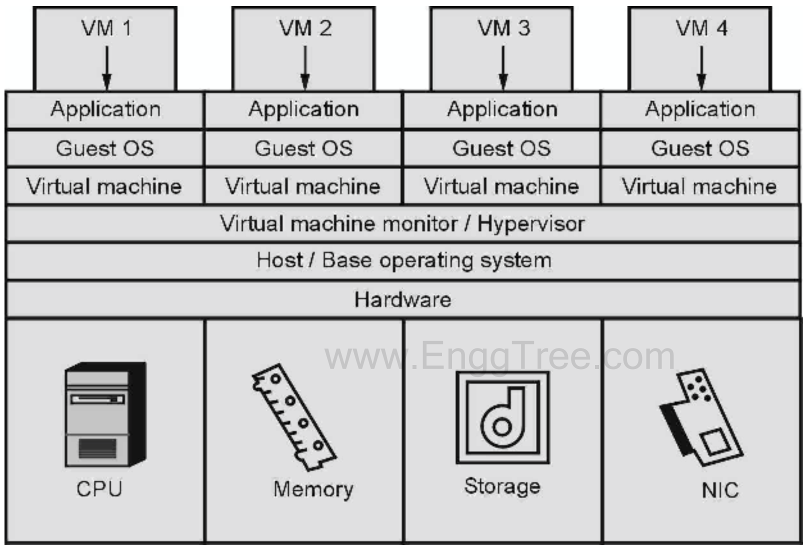
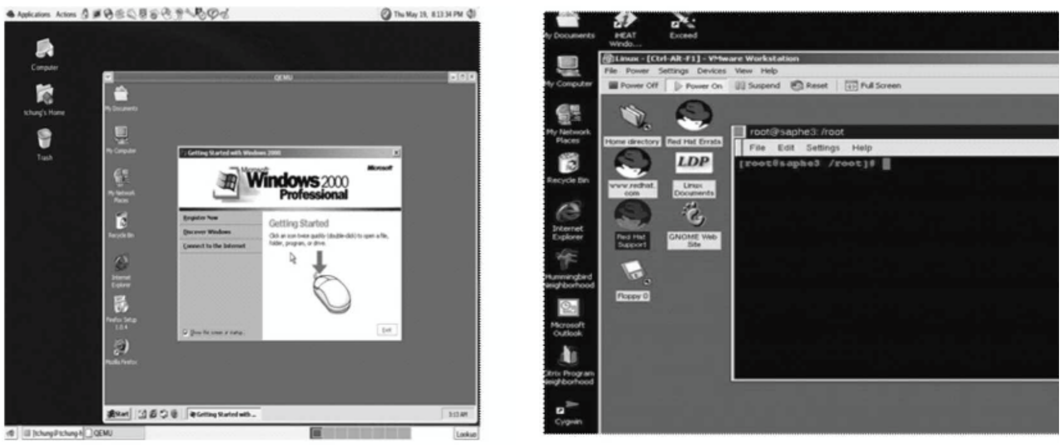


Fig. 2.2.1 Hosted Structure (Type II Hypervisor)

To implement Hosted structure, a base OS needs to be installed first over which VMM can be installed. The hosted structure is simple solution to run multiple desktop OS independently. Fig. 2.2.2 (a) and (b) shows Windows running on Linux base OS and Linux running on Windows base OS using hosted Hypervisor


Windows 2000 Professional
Getting Started
QEMU
Linux - [Ctrl-Alt-F1] - VMware Workstation
Power Off
Power On
Suspend
Reset
Full Screen

```
root@saphe3:/root
[root@saphe3 /root]#
```

(a) Windows running inside Linux QEMU Hypervisor

(b) Linux running Inside Windows on VMwar Hypervisor

Fig. 2.2.2 Hosted Hypervisors

The popular hosted hypervisors are QEMU, VMware Workstation, Microsoft Virtual PC, Oracle VirtualBox etc.

The advantages of hosted structure are

- ✓ It is easy to install and manage without disturbing host systems hardware.
- ✓ It supports legacy operating systems and applications.
- ✓ It provides ease of use with greater hardware compatibility.
- ✓ It does not require to install any drivers for IO devices as they are installed through built-in driver stack.
- ✓ It can be used for testing beta software.
- ✓ The hosted hypervisors are usually free software and can be run on user workstations.

The disadvantages of hosted structure are

- ✓ It does not allow guest OS to directly access the hardware instead it has to go through base OS, which increases resource overhead.
- ✓ It has very slow and degraded virtual machines performance due to relying on intermediate host OS for getting hardware access.
- ✓ It doesn't scale up beyond the limit.

Bare-Metal Structure (Type I)(or) Naïve Bare Metal Structure:

- ✓ In Bare-Metal Structure, the VMM can be directly installed on the top of Hardware, therefore no intermediate host OS is needed. The VMM can directly communicate with the hardware and does not rely on the host system for pass through permission which results in better performance, scalability and stability. The Bare-Metal structure is shown in Fig. 2.2.3. (See Fig. 2.2.3 on next page).
- ✓ Bare-metal virtualization is mostly used in enterprise data centers for getting the advanced features like resource pooling, high availability, disaster recovery and security.

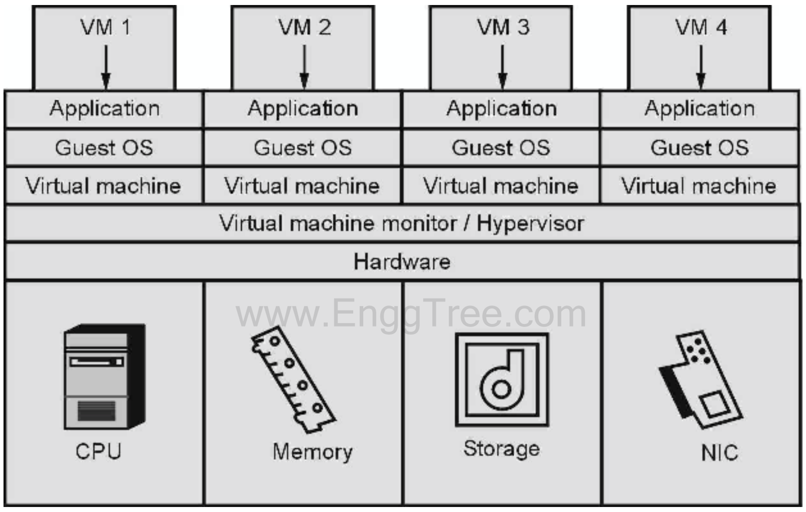
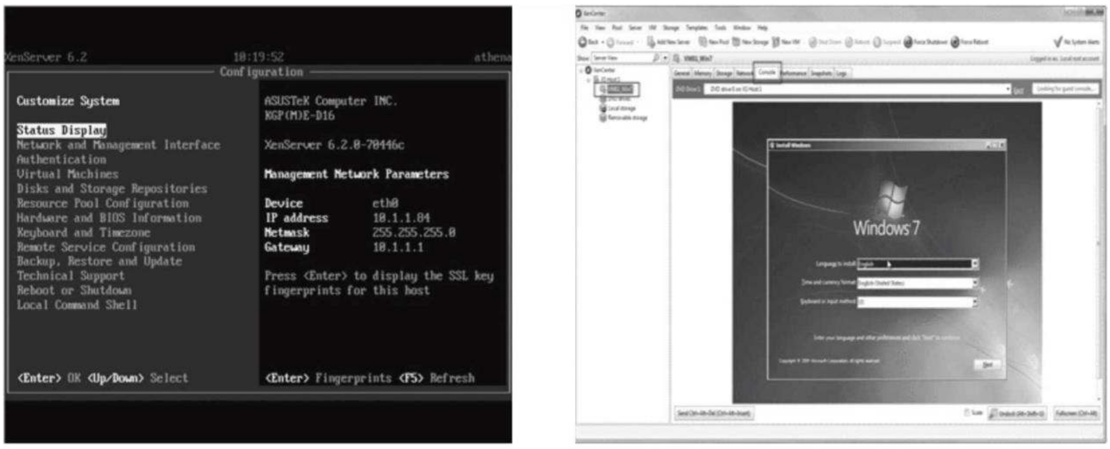


Fig. 2.2.3 Bare-Metal Structure (Type-I Hypervisor)



```
XenServer 6.2                    10:19:52                    athena
─────────────────────── Configuration ───────────────────────
Customize System                 ASUSTeK Computer INC.
                                 KGP(M)E-D16
Status Display
Network and Management Interface XenServer 6.2.0-70446c
Authentication
Virtual Machines                 Management Network Parameters
Disks and Storage Repositories
Resource Pool Configuration      Device          eth0
Hardware and BIOS Information    IP address      10.1.1.84
Keyboard and Timezone            Netmask         255.255.255.0
Remote Service Configuration     Gateway         10.1.1.1
Backup, Restore and Update
Technical Support                Press <Enter> to display the SSL key
Reboot or Shutdown               fingerprints for this host
Local Command Shell

<Enter> OK <Up/Down> Select      <Enter> Fingerprints <F5> Refresh
```

(a) Post installation screen of Xen server running on server hardware

(b) Xen center showing Windows 7 VM running on Xen server

Fig. 2.2.4 Bare-Metal Xen Server Hypervisor

The popular Bare-Metal Hypervisors are Citrix Xen Server, VMware ESXI and Microsoft Hyper V.

The advantages of Bare-Metal structure are

- It is faster in performance and more efficient to use.
- It provides enterprise features like high scalability, disaster recovery and high availability.
- It has high processing power due to the resource pooling.

Implementation Levels of Virtualization

3. **Discuss in detail about the categories of hardware virtualization depending on implementation technologies. Nov/Dec 2021(or)Discuss how Virtualization implemented in different layers of cloud in detail.(May-2022*)***

The virtualization is implemented at various levels by creating a software abstraction layer between host OS and Guest OS. The main function of software layer is to virtualize physical hardware of host machine in to virtual resources used by VMs by using various operational layers. The different levels at which the virtualization can be implemented is shown in Fig. 2.3.1.

There are five implementation levels of virtualization, that are Instruction Set Architecture (ISA) level, Hardware level, Operating System level, Library support level and Application level which are explained as follows.

1) Instruction Set Architecture Level

- Virtualization at the instruction set architecture level is implemented by emulating an instruction set architecture completely on software stack. An emulator tries to execute instructions issued by the guest machine (the virtual machine that is being emulated) by translating them to a set of native instructions and then executing them on the available hardware.

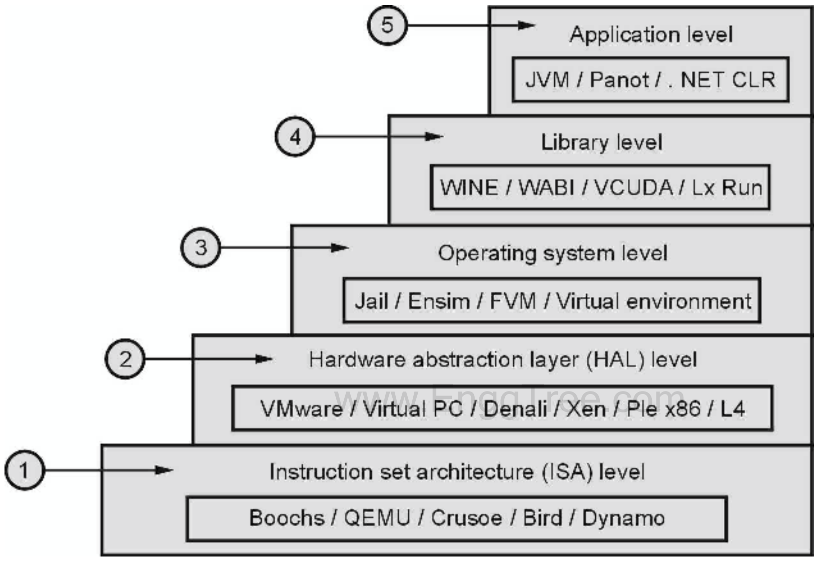

5
Application level
JVM / Panot / . NET CLR
4
Library level
WINE / WABI / VCUDA / Lx Run
3
Operating system level
Jail / Ensim / FVM / Virtual environment
2
Hardware abstraction layer (HAL) level
VMware / Virtual PC / Denali / Xen / Ple x86 / L4
1
Instruction set architecture (ISA) level
Boochs / QEMU / Crusoe / Bird / Dynamo

Fig. 2.3.1 Implementation Levels of Virtualization

- That is emulator works by translating instructions from the guest platform to instructions of the host platform. These instructions would include both processor oriented (add, sub, jump etc.), and the I/O specific (IN/OUT) instructions for the devices. Although this virtual machine architecture works fine in terms of simplicity and robustness, it has its own pros and cons.
- The advantages of ISA are, it provides ease of implementation while dealing with multiple platforms and it can easily provide infrastructure through which one can create virtual machines based on x86 platforms such as Sparc and Alpha. The disadvantage of ISA is since every instruction issued by the emulated computer needs to be interpreted in software first which degrades the performance.

- The popular emulators of ISA level virtualization are :

a) Boochs

It is a highly portable emulator that can be run on most popular platforms that include x86, PowerPC, Alpha, Sun, and MIPS. It can be compiled to emulate most of the versions of x86 machines including 386, 486, Pentium, Pentium Pro or AMD64 CPU, including optional MMX, SSE, SSE2, and 3DNow instructions.

b) QEMU

QEMU (Quick Emulator) is a fast processor emulator that uses a portable dynamic translator. It supports two operating modes: user space only, and full system emulation. In the earlier mode, QEMU can launch Linux processes compiled for one CPU on another CPU, or for cross-compilation and cross-debugging. In the later mode, it can emulate a full system that includes a processor and several peripheral devices. It supports emulation of a number of processor architectures that includes x86, ARM, PowerPC, and Sparc.

c) Crusoe

The Crusoe processor comes with a dynamic x86 emulator, called code morphing engine that can execute any x 86 based application on top of it. The Crusoe is designed to handle the x86 ISA's precise exception semantics without constraining speculative scheduling. This is accomplished by shadowing all registers holding the x86 state.

d) BIRD

BIRD is an interpretation engine for x86 binaries that currently supports only x86 as the host ISA and aims to extend for other architectures as well. It exploits the similarity between the architectures and tries to execute as many instructions as possible on the native hardware. All other instructions are supported through software emulation.

2) Hardware Abstraction Layer

- Virtualization at the Hardware Abstraction Layer (HAL) exploits the similarity in architectures of the guest and host platforms to cut down the interpretation latency. The time spent in instruction interpretation of guest platform to host platform is reduced by taking the similarities exist between them Virtualization technique helps map the virtual resources to physical resources and use the native hardware for computations in the virtual machine. This approach generates a virtual hardware environment which virtualizes the computer resources like CPU, Memory and IO devices.

- For the successful working of HAL the VM must be able to trap every privileged instruction execution and pass it to the underlying VMM, because multiple VMs running own OS might issue privileged instructions need full attention of CPU's .If it is not managed properly then VM may issues trap rather than generating an exception that makes crashing of instruction is sent to the VMM. However, the most popular platform, x86, is not fully-virtualizable, because it is been observed that certain privileged instructions fail silently rather than trapped when executed with insufficient privileges. Some of the popular HAL virtualization tools are

a) VMware

The VMware products are targeted towards x86-based workstations and servers. Thus, it has to deal with the complications that arise as x86 is not a fully-virtualizable architecture. The VMware deals with this problem by using a patent-pending technology that dynamically rewrites portions of the hosted machine code to insert traps wherever VMM intervention is required. Although it solves the problem, it adds some overhead due to the translation and execution costs. VMware tries to reduce the cost by caching the results and reusing them wherever possible. Nevertheless, it again adds some caching cost that is hard to avoid.

b) Virtual PC

The Microsoft Virtual PC is based on the Virtual Machine Monitor (VMM) architecture that lets user to create and configure one or more virtual machines. It provides most of the functions same as VMware but additional functions include undo disk operation that lets the user easily undo some previous operations on the hard disks of a VM. This enables easy data recovery and might come handy in several circumstances.

c) Denali

The Denali project was developed at University of Washington's to address this issue related to scalability of VMs. They come up with a new virtualization architecture also called Para virtualization to support thousands of simultaneous machines, which they call Lightweight Virtual Machines. It tries to increase the scalability and performance of the Virtual Machines without too much of implementation complexity.

3) Operating System Level Virtualization

- The operating system level virtualization is an abstraction layer between OS and user applications. It supports multiple Operating Systems and applications to be run simultaneously without required to reboot or dual boot. The degree of isolation of each OS is very high and can be implemented at low risk with easy maintenance. The implementation of operating system level virtualization includes, operating system installation, application suites installation, network setup, and so on. Therefore, if the required OS is same as the one on the physical machine then the user basically ends up with duplication of most of the efforts, he/she has already invested in setting up the physical machine. To run applications properly the operating system keeps the application specific data structure, user level libraries, environmental settings and other requisites separately.
- The key idea behind all the OS-level virtualization techniques is virtualization layer above the OS produces a partition per virtual machine on demand that is a replica of the operating environment on the physical machine. With a careful partitioning and multiplexing technique, each VM can be able to export a full operating environment and fairly isolated from one another and from the underlying physical machine.
- The popular OS level virtualization tools are

a) Jail

The Jail is a FreeBSD based virtualization software that provides the ability to partition an operating system environment, while maintaining the simplicity of UNIX ”root” model. The environments captured within a jail are typical system resources and data structures such as processes, file system, network resources, etc. A process in a partition is referred to as “in jail” process. When the system is booted up after a fresh install, no processes will be in jail. When a process is placed in a jail, all of its descendants after the jail creation, along with itself, remain within the jail. A process may not belong to more than one jail. Jails are created by a privileged process when it invokes a special system call jail. Every call to jail creates a new jail; the only way for a new process to enter the jail is by inheriting access to the jail from another process that already in that jail.

b) Ensim

The Ensim virtualizes a server's native operating system so that it can be partitioned into isolated computing environments called virtual private servers. These virtual private servers operate independently of each other, just like a dedicated server. It is commonly used in creating hosting environment to allocate hardware resources among large number of distributed users.

4) Library Level Virtualization

Most of the system uses extensive set of Application Programmer Interfaces (APIs) instead of legacy System calls to implement various libraries at user level. Such APIs are designed to hide the operating system related details to keep it simpler for normal programmers. In this technique, the virtual environment is created above OS layer and is mostly used to implement different Application Binary Interface (ABI) and Application Programming Interface (API) using the underlying system.

The example of Library Level Virtualization is WINE. The Wine is an implementation of the Windows API, and can be used as a library to port Windows applications to UNIX. It is a virtualization layer on top of X and UNIX to export the Windows API / ABI which allows to run Windows binaries on top of it.

5) Application Level Virtualization

In this abstraction technique the operating systems and user-level programs executes like applications for the machine. Therefore, specialize instructions are needed for hardware manipulations like I / O mapped (manipulating the I / O) and Memory mapped (that is mapping a chunk of memory to the I/O and then manipulating the memory). The group of such special instructions constitutes the application called Application level Virtualization. The Java Virtual Machine (JVM) is the popular example of application level virtualization which allows creating a virtual machine at the application-level than OS level. It supports a new self-defined set of instructions called java byte codes for JVM.

Such VMs pose little security threat to the system while letting the user to play with it like physical machines. Like physical machine it has to provide an operating environment to its applications either by hosting a commercial operating system, or by coming up with its own environment.

The comparison between different levels of virtualization is shown in Table 2.4.1.

Implementation Level	Performance	Application Flexibility	Implementation Complexity	Application Isolation
Instruction Set Architecture Level (ISA)	Very Poor	Very Good	Medium	Medium
Hardware Abstraction Level (HAL)	Very Good	Medium	Very Good	Good
Operating System Level	Very Good	Poor	Medium	Poor
Library Level	Medium	Poor	Poor	Poor
Application Level	Poor	Poor	Very Good	Very Good

Table 2.4.1 Comparison between different implementation levels of virtualization

4. What are different Mechanisms of Virtualizations?

Virtualization Mechanisms

Every hypervisor uses some mechanisms to control and manage virtualization strategies that allow different operating systems such as Linux and Windows to be run on the same physical machine, simultaneously. Depending on the position of the

virtualization layer, there are several classes of VM mechanisms, namely the binary translation, para-virtualization, full virtualization, hardware assist virtualization and host-based virtualization. The mechanisms of virtualization defined by VMware and other virtualization providers are explained as follows.

Binary Translation with Full Virtualization:

Based on the implementation technologies, hardware virtualization can be characterized into two types namely full virtualization with binary translation and host- based virtualization. The binary translation mechanisms with full and host-based virtualization are explained as follows.

a) Binary translation

In Binary translation of guest OS, The VMM runs at Ring 0 and the guest OS at Ring 1. The VMM checks the instruction stream and identifies the privileged, control and behavior-sensitive instructions. At the point when these instructions are identified, they are trapped into the VMM, which emulates the behavior of these instructions. The method used in this emulation is called binary translation. The binary translation mechanism is shown in Fig. 2.5.3.

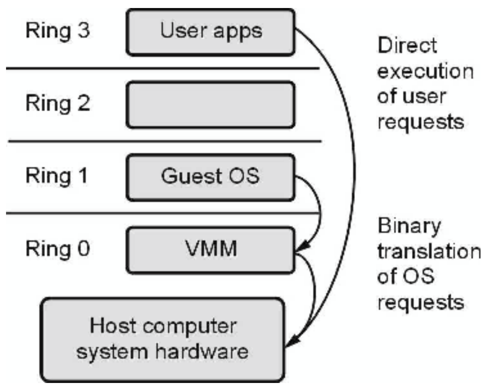

Ring 3
User apps
Ring 2
Ring 1
Guest OS
Ring 0
VMM
Host computer system hardware
Direct execution of user requests
Binary translation of OS requests

Fig. 2.5.3 Binary Translation mechanism

b) Full Virtualization

In full virtualization, host OS doesn't require any modification to its OS code. Instead it relies on binary translation to virtualize the execution of some sensitive, non-virtualizable instructions or execute trap. Most of the guest operating systems and their applications composed of critical and noncritical instructions. These instructions are executed with the help of binary translation mechanism.

With full virtualization, noncritical instructions run on the hardware directly while critical instructions are discovered and replaced with traps into the VMM to be emulated by software. In a host- based virtualization, both host OS and guest OS takes part in virtualization where virtualization software layer lies between them.

Therefore, full virtualization works with binary translation to perform direct execution of instructions where guest OS is completely decoupled from the underlying hardware and consequently, it is unaware that it is being virtualized.

The full virtualization gives degraded performance, because it involves binary translation of instructions first rather than executing which is rather time-consuming. Specifically, the full virtualization of I/O intensive applications is a really a big challenge as Binary translation employs a code cache to store translated instructions to improve performance, however it expands the cost of memory usage.

c) Host-based virtualization

In host-based virtualization, the virtualization layer runs on top of the host OS and guest OS runs over the virtualization layer. Therefore, host OS is responsible for managing the hardware and control the instructions executed by guest OS.

The host- based virtualization doesn't require to modify the code in host OS but virtualization software has to rely on the host OS to provide device drivers and other low-level services. This architecture simplifies the VM design with ease of deployment but gives degraded performance compared to other hypervisor architectures because of host OS interventions.

The host OS performs four layers of mapping during any IO request by guest OS or VMM which downgrades performance significantly.

Para-Virtualization

The para-virtualization is one of the efficient virtualization techniques that require explicit modification to the guest operating systems. The APIs are required for OS modifications in user applications which are provided by para-virtualized VM.

In some of the virtualized system, performance degradation becomes the critical issue. Therefore, para-virtualization attempts to reduce the virtualization overhead, and thus improve performance by modifying only the guest OS kernel. The para-virtualization architecture is shown in Fig. 2.5.4.

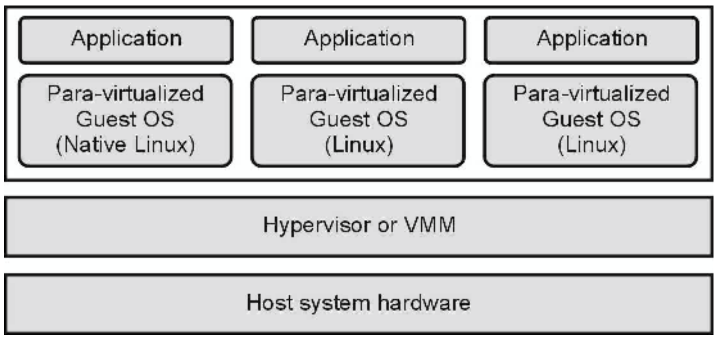

Application
Application
Application
Para-virtualized Guest OS (Native Linux)
Para-virtualized Guest OS (Linux)
Para-virtualized Guest OS (Linux)
Hypervisor or VMM
Host system hardware

Fig. 2.5.4 Para-virtualization architecture

The x86 processor uses four instruction execution rings namely Ring 0, 1, 2, and 3. The ring 0 has higher privilege of instruction being executed while Ring 3 has lower privilege. The OS is responsible for managing the hardware and the privileged instructions to execute at Ring 0, while user-level applications run at Ring 3. The KVM hypervisor is the best example of para-virtualization. The functioning of para-virtualization is shown in Fig. 2.5.5.

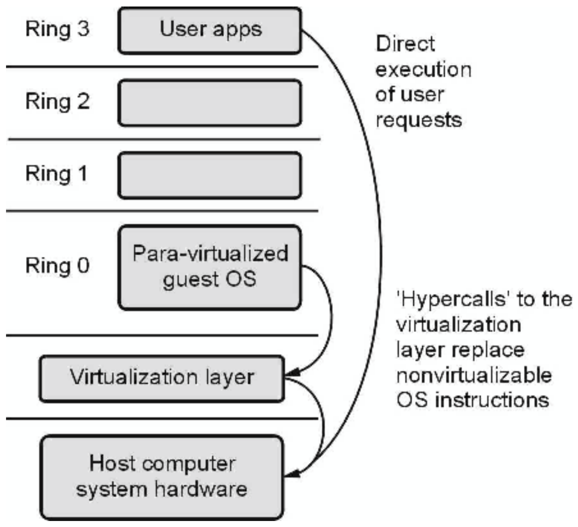

Ring 3
User apps
Ring 2
Ring 1
Ring 0
Para-virtualized guest OS
Virtualization layer
Host computer system hardware
Direct execution of user requests
'Hypercalls' to the virtualization layer replace nonvirtualizable OS instructions

Fig. 2.5.5 Para-virtualization (Source : VMware)

In para-virtualization, virtualization layer is inserted between the hardware and the OS. As x86 processor requires virtualization layer should be installed at Ring 0, the other instructions at Ring 0 may cause some problems. In this architecture, the nonvirtualizable instructions are replaced with hypercalls that communicate directly with the hypervisor or VMM. The user applications directly get executed upon user request on host system hardware.

Some disadvantages of para-virtualization are although para-virtualization reduces CPU overhead, but still has many issues with compatibility and portability of virtual system, it incurs high cost for implementation and maintenance and performance of virtualization varies due to workload variation. The popular examples of para- virtualization are Xen, KVM, and VMware ESXi.

a) Para-Virtualization with Compiler Support

The para-virtualization supports privileged instructions to be executed at run time. As full virtualization architecture executes the sensitive privileged instructions by intercepting and emulating them at runtime, para-virtualization can handle such instructions at compile time. In Para-Virtualization with Compiler Support thee guest OS kernel is modified to replace the privileged and sensitive instructions with hypercalls to the hypervisor or VMM at compile time itself. The Xen hypervisor assumes such para-virtualization architecture.

Here, guest OS running in a guest domain may run at Ring 1 instead of at Ring 0 that's why guest OS may not be able to execute some privileged and sensitive instructions. Therefore, such privileged instructions are implemented by hypercalls to the hypervisor. So, after replacing the instructions with hypercalls, the modified guest OS emulates the behavior of the original guest OS.

<u>**Virtualization of CPU, Memory, And I/O Devices**</u>

5. **Explain in detail about Virtualization of CPU, Memory, And I/O Devices.(Nov/Dec 2021)**

Virtualization of CPU

The CPU Virtualization is related to range protection levels called rings in which code can execute. The Intel x86 architecture of CPU offers four levels of privileges known as Ring 0, 1, 2 and 3.

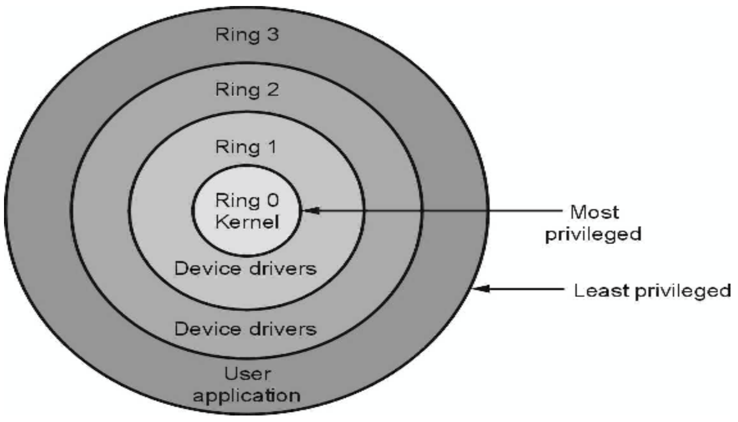


Fig. 2.6.1 CPU Privilege Rings

Among that Ring 0, Ring 1 and Ring 2 are associated with operating system while Ring 3 is reserved for applications to manage access to the computer hardware. As Ring 0 is used by kernel because of that Ring 0 has the highest-level privilege while Ring 3 has lowest privilege as it belongs to user level application shown in Fig. 2.6.1.

The user level applications typically run in Ring 3, the operating system needs to have direct access to the memory and hardware and must execute its privileged instructions in Ring 0. Therefore, Virtualizingx86 architecture requires placing a virtualization layer under the operating system to create and manage the virtual machines that delivers shared resources. Some of the sensitive instructions can't be virtualized as they have different semantics. If virtualization is not provided then there is a difficulty in trapping and translating those sensitive and privileged instructions at runtime which become the challenge. The x86 privilege level architecture without virtualization is shown in Fig. 2.6.2.

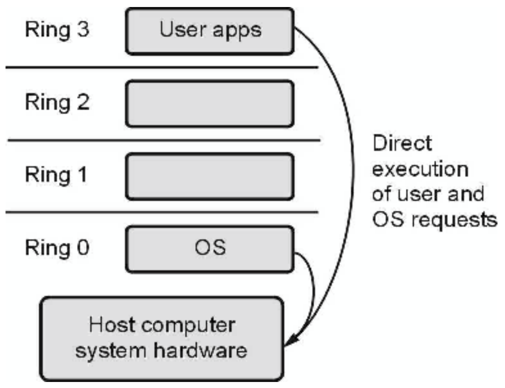

Ring 3
User apps
Ring 2
Ring 1
Ring 0
OS
Host computer system hardware
Direct execution of user and OS requests

Fig. 2.6.2 X86 privilege level architecture without virtualization

In most of the virtualization system, majority of the VM instructions are executed on the host processor in native mode. Hence, unprivileged instructions of VMs can run directly on the host machine for higher efficiency.

The privileged instructions are executed in a privileged mode and get trapped if executed outside this mode. The control-sensitive instructions allow to change the configuration of resources used during execution while Behavior-sensitive instructions uses different behaviors of CPU depending on the configuration of resources, including the load and store operations over the virtual memory.

Generally, the CPU architecture is virtualizable if and only if it provides ability to run the VM's privileged and unprivileged instructions in the CPU's user mode during which VMM runs in supervisor mode. When the privileged instructions along with control and behavior-sensitive instructions of a VM are executed, then they get trapped in the VMM. In such scenarios, the VMM becomes the unified mediator for hardware access from different VMs and guarantee the correctness and stability of the whole system. However, not all CPU architectures are virtualizable. There are three techniques can be used for handling sensitive and privileged instructions to virtualize the CPU on the x86 architecture :

1) Binary translation with full virtualization

2) OS assisted virtualization or para-virtualization

3) Hardware assisted virtualization

The above techniques are explained in detail as follows.

Binary translation with full virtualization

In binary translation, the virtual machine issues privileged instructions contained within their compile code. The VMM takes control on these instructions and changes the code under execution to avoid the impact on state of the system. The full virtualization technique does not need to modify host operating system. It relies on binary translation to trap and virtualize the execution of certain instructions.

The noncritical instructions directly run on the hardware while critical instructions have to be discovered first then they are replaced with

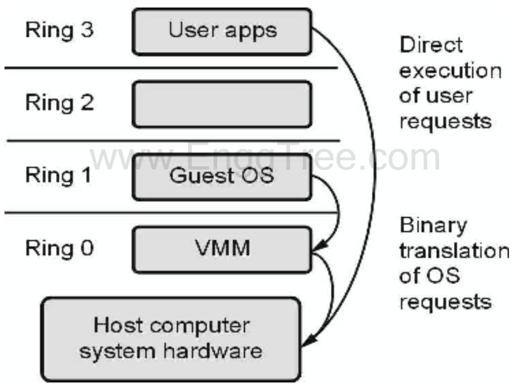

Ring 3
User apps
Ring 2
Ring 1
Guest OS
Ring 0
VMM
Host computer system hardware
Direct execution of user requests
Binary translation of OS requests

Fig. 2.6.3 Binary Translation with Full Virtualization

trap in to VMM to be emulated by software. This combination of binary translation and direct execution provides full virtualization as the guest OS is completely decoupled from the underlying hardware by the virtualization layer. The guest OS is not aware that it is being virtualized and requires no modification. The performance of full virtualization may not be ideal because it involves binary translation at run-time which is time consuming and can incur a large performance overhead. Full virtualization offers the best isolation and security for virtual machines, and simplifies migration and portability as the same guest OS instance can run virtualized or on native hardware. The full virtualization is only supported by VMware and Microsoft's hypervisors. The binary translation with full virtualization is shown in Fig. 2.6.3.

2) OS assisted virtualization or para-virtualization

The para-virtualization technique refers to making communication between guest OS and the hypervisor to improve the performance and efficiency. The para-virtualization involves modification to the OS kernel that replaces the non-virtualized instructions with hypercalls and can communicate directly with the virtualization or layer hypervisor. A hypercall is based on the same concept as a system call. The call made by hypervisor to the hardware is called hypercall. In para-virtualization the hypervisor is responsible for providing hypercall interfaces for other critical kernel operations such as memory management, interrupt handling and time keeping.

Fig. 2.6.4 shows para-virtualization.

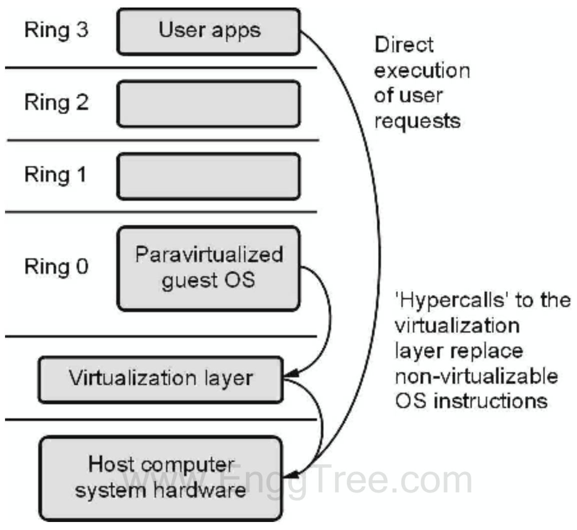

Ring 3
User apps
Ring 2
Ring 1
Ring 0
Paravirtualized guest OS
Virtualization layer
Host computer system hardware
Direct execution of user requests
'Hypercalls' to the virtualization layer replace non-virtualizable OS instructions

Fig. 2.6.4 Para-virtualization

3) Hardware Assisted Virtualization (HVM)

This technique attempts to simplify virtualization because full or para-virtualization is complicated in nature. The Processor makers like Intel and AMD provides their own proprietary CPU Virtualization Technologies called Intel VT-x and AMD-V. Intel and AMD CPUs add an additional mode called privilege mode level to x86 processors. All the privileged and sensitive instructions are trapped in the hypervisor automatically. This technique removes the difficulty of implementing binary translation of full virtualization. It also lets the operating system run in VMs without modification. Both of them target privileged instructions with a new CPU execution mode feature that allows the VMM to run in a new root mode below ring 0, also referred to as Ring 0P (for privileged root mode) while the Guest OS runs in Ring 0D (for de-privileged non-root mode). The

Privileged and sensitive calls are set automatically to trap the hypervisor running on hardware that removes the need for either binary translation or para-virtualization. The Fig. 2.6.5 shows Hardware Assisted Virtualization.

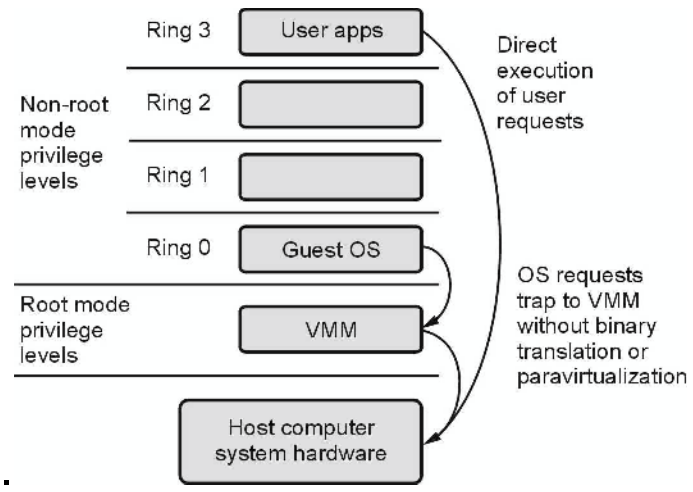

Ring 3
User apps
Direct execution of user requests
Non-root mode privilege levels
Ring 2
Ring 1
Ring 0
Guest OS
Root mode privilege levels
VMM
OS requests trap to VMM without binary translation or paravirtualization
Host computer system hardware

Fig. 2.6.5 Hardware Assisted Virtualization

Virtualization Of Memory

6. Explain in detail bout virtualization of memory with an example.

Virtualization of Memory

The memory virtualization involves physical memory to be shared and dynamically allocated to virtual machines. In a traditional execution environment, the operating system is responsible for maintaining the mappings of virtual memory to machine memory using page tables. The page table is a single-stage mapping from virtual memory to machine memory. All recent x86 CPUs comprises built-in Memory Management Unit (MMU) and a Translation Lookaside Buffer (TLB) to improve the virtual memory performance. However, in a virtual execution environment, the mapping is required from virtual memory to physical memory and physical memory to machine memory; hence it requires two-stage mapping process.

The modern OS provides virtual memory support that is similar to memory virtualization. The Virtualized memory is seen by the applications as a contiguous address space which is not tied to the underlying physical memory in the system. The operating system is responsible for mappings the virtual page numbers to physical page numbers stored in page tables. To optimize the Virtual memory performance all modern x86 CPUs include a Memory Management Unit (MMU) and a Translation Lookaside Buffer (TLB). Therefore, to run multiple virtual machines with Guest OS on a single system, the MMU has to be virtualized shown in Fig. 2.7.1.

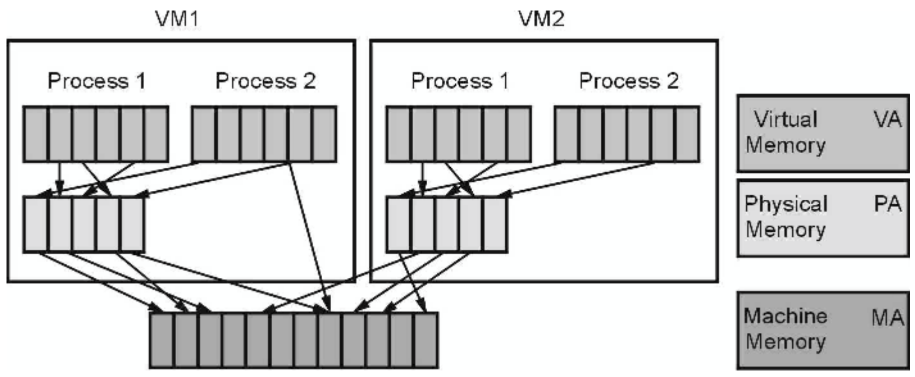


Fig. 2.7.1 Memory Virtualization

The Guest OS is responsible for controlling the mapping of virtual addresses to the guest memory physical addresses, but the Guest OS cannot have direct access to the actual machine memory. The VMM is responsible for mapping the Guest physical memory to the actual machine memory, and it uses shadow page tables to accelerate the mappings. The VMM uses TLB (Translation Lookaside Buffer) hardware to map the virtual memory directly to the machine memory to avoid the two levels of translation on every access. When the guest OS changes the virtual memory to physical memory mapping, the VMM updates the shadow page tables to enable a direct lookup. The hardware-assisted memory virtualization by AMD processor provides hardware assistance to the two-stage address translation in a virtual execution environment by
using a technology called nested paging.

Virtualization of I/O Device:

The virtualization of devices and I/O's is bit difficult than CPU virtualization. It involves managing the routing of I/O requests between virtual devices and the shared physical hardware. The software based I/O virtualization and management techniques can be used for device and I/O virtualization to enables a rich set of features and simplified management. The network is the integral component of the system which enables communication between different VMs. The I/O virtualization provides virtual NICs and switches that create virtual networks between the virtual machines without the network traffic and consuming bandwidth on the physical network. The NIC teaming allows multiple physical NICS to be appearing as one and provides failover transparency for virtual machines. It allows virtual machines to be seamlessly relocated to different systems using VMware VMotion by keeping their existing MAC addresses. The key for effective I/O virtualization is to preserve the virtualization benefits with minimum CPU utilization. Fig. 2.7.2 shows device and I/O virtualization.

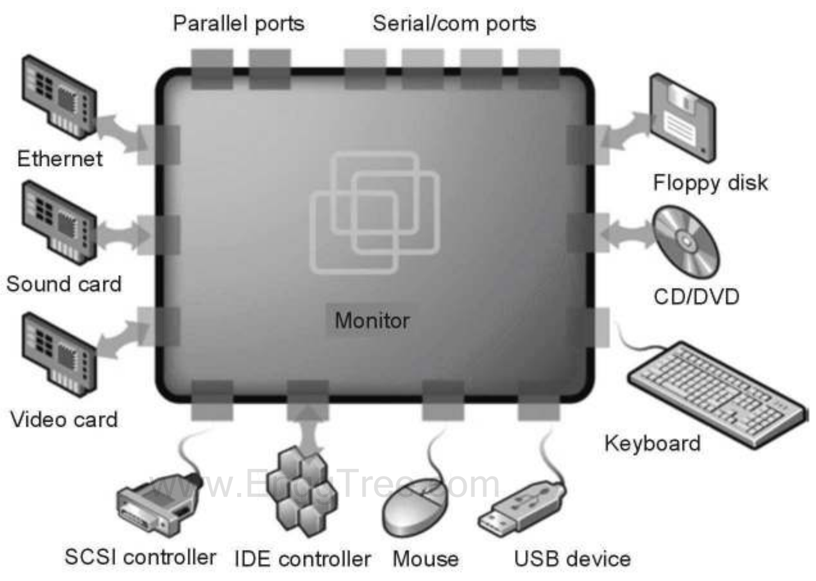


Fig. 2.7.2 Device and I/O virtualization

The virtual devices shown in above Fig. 2.7.2 can be effectively emulate on well-known hardware and can translate the virtual machine requests to the system hardware. The standardize device drivers help for virtual machine standardization. The portability in I/O Virtualization allows all the virtual machines across the platforms to be configured and run on the same virtual hardware regardless of their actual physical hardware in the system. There are three ways of implementing I/O virtualization. The full device emulation approach emulates well-known real-world devices where all the functions of device such as enumeration, identification, interrupt and DMA are replicated in software. The para-virtualization method of IO virtualization uses split driver model that consist of frontend and backend drivers. The front-end driver runs on Domain U which manages I/O request of guest OS. The backend driver runs Domain 0 which manages real I/O devices with multiplexing of I/O data of different VMs.

They interact with each other via block of shared memory. The direct I/O virtualization let the VM to access devices directly.it mainly focus on networking of mainframes. There are four methods to implement I/O virtualization namely full device emulation, para- virtualization, and direct I/O virtualization and through self-virtualized I/O.

In full device emulation, the IO devices are virtualized using emulation software. This method can emulate all well-known and real-world devices. The emulation software is responsible for performing all the functions of a devices or bus infrastructure, such as device enumeration, identification, interrupts, and DMA which are replicated. The software runs inside the VMM and acts as a virtual device. In this method, the I/O access
requests of the guest OS are trapped in the VMM which interacts with the I/O devices. The multiple VMs share a single hardware device for running them concurrently. However, software emulation consumes more time in IO access that's why it runs much slower than the hardware it emulates.

In para-virtualization method of I/O virtualization, the split driver model is used which consist of frontend driver and backend driver. It is used in Xen hypervisor with different drivers like Domain 0 and Domain U. The frontend driver runs in Domain U while backend driver runs in Domain 0. Both the drivers interact with each other via a block of shared memory. The frontend driver is responsible for managing the I/O requests of the guest OSes while backend driver is responsible for managing the real I/O devices and multiplexing the I/O data of different VMs.

The para-virtualization method of I/O virtualization achieves better device performance than full device emulation but with a higher CPU overhead.

In direct I/O virtualization, the virtual machines can access IO devices directly. It does not have to rely on any emulator of VMM. It has capability to give better IO performance without high CPU costs than para-virtualization method. It was designed for focusing on networking for mainframes.

In self-virtualized I/O method, the rich resources of a multicore processor and harnessed together. The self-virtualized I/O encapsulates all the tasks related with virtualizing an I/O device. The virtual devices with associated access API to VMs and a management API to the VMM are provided by self-virtualized I/O that defines one Virtual Interface (VIF) for every kind of virtualized I/O device.

The virtualized I/O interfaces are virtual network interfaces, virtual block devices (disk), virtual camera devices, and others. The guest OS interacts with the virtual interfaces via device drivers. Each VIF carries a unique ID for identifying it in self- virtualized I/O and consists of two message queues. One message queue for outgoing messages to the devices and another is for incoming messages from the devices.

As there are a many of challenges associated with commodity hardware devices, the multiple IO virtualization techniques need to be incorporated for eliminating those associated challenges like system crash during reassignment of IO devices, incorrect functioning of IO devices and high overhead of device emulation.

PART-A

1. **"Although virtualization is widely accepted today; it does have its limits". Comment on the statement. (May-2021)**

   Although virtualization is widely accepted today; it does have its limitations that are listed below.

   - High upfront Investments : Organisations need to acquire resources beforehand to implement Virtualization. Also, there might occur a need to incur additional resources with time.
   - Performance Issues : Although virtualization is an efficient technique and efficiency can be increased by applying some techniques, there may be chances when the efficiency is not as good as that of the actual physical systems.
   - Licensing Issues : All software may not be supported on virtual platforms. Although vendors are becoming aware of the increasing popularity of virtualization and have started providing licenses for software to run on these platforms, the problem has not completely vanished. Therefore, it is advised to check the licenses with the vendor before using the software.
   - Difficulty in Root Cause Analysis : With the addition of an additional layer in virtualization, complexity gets increased. This increased complexity makes root cause analysis difficult in case of unidentified problems.

2. **List the requirements of VMM.(Nov/Dec 2021)**

   The requirements of VMM or hypervisor are

   - VMM must support efficient task scheduling and resource allocation techniques.
   - VMM should provide an environment for programs which is essentially identical to the original physical machine.
   - A VMM should be in complete control of the system resources.

- Any program run under a VMM should exhibit a function identical to that which it runs on the original physical machine directly.
- VMM must be tightly related to the architectures of processors

3. Give the role of a VM. (or) Give the basic operations of a VM. (May-2017)

Virtualization allows running multiple operating systems on a single physical machine. Each instance of operating system running inside called Virtual machine (VM). The main role of VM is to allocate the host machine resources to run Operating system. The other roles of VM are

- Provide virtual hardware, including CPUs, memory, storage, hard drives, network interfaces and other devices to run virtual operating system.
- Provide fault and security isolation at the hardware level.
- Preserve performance with advanced resource controls.
- Save the entire state of a virtual machine to files.
- Move and copy virtual machines data as easily as like moving and copying files.
- Provision to migrate any virtual machine to any physical server.

4. Give the significance of virtualization. (Dec 2019)(May-2021)

As we know that the large amounts of compute, storage, and networking resources are needed to build a cluster, grid or cloud solution. These resources need to be aggregated at one place to offer a single system image. Therefore, the concept of virtualization comes into the picture where resources can be aggregated together to fulfill the request for resource provisioning with rapid speed as a single system image. The virtualization is a novel solution that can offer application inflexibility, software manageability, optimum resource utilization and security concerns in existing physical machines. In particular, every cloud solution has to rely on virtualization solution for provisioning the resources dynamically. Therefore, virtualization technology is one of the fundamental components of cloud computing. It provides secure, customizable, and isolated execution environment for running applications on abstracted hardware. It is mainly used for providing different computing environments. Although these computing environments are virtual but appear like to be physical. The different characteristics of virtualization are,

- Maximum resource utilization
- Minimize the maintenance cost
- Supports Server Consolidation
- Reduces Hardware Cost
- Supports Dynamic Load balancing
- Supports Disaster recovery

- Can run Legacy applications and can test Beta Softwares.

5. Define Virtualization. (May-2019)(May-2022)

The term Virtualization is nothing but creation of a virtual version of hardware platform, operating system, storage or network resources rather than actual. It allows to run multiple operating systems on a single physical machine called host machine. Each instance of operating system called Virtual Machine (VM) and operating system runs inside virtual machine is called guest operating system.

- **Enlist the pros and cons of virtualization ?**
  - Cost Reduction
  - Efficient resource utilization
  - Optimization
  - Budgeting
  - Increased Return on Investment
  - Increased Flexibility
  - Upfront Investments
  - Performance Issues
  - Licensing Issues
  - Difficulty in Root Cause Analysis

6. What is server virtualization ?

A server virtualization is the process of dividing a physical server into multiple unique and isolated virtual servers by means of software. It partitions a single physical server into the multiple virtual servers; each virtual server can run its own operating system and applications independently. The virtual server is also termed as virtual machine. The consolidation helps in running many virtual machines under a single physical server. The popular server virtualization softwares are VMware's vSphere, Citrix Xen Server, Microsoft's Hyper-V, and Red Hat's Enterprise Virtualization.

7. Enlist advantages and disadvantages of Bare-Metal structure.

The advantages of Bare-Metal structure are

- It is faster in performance and more efficient to use.
- It provides enterprise features like high scalability, disaster recovery and high availability.

- It has high processing power due to the resource pooling.
- It has lower overhead or maintenance cost.
- It provides ease of backup and recovery.
- It provides built-in fault-tolerance mechanisms.
- It has improved mobility and security.

The disadvantages of Bare-Metal structure are

- It has limited hardware support and poor stack of device drivers.
- It has high implementation cost
- It requires specialized servers to install and run hypervisor and do not run on user workstations.
- In some cases, it becomes complex for management.

8. What is Xen ?

Xen is an open source Bare-Metal (Type I) hypervisor developed by Cambridge University. It runs on the top of hardware without needing a host operating system. The absence of host OS eliminate the need for pass through permission by the hypervisor. Xen is a microkernel hypervisor, which separates the policy from the mechanism. It provides a virtual environment located between the hardware and the OS. As Xen hypervisor runs directly on the hardware devices, it runs many guest operating systems on the top of it. The various operating system platforms supported as a guest OS by Xen hypervisor are Windows, Linux, BSD and Solaris.

9. Differentiate full Virtualization and Para-Virtualization.(Nov-2020)

S.No.	Full Virtualization	Para virtualization
1	In Full virtualization, virtual machines permit the execution of the instructions with the running of unmodified OS in an entirely isolated way.	In para virtualization, a virtual machine does not implement full isolation of OS but rather provides a different API which is utilized when OS is subjected to alteration.
2	Full Virtualization is less secure.	While the Para virtualization is more secure than the Full Virtualization.
3	Full Virtualization uses binary translation and a direct approach as a technique for operations.	While Para virtualization uses hyper calls at compile time for operations.

4	Full Virtualization is slow than para virtualization in operation.	Para virtualization is faster in operation as compared to full virtualization.
5	Full Virtualization is more portable and compatible.	Para virtualization is less portable and compatible.
6	Examples of full virtualization are Microsoft and Parallels systems.	Examples of para virtualization are Microsoft Hyper-V, Citrix Xen, etc.
7	It supports all guest operating systems without modification.	The guest operating system has to be modified and only a few operating systems support it.
8	The guest operating system will issue hardware calls.	Using the drivers, the guest operating system will directly communicate with the hypervisor.
9	It is less streamlined compared to para-virtualization.	It is more streamlined.
10	It provides the best isolation.	It provides less isolation compared to full virtualization.

11.Distinguish between Virtual Machine and Containers.

S. No.	Virtual Machine (VM)	Containers
1	The hardware is virtualized to execute several Operating system instances with VMs.	Containers facilitate a way for virtualizing the operating system so that several workloads can execute on an individual operating system instance
2	VM is managed via hypervisor and uses VM hardware.	Containers give services of OS from an underlying host and also separate the applications utilizing virtual-memory hardware.
3	VM facilitates the abstract machine which utilizes device drivers addressing an abstract machine.	Container facilitates the abstract operating system.

4	VM technologies are well-known within various embedded communities.	The container has been grown on several clouds and servers with organizations like Google and Facebook. For example, all services of Google Docs get a container/instance.
5	Higher overhead	Lower overhead
6	VM permits us for installing other software so virtually we control it as disputed to install the software on a computer directly.	The containers are software that permits distinct application's functionalities independently.
7	Applications executing on virtual machine system can execute distinct OS.	Applications executing within the container environment contribute to an individual OS.
8	VM facilitates a way for virtualizing any computer system.	Container only virtualizes the OS.
9	VMs have a large size.	Containers are very light (some megabytes).
10	VM runs in minutes due to its large size.	Containers run in seconds.
11	It utilizes a lot of memory of the system.	Containers utilize very less system memory.
12	It is highly secured.	It is less secure.
13	VM is helpful if we need each resource of OS to execute several applications.	A container is helpful if we needed to maximize various executing applications with minimal servers.
14	VM examples: **VMware, Xen, KVM**	Container examples: **Containers via Docker, PhotonOS, RancherOS**.

UNIT III

VIRTUALIZATION INFRASTRUCTURE AND DOCKER

Desktop Virtualization – Network Virtualization – Storage Virtualization – System-level of Operating Virtualization – Application Virtualization – Virtual clusters and Resource Management – Containers vs. Virtual Machines – Introduction to Docker – Docker Components – Docker Container – Docker Images and Repositories.

Types of Virtualization

1. **Virtualization ranging from hardware to applications in five abstraction levels.**

Based on the functionality of virtualized applications, there are five basic types of virtualization which are explained as follows.

Desktop Virtualization

The processing of multiple virtual desktops occurs on one or a few physical servers, typically at the centralized data center. The copy of the OS and applications that each end user utilizes will typically be cached in memory as one image on the physical server.

The Desktop virtualization provides a virtual desktop environment where client can access the system resources remotely through the network.

The ultimate goal of desktop virtualization is to make computer operating system accessible from anywhere over the network. The virtual desktop environments do not require a specific system or hardware resources on the client side; however, it requires just a network connection.

The user can utilize the customized and personalized desktop from a remote area through the network connection. The virtualization of the desktop is sometimes referred as Virtual Desktop Infrastructure (VDI) where all the operating systems like windows, or Linux are installed as a virtual machine on a physical server at one place and deliver them remotely through the Remote Desktop Protocols like RDP (in windows) or VNC (in Linux).

The processing of multiple virtual desktops occurs on one or more physical servers placed commonly at the centralized data center. The copy of the OS and applications that each end client uses will commonly be stored in memory as one image the physical server.

Currently, VMware Horizon and Citrix Xen Desktop are the two most popular VDI solutions available in the market with so many dominating features. Although, Desktop operating system provided by VDI is virtual but appears like a physical desktop operating system. The virtual desktop can run all the types of applications that are supported on physical computer but only difference is they are delivered through the network.

Some of the benefits provided by Desktop virtualization are :

- It provides easier management of devices and operating systems due to centralized management.
- It reduces capital expenditure and maintenance cost of hardware due to consolidation of multiple operating systems into a single physical server,
- It provides enhance security as confidential data is stored in data center instead of personal devices that could easily be lost, stolen or tampered with.
- With Desktop virtualization, operating systems can be quickly and easily provisioned for the new users without doing any manual setup.
- Upgradation of operating system is easier
- It can facilitate Work from Home feature for IT Employees due to the desktop operating system delivery over the internet.

Network Virtualization

The Network virtualization is the ability to create virtual networks that are decoupled from the underlying network hardware. This ensures the network can better integrate with and support increasingly virtual environments. It has capability to combine multiple physical networks into one virtual, or it can divide one physical network into separate, independent virtual networks.

The Network virtualization is the ability to make virtual networks that are decoupled from the underlying network hardware. This ensures the network can better integrate with and support increasingly virtual environments. It has capacity to combine multiple physical networks into single virtual, or it can divide one physical network into separate, independent virtual networks.

The Network virtualization can combine the entire network into a single mode and allocates its bandwidth, channels, and other resources based on its workload.

Network virtualization is similar to server virtualization but instead of dividing up a physical server among several virtual machines, physical network resources are divided up among multiple virtual networks.

Network virtualization uses specialized software to perform network functionality by decoupling the virtual networks from the underlying network hardware. Once network virtualization is established, the physical network is only used for packet forwarding and network management is done using the virtual or software- based switches.

The VMware's NSX platform is the popular example of network virtualization which decouples network services from the underlying hardware and allows virtual provisioning of an entire network. The physical network resources, such as switches and routers, are pooled and accessible by any user via a centralized management system. The benefits of network virtualization are

- It consolidates the physical hardware of a network into a single virtual network that reduce the management overhead of network resources.
- It gives better scalability and flexibility in network operations.
- It provides automated provisioning and management of network resources.
- It reduces the hardware requirements and will have a corresponding impact on your power consumption.
- It is cost effective as it requires reduced the number of physical devices.

Storage Virtualization

Storage virtualization is the process of grouping multiple physical storages using software to appear as a single storage device in a virtual form.

It pools the physical storage from different network storage devices and makes it appear to be a single storage unit that is handled from a single console. Storage virtualization helps to address the storage and data management issues by facilitating easy backup, archiving and recovery tasks in less time.

It aggregates the functions and hides the actual complexity of the storage area network. The storage virtualization can be implemented with data storage technologies like snapshots and RAID that take physical disks and present them in a virtual format. These features can allow to perform redundancy to the storage and gives optimum performance by presenting host as a volume.

Virtualizing storage separates the storage management software from the underlying hardware infrastructure in order to provide more flexibility and scalable pools of storage resources. The benefits provided by storage virtualization are

- Automated management of storage mediums with estimated of down time.

- Enhanced storage management in heterogeneous IT environment.
- Better storage availability and optimum storage utilization.
- It gives scalability and redundancy in storage.
- It provides consummate features like disaster recovery, high availability, consistency, replication & re-duplication of data.
- The backup and recovery are very easier and efficient in storage virtualization.

Server Virtualization

A Server virtualization is the process of dividing a physical server into multiple unique and isolated virtual servers by means of software. It partitions a single physical server into the multiple virtual servers; each virtual server can run its own operating system and applications independently. The virtual server is also termed as virtual machine. The consolidation helps in running many virtual machines under a single physical server.

Each virtual machine shares the hardware resources from physical server that leads to better utilization of the physical servers' resources. The resources utilized by virtual machine include CPU, memory, storage, and networking. The hypervisor is the operating system or software that runs on the physical machine to perform server virtualization.

The hypervisor running on physical server is responsible for providing the resources to the virtual machines. Each virtual machine runs independently of the other virtual machines on the same box with different operating systems that are isolated from each other.

The popular server virtualization softwares are VMware's vSphere, Citrix Xen Server, Microsoft's Hyper-V, and Red Hat's Enterprise Virtualization.

The benefits of server virtualization are

- It gives quick deployment and provisioning of virtual operating system.
- It has reduced the capital expenditure due to consolidation of multiple servers into a single physical server which eliminate the cost of multiple physical hardware.
- It provides ease in development & testing.
- It makes optimum resource utilization of physical server.
- It provides centralize the server administration and disaster recovery feature.
- It reduces cost because less hardware is required.

Application Virtualization

Application virtualization is a technology that encapsulates an application from the underlying operating system on which it is executed. It enables access to an application without needing to install it on the local or target device. From the user's perspective, the application works and interacts like it's native on the device.

It allows to use any cloud client which supports BYOD like Thin client, Thick client, Mobile client, PDA and so on.

Application virtualization utilizes software to bundle an application into an executable and run anywhere type of application. The software application is isolated from the operating system and runs in an environment called as "sandbox".

There are two types of application virtualization: remote and streaming of the application. In first type, the remote application will run on a server, and the client utilizes some kind of remote display protocol to communicate back. For large number of administrators and users, it's fairly simple to set up remote display protocol for applications.

In second type, the streaming application will run one copy of the application on the server, and afterward have client desktops access and run the streaming application locally. With streaming application, the upgrade process is simpler, since you simply set up another streaming application with the upgrade version and have the end users point to the new form of the application.

Some of the popular application virtualization softwares in the commercial center are VMware ThinApp, Citrix XenApp, Novell ZENworks Application Virtualization and so on.

Some of the prominent benefits of application virtualization are

- It allows for cross-platform operations like running Windows applications on Linux or android and vice versa.
- It allows to run applications that have legacy issues like supported on older
- Operating systems.
- It avoids conflict between the other virtualized applications
- It allows a user to run more than one instance of an application at same time
- It reduces system integration and administration costs by maintaining a common software baseline across multiple diverse computers in an organization.
- It allows to run incompatible applications side by side, at the same time
- It utilizes less resource than a separate virtual machine.
- It provides greater security because of isolating environment between applications and

operating system.

VIRTUAL CLUSTERS AND RESOURCE MANAGEMENT

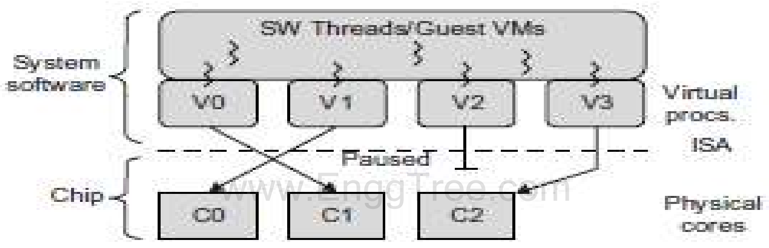
2. **Explain in detail about Virtual Clusters and Resource Management with an example.**

Physical versus Virtual Processor Cores

A multicore virtualization method is provides to allow hardware designers to get an abstraction of the low-level details of the processor cores.

This technique alleviates the burden and inefficiency of managing hardware resources by software.

It is located under the ISA and remains unmodified by the operating system or VMM (hypervisor). The below figure illustrates the technique of a software-visible VCPU moving from one core to another and temporarily suspending execution of a VCPU when there are no appropriate cores on which it can run.



Multicore virtualization

Virtual Hierarchy

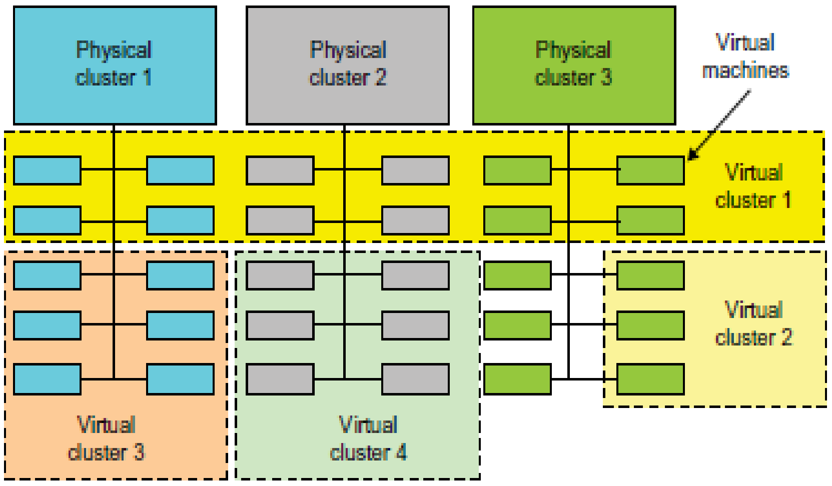
➢ The emerging many-core chip multiprocessors (CMPs) provide a new computing landscape. Instead of supporting time-sharing jobs on one or a few cores, abundant cores are used in a space-sharing, where single-threaded or multithreaded jobs are simultaneously assigned to separate groups of cores for long time intervals.

➢ To optimize for space-shared workloads, they propose using virtual hierarchies to overlay a coherence and caching hierarchy onto a physical processor. Unlike a fixed physical hierarchy, a virtual hierarchy can adapt to fit how the work is space shared for improved performance and performance isolation.

a) Physical clusters vs. virtual clusters

Virtual clusters are built with VMs installed at distributed servers from one or more physical clusters.The VMs in a virtual cluster are interconnected logically by a virtual network across several physical networks. The below Figure illustrates the concepts of virtual clusters and physical clusters.

Each virtual cluster is formed with physical machines or a VM hosted by multiple physical clusters. The virtual cluster boundaries are shown as distinct boundaries.



Virtual clusters and physical clusters

The Virtual clusters are advantageous than physical clusters by following properties.

Fast Deployment and Effective Scheduling:

The system should have the capability of fast deployment. Here, deployment means two things:

1. To construct and distribute software stacks (OS, libraries, applications) to a physical node inside clustersas fast as possible, and
2. To quickly switch runtime environments from one user's virtual cluster toanother user's virtual cluster. If one user finishes using his system, the corresponding virtual clustershould shut down or suspend quickly to save the resources to run other VMs for other users.

High-Performance Virtual Storage:

It is important to efficiently manage the disk spaces occupied by template software packages. Some storage architecture design can be applied to reduce duplicated blocks in a distributed file system of virtual clusters. Hash values are used to compare the contents of data blocks. Users have their own profiles which store the identification of the data blocks for corresponding VMs in a user-specific virtual cluster.Basically, there are four steps to deploy a group of VMs onto a target cluster:

- ✓ Preparing the diskimage,
- ✓ Configuring the VMs,
- ✓ Choosing the destination nodes, and
- ✓ Executing the VM deployment commandon every host

3. Explain about Virtualization for Linux and windows and NT Platform. Design the process of Live Migration of VM from one host to another.

b)Live VM Migration Steps and Performance Effects

Live migration of a VM consists of the following six steps:

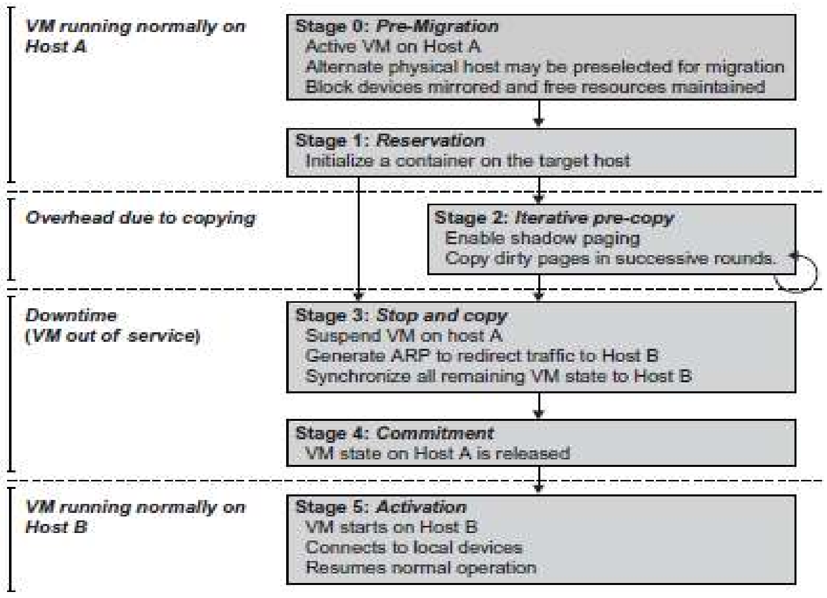
Steps 0 and 1: Start migration. This step makes preparations for the migration, including determining the migrating VM and the destination host. Although users could manually make a VM migrate to an appointed host, in most circumstances, the migration is automatically started by strategies such as load balancing and server consolidation.

Steps 2: Transfer memory. Since the whole execution state of the VM is stored in memory, sending the VM's memory to the destination node ensures continuity of the service provided by the VM. All of the memory data is transferred in the first round, and then the migration controller recopies the memory data which is changed in the last round. These steps keep iterating until the dirty portion of the memory is small enough to handle the final copy. Although precopying memory is performed iteratively, the execution of programs is not obviously interrupted.

Step 3: Suspend the VM and copy the last portion of the data. The migrating VM's execution is suspended when the last round's memory data is transferred. Other nonmemory data such as CPU and network states should be sent as well. During this step, the VM is stopped and its applications will no longer run. This "service unavailable" time is called the "downtime" of migration, which should be as short as possible so that it can be negligible to users.

Steps 4 and 5: Commit and activate the new host. After all the needed data is copied, on the destination host, the VM reloads the states and recovers the execution of programs in it, and the service provided by this VM continues. Then the network connection is redirected to the new VM and the dependency to the source host is cleared.

The whole migration process finishes by removing the original VM from the source host.

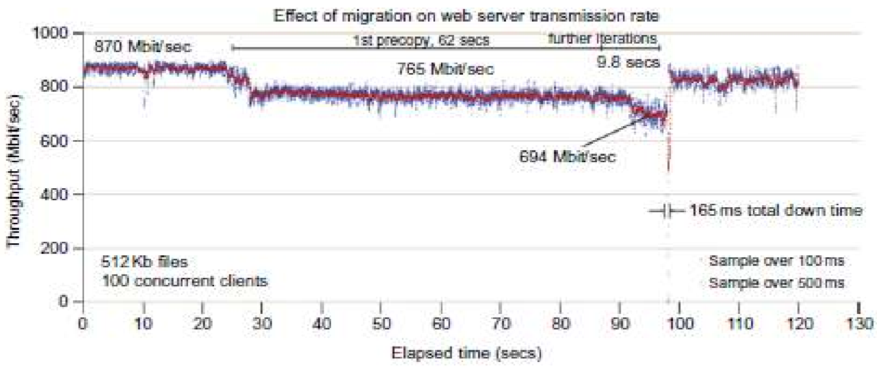

VM running normally on Host A
Stage 0: Pre-Migration
Active VM on Host A
Alternate physical host may be preselected for migration
Block devices mirrored and free resources maintained
Stage 1: Reservation
Initialize a container on the target host
Overhead due to copying
Stage 2: Iterative pre-copy
Enable shadow paging
Copy dirty pages in successive rounds.
Downtime (VM out of service)
Stage 3: Stop and copy
Suspend VM on host A
Generate ARP to redirect traffic to Host B
Synchronize all remaining VM state to Host B
Stage 4: Commitment
VM state on Host A is released
VM running normally on Host B
Stage 5: Activation
VM starts on Host B
Connects to local devices
Resumes normal operation

Live migration process of a VM from one host to another

Performance Effects:

The below diagram shows the effect on the data transmission rate (Mbit/second) of live migration of a VM from one host to another.Before copying the VM with 512 KB files for 100 clients, the data throughput was 870 MB/second.

The first precopy takes 63 seconds, during which the rate is reduced to 765 MB/second. Then the data rate reduces to 694 MB/second in 9.8 seconds for more iterations of the copying process. The system experiences only 165 ms of downtime, before the VM is restored at the destination host. This experimental result shows a very small migration overhead in live transfer of a VM between host nodes.



Effect on data transmission rate of a VM migrated from one failing web server to another.

c) Migration of Memory, Files, and Network Resources

File Migration:

- To support VM migration, a system must provide each VM with a consistent, location-independent view of the file system that is available on all hosts.
- A simple way to achieve this is to provide each VM with its own virtual disk, which the file system is mapped to, and transport the contents of this virtual disk along with the other states of the VM.
- However, due to the current trend of high capacity disks, migration of the contents of an entire disk over a network is not a viable solution.
- Another way is to have a global file system across all machines where a VM could be located. This way removes the need to copy files from one machine to another because all files are network accessible.
- A distributed file system is used in ISR serving as a transport mechanism for propagating a suspended VM state. The actual file systems themselves are not mapped onto the distributed file system.
- Instead, the VMM only accesses its local file system. The relevant VM files are explicitly copied into the local file system for a resume operation and taken out of the local file system for a suspend operation.

Memory Migration:

- Memory Migration is one of the most important aspects of VM migration. Moving the memory instance of a VM from one physical host to another can be approached in any number of ways.
- Memory migration can be in a range of hundreds of megabytes to a few gigabytes in a typical system today, and it needs to be done in an efficient manner.
- The Internet Suspend-Resume (ISR) technique exploits temporal locality, as memory states are likely to have considerable overlap in the suspended and the resumed instances of a VM.
- Temporal locality refers to the fact that the memory states differ only by the amount of work done since a VM was last suspended before being initiated for migration.

Network Migration: A migrating VM should maintain all open network connections without relying on forwarding mechanisms on the original host or on support from mobility or redirection mechanisms.

- To enable remote systems to locate and communicate with a VM, each VM must beassigned a virtual IP address known to other entities.

- This address can be distinct from the IP address of the host machine where the VM is currently located. Each VM can also have its own distinct virtual MAC address.
- The VMM maintains a mapping of the virtual IP and MAC addresses to their corresponding VMs. In general, a migrating VM includes all the protocol states and carries its IP address with it.

d. Dynamic Deployment of Virtual Clusters

The Cellular Disco at Stanford is a virtual cluster built in a shared-memory multiprocessor system. The INRIA virtual cluster was built to test parallel algorithm performance. The COD (Cluster-on-Demand) project is a virtual cluster management system for dynamic allocation of servers from a computing pool to multiple virtual clusters.

The idea is illustrated by the prototype implementation of the COD shown in figure. The COD partitions a physical cluster into multiple virtual clusters (vClusters). vCluster owners specify the operating systems and software for their clusters through an XML-RPC interface. The vClusters run a batch schedule from Sun's GridEngine on a web server cluster. The COD system can respond to load changes in restructuringthe virtual clusters dynamically.

3. Explain in detail about Virtual Storage Management with an example

The term "storage virtualization" was widely used before the renaissance of system virtualization. Yet the term has a different meaning in a system virtualization environment. Previously, storage virtualization was largely used to describe the aggregation and repartitioning of disks at very coarse time scales for use by physical machines. In system virtualization, virtual storage includes the storage managed by VMMs and guest OSes. Generally, the data stored in this environment can be classified into two categories:

i. VM images and
ii. Application Data.

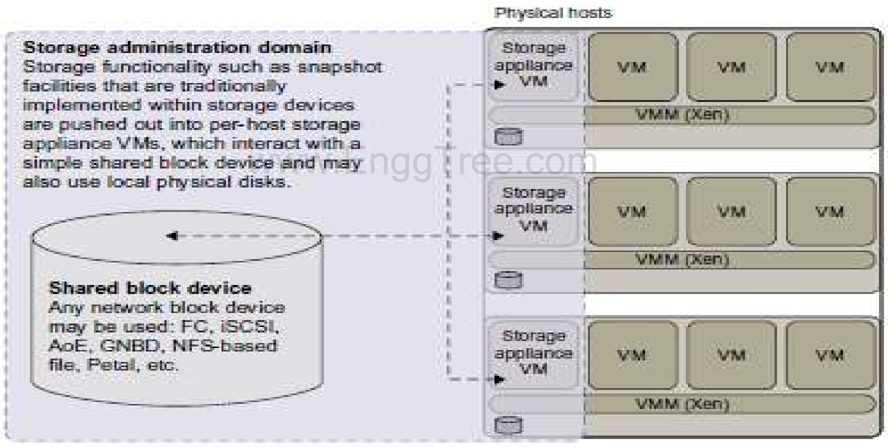
The VM images are special to the virtual environment, while application data includes all other data which is the same as the data in traditional OS environments. The most important aspects of system virtualization are encapsulation and isolation.The following are the major functions provided by Virtual Storage Management:

- In virtualization environments, a virtualization layer is inserted between the hardware and traditional operating systems or a traditional operating system is modified to support virtualization.

- This procedure complicates storage operations. On the one hand, storage management of the guest OS performs as though it is operating in a real hard disk while the guest OSes cannot access the hard disk directly.
- On the other hand, many guest OSes contest the hard disk when many VMs are running on a single physical machine. Therefore, storage management of the underlying VMM is much more complex than that of guest OSes (traditional OSes).

Example:Parallax Providing Virtual Disks to Client VMs from a Large CommonShared Physical Disk.

The architecture of Parallax is scalable and especially suitable for use in cluster-based environments.The below figure shows a high-level view of the structure of a Parallax-based cluster. A cluster-wide administrative domain manages all storage appliance VMs, which makes storage management easy.This mechanism enables advanced storage features such as snapshot facilities to be implemented in software and delivered above commodity network storage targets.


Physical hosts
Storage administration domain
Storage functionality such as snapshot facilities that are traditionally implemented within storage devices are pushed out into per-host storage appliance VMs, which interact with a simple shared block device and may also use local physical disks.
Shared block device
Any network block device may be used: FC, iSCSI, AoE, GNBD, NFS-based file, Petal, etc.
Storage appliance VM
VM
VM
VM
VMM (Xen)
Storage appliance VM
VM
VM
VM
VMM (Xen)
Storage appliance VM
VM
VM
VM
VMM (Xen)

High-level view of the structure of a Parallax-based cluster

<u>Dockers:</u>

4. **Explain in detail about Dockers and its types of components with an example.**

Introduction to Dockers

What is Docker?

Docker is an open-source centralized platform designed to create, deploy, and run applications. Docker uses container on the host's operating system to run applications. It allows applications to use the same Linux kernel as a system on the host computer, rather than creating a whole virtual operating system. Containers ensure that our application works in any environment like development, test, or production.

Docker includes components such as Docker client, Docker server, Docker machine, Docker hub, Docker composes, etc.

Why Docker?

Docker is designed to benefit both the Developer and System Administrator. There are the following reasons to use Docker -

- Docker allows us to easily install and run software without worrying about setup or dependencies.
- Developers use Docker to eliminate machine problems, i.e. "**but code is worked on my laptop**." when working on code together with co-workers.
- Operators use Docker to run and manage apps in isolated containers for better compute density.
- Enterprises use Docker to securely built agile software delivery pipelines to ship new application features faster and more securely.
- Since docker is not only used for the deployment, but it is also a great platform for development, that's why we can efficiently increase our customer's satisfaction.

Advantages of Docker

There are the following advantages of Docker -

- It runs the container in seconds instead of minutes.
- It uses less memory.
- It provides lightweight virtualization.
- It does not a require full operating system to run applications.
- It uses application dependencies to reduce the risk.
- Docker allows you to use a remote repository to share your container with others.
- It provides continuous deployment and testing environment.

Disadvantages of Docker

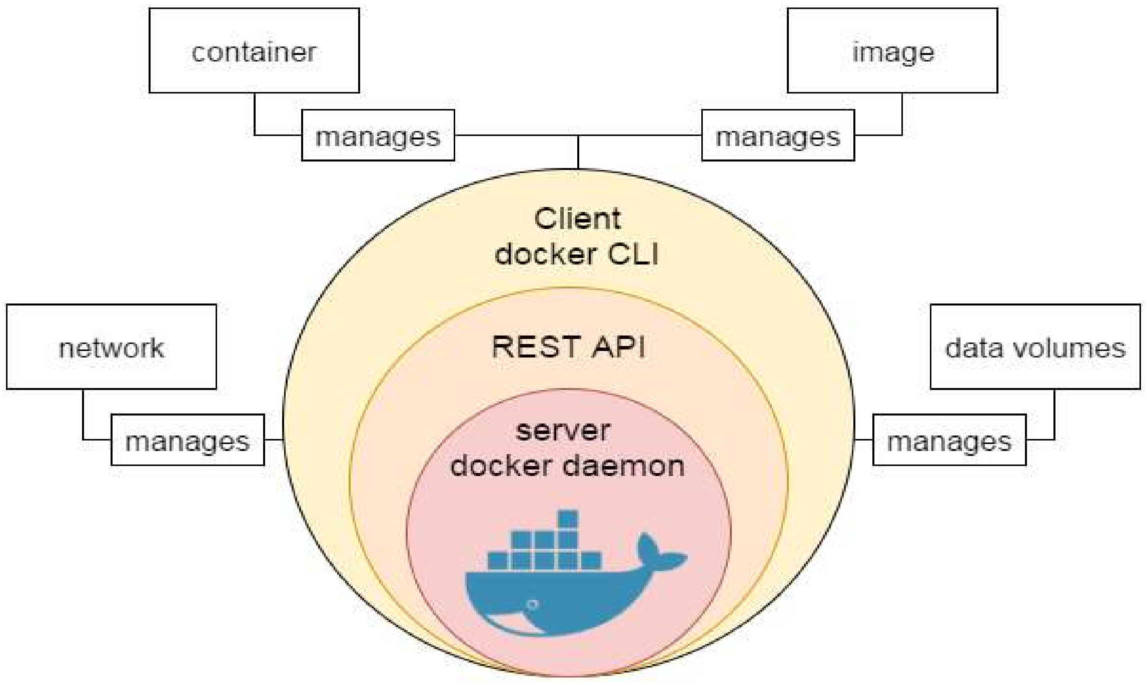
There are the following disadvantages of Docker -

- It increases complexity due to an additional layer.
- In Docker, it is difficult to manage large amount of containers.
- Some features such as container self -registration, containers self-inspects, copying files form host to the container, and more are missing in the Docker.
- Docker is not a good solution for applications that require rich graphical interface.
- Docker provides cross-platform compatibility means if an application is designed to run in a Docker container on Windows, then it can't run on Linux or vice versa.

Docker Engine

It is a client server application that contains the following major components.

- A server which is a type of long-running program called a daemon process.
- The REST API is used to specify interfaces that programs can use to talk to the daemon and instruct it what to do.
- A command line interface client.



Client Server Application

Docker components

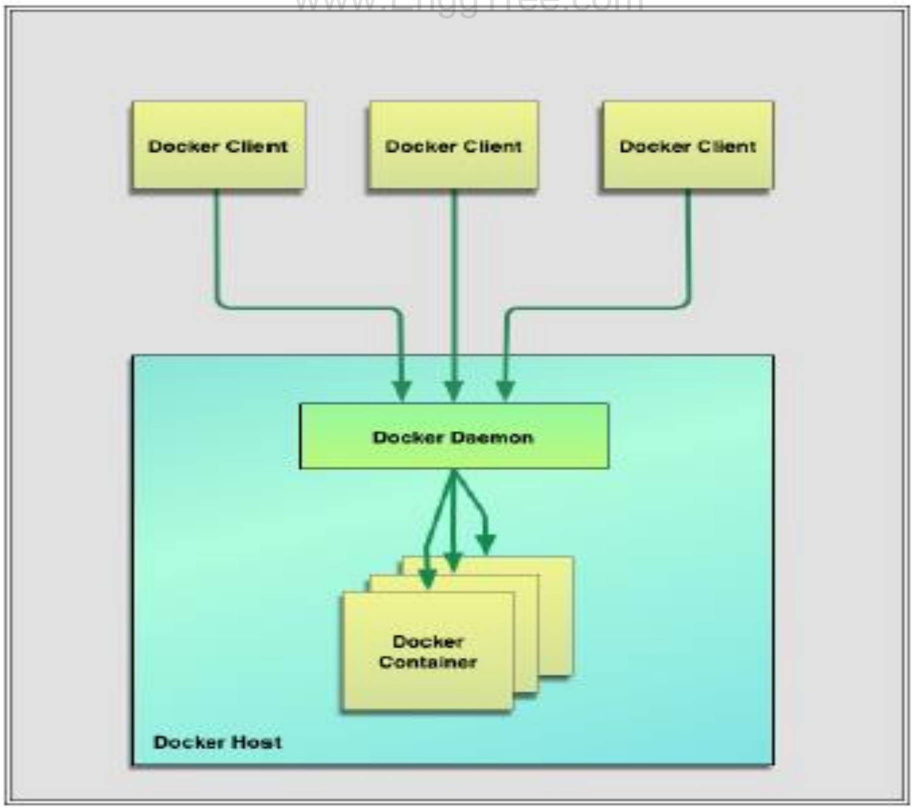
Let's look at the core components that compose Docker:

- The Docker client and server
- Docker Images
- Registries
- Docker Containers

Docker client and server:

Docker is a client-server application. The Docker client talks to the Docker server

or daemon, which, in turn, does all the work. Docker ships with a command line client binary, docker, as well as a full RESTful API. You can run the Docker daemon and client on the same host or connect your local Docker client to a remote daemon running on another host. You can see Docker's architecture depicted here:



Docker Architecture

5. **Explain in detail about Docker images and repositories procedures.**

Docker images

Images are the building blocks of the Docker world. You launch your containers from images. Images are the "build" part of Docker's life cycle. They are a layered format, using Union file systems, that are built step-by-step using a series of instructions. For example:

- Add a file.
- Run a command.
- Open a port.

You can consider images to be the "source code" for your containers. They are highly portable and can be shared, stored, and updated. In the book, we'll learn how to use existing images as well as build our own images.

Registries:

- Docker stores the images you build in registries. There are two types of registries: public and private. Docker, Inc., operates the public registry for images, called the Docker Hub. You can create an account on the Docker Hub and use it to share and store your own images.
- The Docker Hub also contains, at last count, over 10,000 images that other people have built and shared. Want a Docker image for an Nginx web server, the Asterisk open source PABX system, or a MySQL database? All of these are available, along with a whole lot more.
- You can also store images that you want to keep private on the Docker Hub. These images might include source code or other proprietary information you want to keep secure or only share with other members of your team or organization.

Containers

Docker helps you build and deploy containers inside of which you can package your applications and services. As we've just learnt, containers are launched from images and can contain one or more running processes. You can think about images as the building or packing aspect of Docker and the containers as the running or execution aspect of Docker.

A Docker container is:

• An image format.

• A set of standard operations.

• An execution environment.

Docker borrows the concept of the standard shipping container, used to transport goods globally, as a model for its containers. But instead of shipping goods, Docker containers ship software.

Each container contains a software image -- its 'cargo' -- and, like its physical counterpart, allows a set of operations to be performed. For example, it can be created, started, stopped, restarted, and destroyed. Like a shipping container, Docker doesn't care about the contents of the container when performing these actions; for example, whether a container is a web server, a database, or an application server. Each container is loaded the same as any other container.

Docker also doesn't care where you ship your container: you can build on your laptop, upload to a registry, then download to a physical or virtual server, test, deploy to a cluster of a dozen Amazon EC2 hosts, and run. Like a normal shipping container, it is interchangeable, stackable, portable, and as generic as possible.

Docker's technical components

Docker can be run on any x64 host running a modern Linux kernel; we recommend kernel version 3.8 and later. It has low overhead and can be used on servers,desktops, or laptops. It includes:

• A native Linux container format that Docker calls libcontainer, as well as the popular container platform, lxc. The libcontainer format is now the default format.

• Linux kernel namespaces, which provide isolation for filesystems, processes, and networks.

• Filesystem isolation: each container is its own root filesystem.

• Process isolation: each container runs in its own process environment.

• Network isolation: separate virtual interfaces and IP addressing between containers.

• Resource isolation and grouping: resources like CPU and memory are allocated individually to each Docker container using the cgroups, or control groups, kernel feature.

• Copy-on-write: filesystems are created with copy-on-write, meaning they are layered and fast and require limited disk usage.

• Logging: STDOUT, STDERR and STDIN from the container are collected, logged, and available for analysis or trouble-shooting.

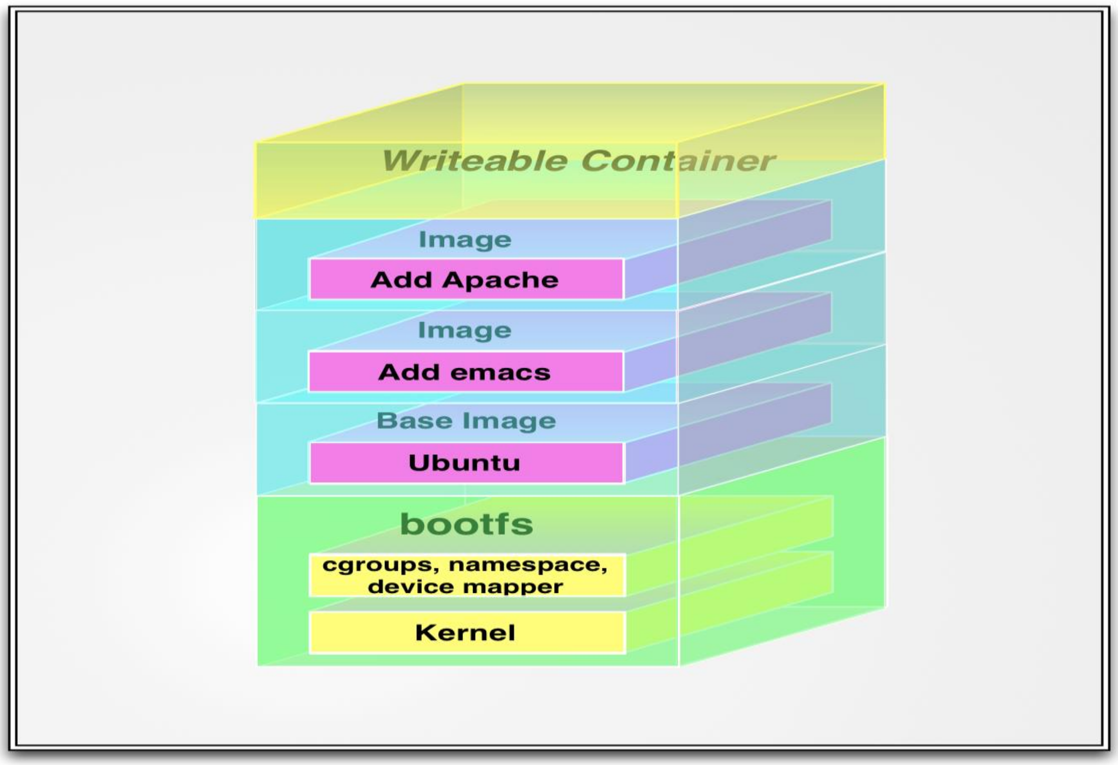
• Interactive shell: You can create a pseudo-tty and attach to STDIN to provide an interactive shell to your container.

What is a Docker image?

A Docker image is made up of filesystems layered over each other. At the base is a boot filesystem, bootfs, which resembles the typical Linux/Unix boot filesystem. A Docker user will probably never interact with the boot filesystem. Indeed, when a container has booted, it is moved into memory, and the boot filesystem is unmounted to free up the RAM used by the initrd disk image.

Docker calls each of these filesystems images. Images can be layered on top of one another. The image below is called the parent image and you can traverse each layer until you reach the bottom of the image stack where the final image is called the base image. Finally, when a container is launched from an image, Docker mounts a read-write filesystem on top of any layers below. This is where whatever processes we want our Docker container to run will execute.

This sounds confusing, so perhaps it is best represented by a diagram.



The Docker file System layers

When Docker first starts a container, the initial read-write layer is empty. As changes occur, they are applied to this layer; for example, if you want to change a file, then that file will be copied from the read-only layer below into the read- write layer. The read-only version of the file will still exist but is now hidden underneath the copy.

Listing Docker images

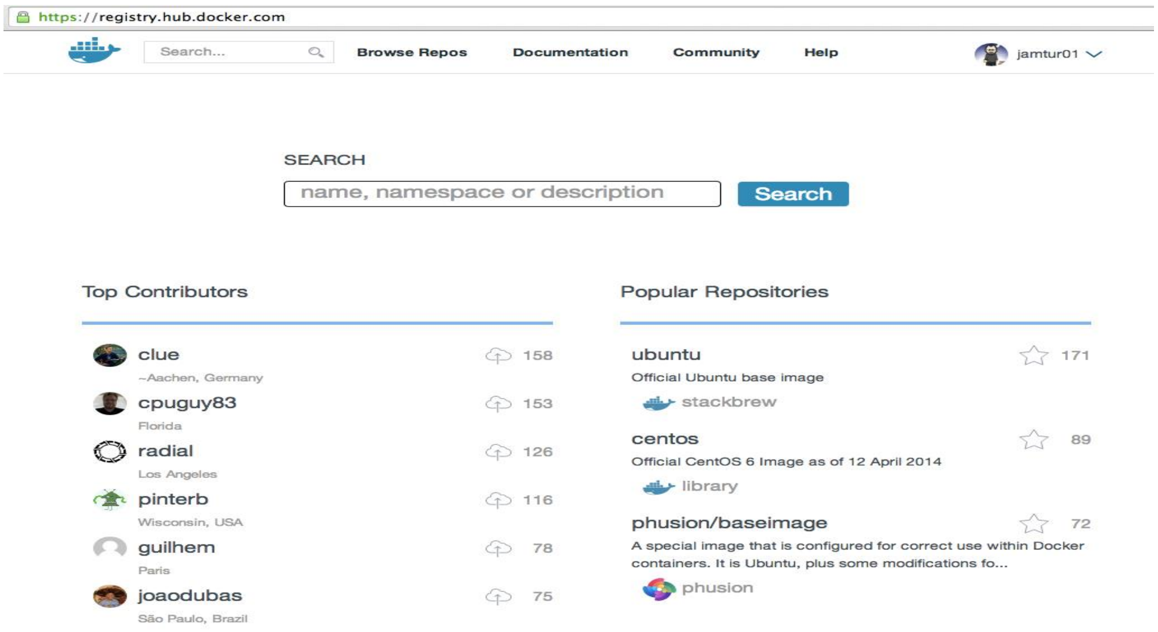
Let's get started with Docker images by looking at what images are available to us on our Docker host. We can do this using the docker images command.

Listing Docker images

```
$ sudo docker images
```

```
REPOSITORY TAG    IMAGE ID      CREATED   VIRTUAL SIZE
ubuntu            c4ff7513909d 6 days 225.4 MB
```

That image was downloaded from a repository. Images live inside repositories, and repositories live on registries. The default registry is the public registry man- aged by Docker, Inc., Docker Hub.



Top Contributors

clue ~Aachen, Germany 158
cpuguy83 Florida 153
radial Los Angeles 126
pinterb Wisconsin, USA 116
guilhem Paris 78
joaodubas São Paulo, Brazil 75

Popular Repositories

ubuntu 171
Official Ubuntu base image
stackbrew

centos 89
Official CentOS 6 Image as of 12 April 2014
library

phusion/baseimage 72
A special image that is configured for correct use within Docker containers. It is Ubuntu, plus some modifications fo...
phusion

Each repository can contain multiple images (e.g., the `ubuntu` repository contains images for Ubuntu 12.04, 12.10, 13.04, 13.10, and 14.04). Let's get the rest of the images in the `ubuntu` repository now.

```
$ sudo docker pull ubuntu
Pulling repository ubuntu
c4ff7513909d: Pulling dependent layers
3db9c44f4520: Pulling dependent layers
75204fdb260b: Pulling dependent layers
. . .
```

Here we've used the `docker pull` command to pull down the entire contents of the `ubuntu` repository.

Let's see what our `docker images` command reveals now.

```
$ sudo docker images
REPOSITORY TAG     IMAGE ID     CREATED     VIRTUAL SIZE
ubuntu     13.10   5e019ab7bf6d 2 weeks ago 180 MB ubuntu
saucy   5e019ab7bf6d 2 weeks ago 180 MB ubuntu     12.04
74fe38d11401  2  weeks  ago  209.6  MB ubuntu       precise
74fe38d11401  2  weeks  ago  209.6  MB ubuntu        12.10
a7cf8ae4e998  2  weeks  ago  171.3  MB ubuntu       quantal
a7cf8ae4e998  2  weeks  ago  171.3  MB ubuntu        14.04
99ec81b80c55  2  weeks  ago  266  MB ubuntu         latest
c4ff7513909d  6  days  ago   225.4  MB ubuntu       trusty
99ec81b80c55  2  weeks  ago  266  MB ubuntu         raring
316b678ddf48  2  weeks  ago  169.4  MB ubuntu        13.04
316b678ddf48  2  weeks  ago  169.4  MB ubuntu        10.04
3db9c44f4520  3  weeks  ago  183  MB ubuntu          lucid
3db9c44f4520 3 weeks ago 183 MB
```

You can see we've now got a series of ubuntu images. We can see that the ubuntu image is actually a series of images collected under a single repository.

We can refer to a specific image inside a repository by suffixing the repository name with a colon and a tag name, for example:

Running a tagged Docker image

```
$ sudo docker run -t -i --name new_container ubuntu:12.04 /bin/↩
  bash
root@79e36bff89b4:/#
```

This launches a container from the ubuntu:12.04 image, which is an Ubuntu 12.04 operating system. We can also see that some images with the same ID (see image ID 74fe38d11401) are tagged more than once. Image ID 74fe38d11401 is actu- ally tagged both 12.04 and precise: the version number and code name for that Ubuntu release, respectively.

Pulling images

When we run a container from images with the **docker run** command, if the image isn't present locally already then Docker will download it from the Docker Hub. By default, if you don't specify a specific tag, Docker will download the **latest** tag, for example:

Docker run and the default latest tag

```
$ sudo docker run -t -i --name next_container ubuntu
/bin/bash root@23a42cee91c3:/#
```

will download the **ubuntu:latest** image if it isnt already present on the host. Alternatively, we can use the **docker pull** command to pull images down our- selves. Using **docker pull** saves us some time launching a container from a new image. Let's see that now by pulling down the **fedora** base image.

Pulling the fedora image

```
$ sudo docker pull fedora
Pulling repository fedora
5cc9e91966f7: Download complete b7de3133ff98:
Download complete
511136ea3c5a: Download complete ef52fb1fe610:
Download complete
```

Let's see this new image on our Docker host using the **docker images** command. This time, however, let's narrow our review of the images to only the **fedora**↩ images. To do so, we can specify the image name after the **docker images**↩ command.

g 4.8: Viewing the fedora image

```
$ sudo docker images fedora
```

```
REPOSITORY TAG IMAGE ID CREATED VIRTUAL SIZE fedora rawhide
5cc9e91966f7 6 days ago 372.7 MB fedora 20 b7de3133ff98 3 weeks
ago 372.7 MB fedora heisenbug b7de3133ff98 3 weeks ago 372.7 MB fedora
latest b7de3133ff98 3 weeks ago 372.7 MB
```

We can see that the `fedora` image contains the development Rawhide release as well as Fedora 20. We can also see that the Fedora 20 release is tagged in three ways -- 20, `heisenbug`, and `latest` -- but it is the same image (we can see all three entries have an ID of `b7de3133ff98`). If we wanted the Fedora 20 image, therefore, we could use any of the following:

- `fedora:20`
- `fedora:heisenbug`
- `fedora:latest`

We could have also just downloaded one tagged image using the `docker pull` command.

```
$ sudo docker pull fedora:20
```

This would have just pulled the `fedora:20` image.

Searching for images

We can also search all of the publicly available images on Docker Hub using the `docker search` command:

```
$ sudo docker search puppet
NAME                    DESCRIPTION STARS OFFICIAL AUTOMATED
wfarr/puppet-module...
jamtur01/puppetmaster
. . .
```

Here, we've searched the Docker Hub for the term `puppet`. It'll search images and return:

- Repository names
- Image descriptions
- Stars - these measure the popularity of an image
- Official - an image managed by the upstream developer (e.g., the `fedora` image managed by the Fedora team)
- Automated - an image built by the Docker Hub's Automated Build process

Let's pull down one of these images.

g 4.11: Pulling down the jamtur01/puppetmaster image

```
$ sudo docker pull jamtur01/puppetmaster
```

This will pull down the `jamtur01/puppetmaster` image (which, by the way, con- tains a pre-installed Puppet master).

We can then use this image to build a new container. Let's do that now using the `docker run` command again.

g 4.12: Creating a Docker container from the Puppet master image

```
$   sudo   docker   run   -i   -t   jamtur01/puppetmaster   /bin/bash
root@4655dee672d3:/# facter
architecture  =>  amd64 augeasversion
=> 1.2.0
. . .
root@4655dee672d3:/# puppet --version
3.4.3
```

You can see we've launched a new container from our `jamtur01/puppetmaster` im- age. We've launched the container interactively and told the container to run the Bash shell. Once inside the container's shell, we've run Facter (Puppet's inventory application), which was pre-installed on our image. From inside the container, we've also run the puppet binary to confirm it is installed.

Docker Hub:

Docker Hub is a repository service and it is a cloud-based service where people push their Docker Container Images and also pull the Docker Container Images from the **Docker Hub** anytime or anywhere via the internet. It provides features such as you can push your images as private or public. Mainly DevOps team uses the Docker Hub. It is an open-source tool and freely available for all operating systems. It is like storage where we store the images and pull the images when it is required. When a person wants to push/pull images from the Docker Hub they must have a basic knowledge of Docker. Let us discuss the requirements of the Docker tool.

Docker is a tool nowadays enterprises adopting rapidly day by day. When a Developer team wants to share the project with all dependencies for testing then the developer can push their code on **Docker Hub** with all dependencies. Firstly create the **Images** and push the Image on Docker Hub. After that, the testing team will pull the same image from the Docker Hub eliminating the need for any type of file, software, or plugins for running the Image because the Developer team shares the image with all dependencies.

Docker Hub Features

- Storage, management, and sharing of images with others are made simple via Docker Hub.
- Docker Hub runs the necessary security checks on our images and generates a full report on any security flaws.
- Docker Hub can automate the processes like Continuous deployment and Continuous testing by triggering the Webhooks when the new image is pushed into Docker Hub.
- With the help of Docker Hub, we can manage the permission for the users, teams, and organizations.
- We can integrate Docker Hub into our tools like GitHub, Jenkins which makes workflows easy

Advantages of Docker Hub

- Docker Container Images are light in weight.
- We can push the images within a minute and with help of a command.
- It is a secure method and also provides a feature like pushing the private image or public image.

- Docker hub plays a very important role in industries as it becomes more popular day by day and it acts as a bridge between the developer team and the testing team.
- If a person wants to share their code, software any type of file for public use, you can just make the images public on the docker hub.

Creating First Repository in Docker Hub Using GUI

Step 1: We must open Docker Hub first, then select Create Repository.

Step 2: After that, we will be taken to a screen for configuring the repository, where we must choose the namespace, repository name, and optional description. In the visibility area, as indicated in the picture, there are two options: Public and Private. We can choose any of them depending on the type of organization you are in. If you chose Public, everyone will be able to push-pull and use the image because it will be accessible to everyone. If you select the private option, only those with access to that image can view and utilize it. it.

Step 3: At finally repository is created with the help of the Docker Commands we can push or pull the image.

docker push <your-username>/my-testprivate-repo>.

How To Push or Pull Images from Docker Hub?

To get started with Docker Hub you should be able to get familiar with the below two commands:

1. Push Command

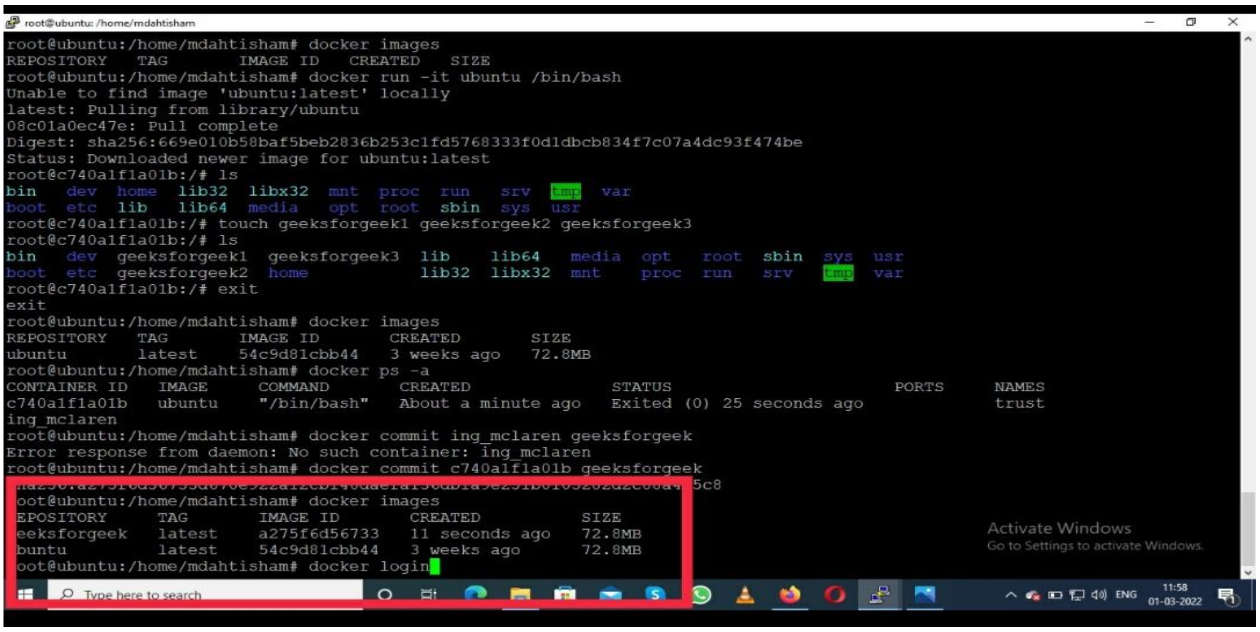
This command as the name suggests itself is used to pushing a docker image onto the docker hub.

Implementation

Follow this example to get an idea of the push command:

- Open Docker in your system.
- Locate the Images that you want to push using the below command:

# docker images



```
root@ubuntu:/home/mdahtisham# docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
root@ubuntu:/home/mdahtisham# docker run -it ubuntu /bin/bash
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
08c01a0ec47e: Pull complete
Digest: sha256:669e010b58baf5beb2836b253c1fd5768333f0d1dbcb834f7c07a4dc93f474be
Status: Downloaded newer image for ubuntu:latest
root@c740a1f1a01b:/# ls
bin   dev  home  lib32  libx32  mnt  proc  run   srv  tmp  var
boot  etc  lib   lib64  media   opt  root  sbin  sys  usr
root@c740a1f1a01b:/# touch geeksforgeek1 geeksforgeek2 geeksforgeek3
root@c740a1f1a01b:/# ls
bin   dev  geeksforgeek1  geeksforgeek3  lib    lib64   media  opt   root  sbin  sys  usr
boot  etc  geeksforgeek2  home           lib32  libx32  mnt    proc  run   srv   tmp  var
root@c740a1f1a01b:/# exit
exit
root@ubuntu:/home/mdahtisham# docker images
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
ubuntu       latest    54c9d81cbb44   3 weeks ago   72.8MB
root@ubuntu:/home/mdahtisham# docker ps -a
CONTAINER ID   IMAGE     COMMAND       CREATED              STATUS                      PORTS     NAMES
c740a1f1a01b   ubuntu    "/bin/bash"   About a minute ago   Exited (0) 25 seconds ago             trust
ing_mclaren
root@ubuntu:/home/mdahtisham# docker commit ing_mclaren geeksforgeek
Error response from daemon: No such container: ing_mclaren
root@ubuntu:/home/mdahtisham# docker commit c740a1f1a01b geeksforgeek
[illegible]5c8
oot@ubuntu:/home/mdahtisham# docker images
EPOSITORY     TAG       IMAGE ID       CREATED          SIZE
eeksforgeek   latest    a275f6d56733   11 seconds ago   72.8MB
buntu         latest    54c9d81cbb44   3 weeks ago      72.8MB
oot@ubuntu:/home/mdahtisham# docker login
```

root@ubuntu: /home/mdahtisham
Activate Windows
Go to Settings to activate Windows.
Type here to search
ENG
11:58
01-03-2022

The above command will list all the images on your system.

Step 1: Go to the browser and search *hub.docker.com.*

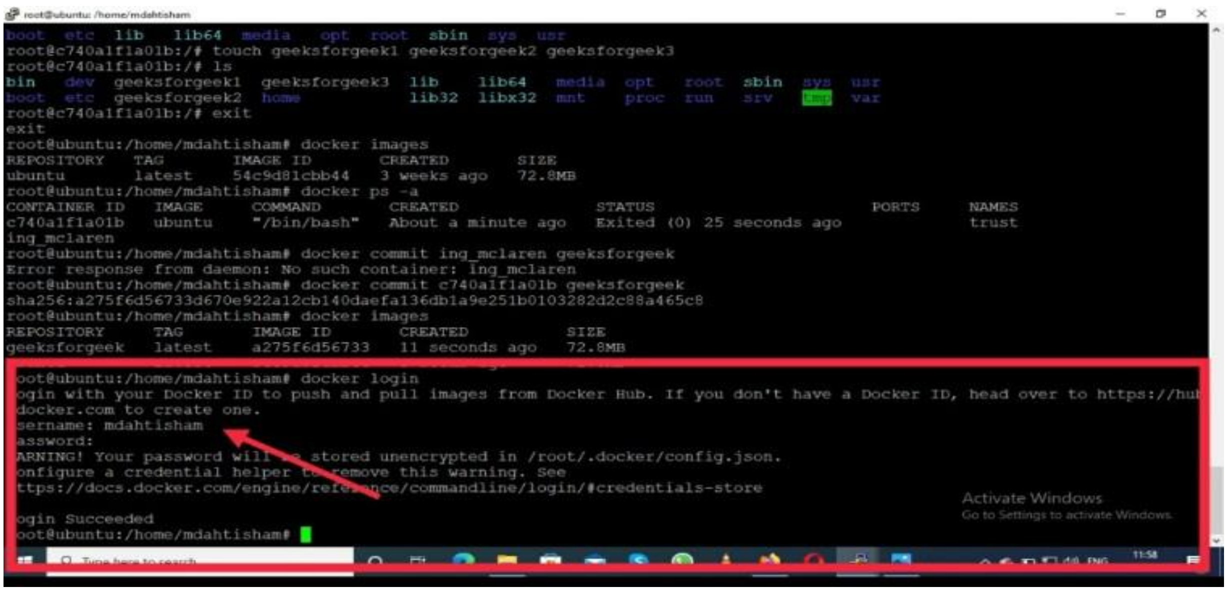
Step 2: Sign up on the docker hub if you do not have a docker hub account, after login on to docker hub.

Step 3: Back to the docker terminal and execute the below command:

# docker login

Step 4: Then give your credential and type in your docker hub username or password.

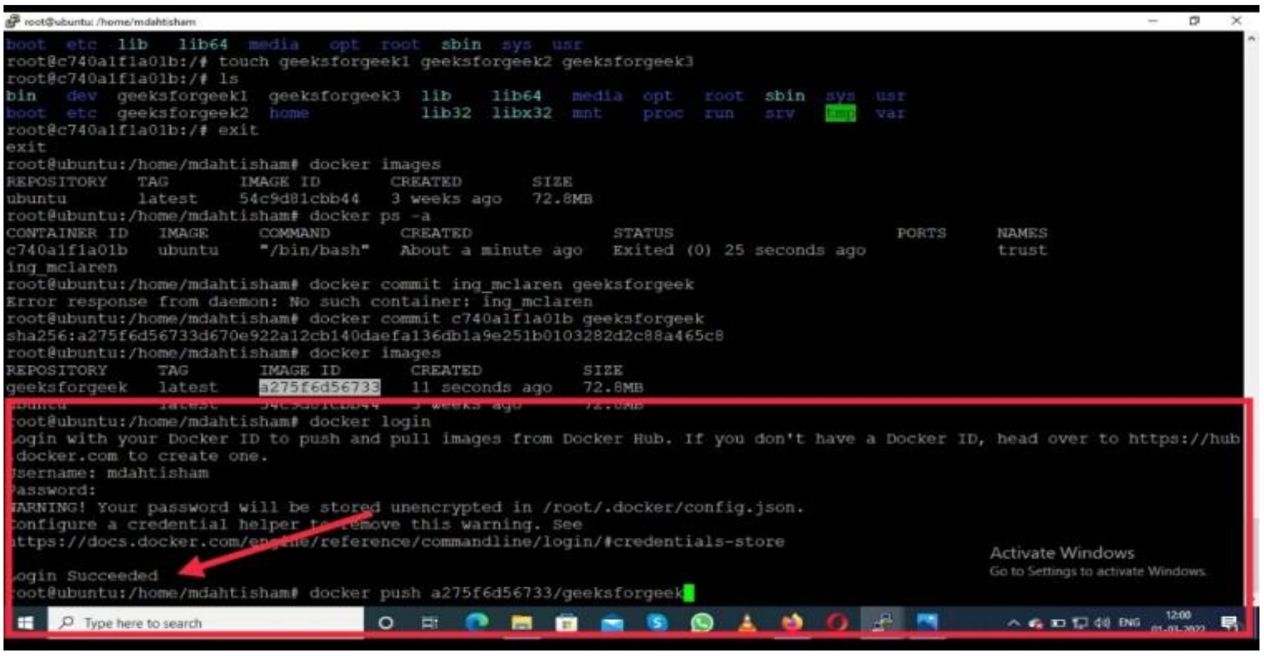
- username
- password



```
boot  etc  lib   lib64  media   opt  root  sbin  sys  usr
root@c740a1f1a01b:/# touch geeksforgeek1 geeksforgeek2 geeksforgeek3
root@c740a1f1a01b:/# ls
bin   dev  geeksforgeek1  geeksforgeek3  lib    lib64  media  opt   root  sbin  sys  usr
boot  etc  geeksforgeek2  home           lib32  libx32 mnt    proc  run   srv   tmp  var
root@c740a1f1a01b:/# exit
exit
root@ubuntu:/home/mdahtisham# docker images
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
ubuntu       latest    54c9d81cbb44   3 weeks ago   72.8MB
root@ubuntu:/home/mdahtisham# docker ps -a
CONTAINER ID   IMAGE    COMMAND       CREATED            STATUS                      PORTS     NAMES
c740a1f1a01b   ubuntu   "/bin/bash"   About a minute ago   Exited (0) 25 seconds ago           trust
ing_mclaren
root@ubuntu:/home/mdahtisham# docker commit ing_mclaren geeksforgeek
Error response from daemon: No such container: ing_mclaren
root@ubuntu:/home/mdahtisham# docker commit c740a1f1a01b geeksforgeek
sha256:a275f6d56733d670e922a12cb140daefa136db1a9e251b0103282d2c88a465c8
root@ubuntu:/home/mdahtisham# docker images
REPOSITORY     TAG       IMAGE ID       CREATED          SIZE
geeksforgeek   latest    a275f6d56733   11 seconds ago   72.8MB
oot@ubuntu:/home/mdahtisham# docker login
ogin with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hu
docker.com to create one.
sername: mdahtisham
assword:
ARNING! Your password will be stored unencrypted in /root/.docker/config.json.
onfigure a credential helper to remove this warning. See
ttps://docs.docker.com/engine/reference/commandline/login/#credentials-store

ogin Succeeded
oot@ubuntu:/home/mdahtisham#
```

Step 5: After that hit the Enter key you will see login success on your screen.



```
boot  etc  lib   lib64  media   opt  root  sbin  sys  usr
root@c740a1f1a01b:/# touch geeksforgeek1 geeksforgeek2 geeksforgeek3
root@c740a1f1a01b:/# ls
bin   dev  geeksforgeek1  geeksforgeek3  lib    lib64  media  opt   root  sbin  sys  usr
boot  etc  geeksforgeek2  home           lib32  libx32 mnt    proc  run   srv   tmp  var
root@c740a1f1a01b:/# exit
exit
root@ubuntu:/home/mdahtisham# docker images
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
ubuntu       latest    54c9d81cbb44   3 weeks ago   72.8MB
root@ubuntu:/home/mdahtisham# docker ps -a
CONTAINER ID   IMAGE    COMMAND       CREATED            STATUS                      PORTS     NAMES
c740a1f1a01b   ubuntu   "/bin/bash"   About a minute ago   Exited (0) 25 seconds ago           trust
ing_mclaren
root@ubuntu:/home/mdahtisham# docker commit ing_mclaren geeksforgeek
Error response from daemon: No such container: ing_mclaren
root@ubuntu:/home/mdahtisham# docker commit c740a1f1a01b geeksforgeek
sha256:a275f6d56733d670e922a12cb140daefa136db1a9e251b0103282d2c88a465c8
root@ubuntu:/home/mdahtisham# docker images
REPOSITORY     TAG       IMAGE ID       CREATED          SIZE
geeksforgeek   latest    a275f6d56733   11 seconds ago   72.8MB
oot@ubuntu:/home/mdahtisham# docker login
ogin with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub
docker.com to create one.
sername: mdahtisham
assword:
ARNING! Your password will be stored unencrypted in /root/.docker/config.json.
onfigure a credential helper to remove this warning. See
ttps://docs.docker.com/engine/reference/commandline/login/#credentials-store

ogin Succeeded
oot@ubuntu:/home/mdahtisham# docker push a275f6d56733/geeksforgeek
```

Step 7: Then type the tag images name, docker hub username, and give the name it appears on the docker hub using the below command:

# docker tag geeksforgeek mdahtisham/geeksimage

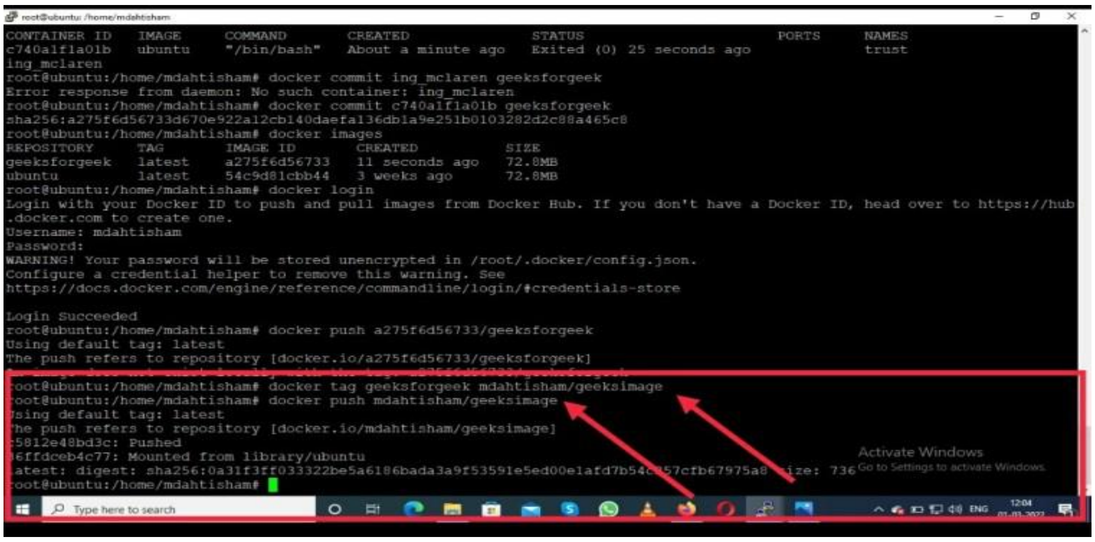
geeksforgeek - Image name

mdahtisham - Docker hub username

geeksimage - With this name Image will appear on the docker hub

Step 8: Now push your image using the below command:

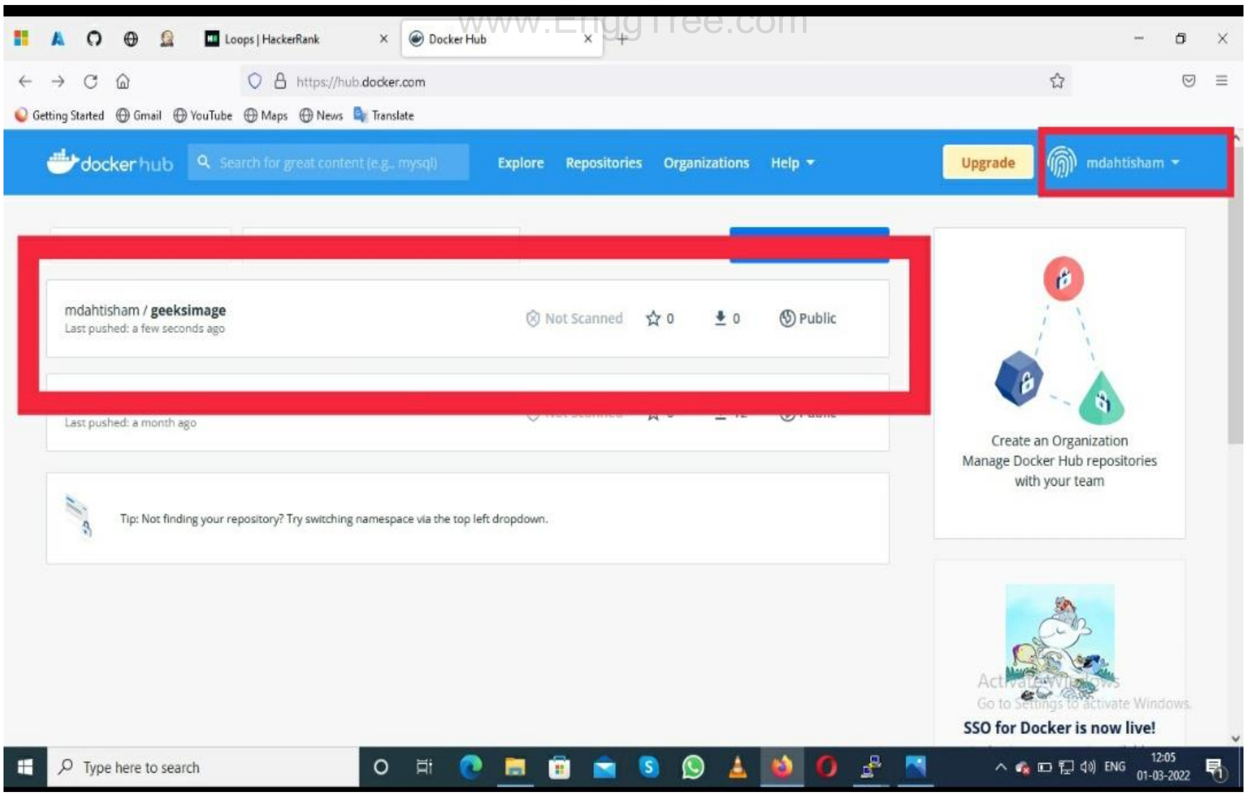
# docker push mdahtisham/geeksimage



```
CONTAINER ID   IMAGE     COMMAND       CREATED            STATUS                      PORTS     NAMES
c740a1f1a01b   ubuntu    "/bin/bash"   About a minute ago Exited (0) 25 seconds ago             trust
ing_mclaren
root@ubuntu:/home/mdahtisham# docker commit ing_mclaren geeksforgeek
Error response from daemon: No such container: ing_mclaren
root@ubuntu:/home/mdahtisham# docker commit c740a1f1a01b geeksforgeek
sha256:a275f6d56733d670e922a12cb140daefa136db1a9e251b0103282d2c88a465c8
root@ubuntu:/home/mdahtisham# docker images
REPOSITORY     TAG      IMAGE ID       CREATED          SIZE
geeksforgeek   latest   a275f6d56733   11 seconds ago   72.8MB
ubuntu         latest   54c9d81cbb44   3 weeks ago      72.8MB
root@ubuntu:/home/mdahtisham# docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub
.docker.com to create one.
Username: mdahtisham
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
root@ubuntu:/home/mdahtisham# docker push a275f6d56733/geeksforgeek
Using default tag: latest
The push refers to repository [docker.io/a275f6d56733/geeksforgeek]
[illegible]
oot@ubuntu:/home/mdahtisham# docker tag geeksforgeek mdahtisham/geeksimage
oot@ubuntu:/home/mdahtisham# docker push mdahtisham/geeksimage
Using default tag: latest
he push refers to repository [docker.io/mdahtisham/geeksimage]
5812e48bd3c: Pushed
6ffdceb4c77: Mounted from library/ubuntu
atest: digest: sha256:0a31f3ff033322be5a6186bada3a9f53591e5ed00e1afd7b54c857cfb67975a8 size: 736
oot@ubuntu:/home/mdahtisham#
```

Note: Below you can see the Docker Image successfully pushed on the docker hub: mdahtisham/geeksimage



mdahtisham / geeksimage
Last pushed: a few seconds ago
Not Scanned ☆ 0 ⬇ 0 Public

Last pushed: a month ago

Tip: Not finding your repository? Try switching namespace via the top left dropdown.

Create an Organization
Manage Docker Hub repositories with your team

SSO for Docker is now live!

2. Pull Command

The pull command is used to get an image from the Docker Hub.

Implementation:

Follow the example to get an overview of the pull command in Docker:

Step 1: Now you can search the image using the below command in docker as follows:

# docker search *imagename*

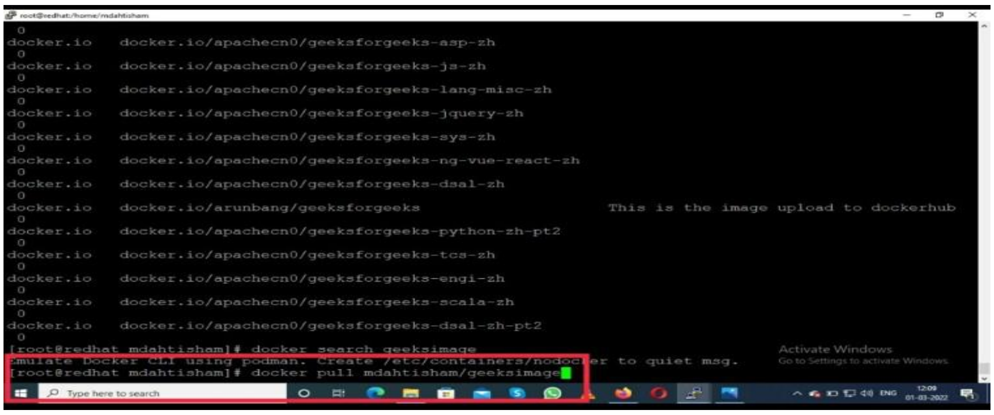
One can see all images on your screen if available images with this name.One can also pull the images if one knows the exact name

Step 2: Now pull the image see the below command.

docker pull mdahtisham/geeksimage

mdahtisham - Docker Hub username

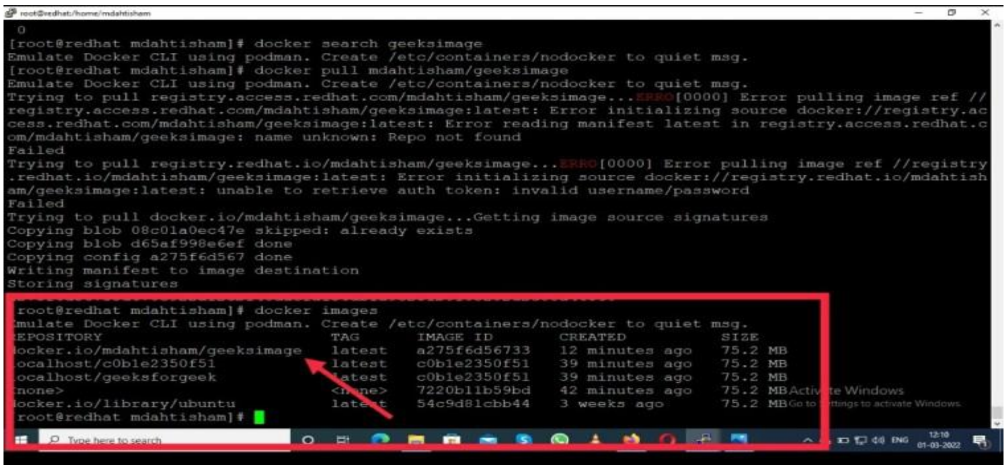
geeksimage - With this name Image will appear on the docker hub



```
root@redhat:/home/mdahtisham
 0
docker.io   docker.io/apachecn0/geeksforgeeks-asp-zh
 0
docker.io   docker.io/apachecn0/geeksforgeeks-js-zh
 0
docker.io   docker.io/apachecn0/geeksforgeeks-lang-misc-zh
 0
docker.io   docker.io/apachecn0/geeksforgeeks-jquery-zh
 0
docker.io   docker.io/apachecn0/geeksforgeeks-sys-zh
 0
docker.io   docker.io/apachecn0/geeksforgeeks-ng-vue-react-zh
 0
docker.io   docker.io/apachecn0/geeksforgeeks-dsal-zh
 0
docker.io   docker.io/arunbang/geeksforgeeks                    This is the image upload to dockerhub
 0
docker.io   docker.io/apachecn0/geeksforgeeks-python-zh-pt2
 0
docker.io   docker.io/apachecn0/geeksforgeeks-tcs-zh
 0
docker.io   docker.io/apachecn0/geeksforgeeks-engi-zh
 0
docker.io   docker.io/apachecn0/geeksforgeeks-scala-zh
 0
docker.io   docker.io/apachecn0/geeksforgeeks-dsal-zh-pt2
 0
[root@redhat mdahtisham]# docker search geeksimage
Emulate Docker CLI using podman. Create /etc/containers/nodocker to quiet msg.
[root@redhat mdahtisham]# docker pull mdahtisham/geeksimage
```

Step 3: Now check for the pulled image using the below command as follows:

# docker images



```
 0
[root@redhat mdahtisham]# docker search geeksimage
Emulate Docker CLI using podman. Create /etc/containers/nodocker to quiet msg.
[root@redhat mdahtisham]# docker pull mdahtisham/geeksimage
Emulate Docker CLI using podman. Create /etc/containers/nodocker to quiet msg.
Trying to pull registry.access.redhat.com/mdahtisham/geeksimage...ERRO[0000] Error pulling image ref //
registry.access.redhat.com/mdahtisham/geeksimage:latest: Error initializing source docker://registry.ac
cess.redhat.com/mdahtisham/geeksimage:latest: Error reading manifest latest in registry.access.redhat.c
om/mdahtisham/geeksimage: name unknown: Repo not found
Failed
Trying to pull registry.redhat.io/mdahtisham/geeksimage...ERRO[0000] Error pulling image ref //registry
.redhat.io/mdahtisham/geeksimage:latest: Error initializing source docker://registry.redhat.io/mdahtish
am/geeksimage:latest: unable to retrieve auth token: invalid username/password
Failed
Trying to pull docker.io/mdahtisham/geeksimage...Getting image source signatures
Copying blob 08c01a0ec47e skipped: already exists
Copying blob d65af998e6ef done
Copying config a275f6d567 done
Writing manifest to image destination
Storing signatures
root@redhat mdahtisham]# docker images
mulate Docker CLI using podman. Create /etc/containers/nodocker to quiet msg.
EPOSITORY                        TAG       IMAGE ID       CREATED          SIZE
ocker.io/mdahtisham/geeksimage   latest    a275f6d56733   12 minutes ago   75.2 MB
ocalhost/c0b1e2350f51            latest    c0b1e2350f51   39 minutes ago   75.2 MB
ocalhost/geeksforgeek            atest     c0b1e2350f51   39 minutes ago   75.2 MB
none>                            <n ne>    7220b11b59bd   42 minutes ago   75.2 MB
ocker.io/library/ubuntu          late t    54c9d81cbb44   3 weeks ago      75.2 MB
root@redhat mdahtisham]#
```

PART-A

1. **Define Virtualization.**

   Virtualization is a process that allows a computer to share its hardware resources with multiple digitally separated environments. Each virtualized environment runs within its allocated resources, such as memory, processing power, and storage.

2. **List the types of Virtualization in cloud.**

   Server Virtualization

   Network Virtualization

   Storage Virtualization

   Desktop Virtualization

   Application Virtualization

3. **What is meant by Desktop Virtualization?**

   The processing of multiple virtual desktops occurs on one or a few physical servers, typically at the centralized data center. The copy of the OS and applications that each end user utilizes will typically be cached in memory as one image on the physical server.

   The Desktop virtualization provides a virtual desktop environment where client can access the system resources remotely through the network.

4. **List out the advantages of Desktop virtualization.**

- It provides easier management of devices and operating systems due to centralized management.
- It reduces capital expenditure and maintenance cost of hardware due to consolidation of multiple operating systems into a single physical server,
- Upgradation of operating system is easier
- It can facilitate Work from Home feature for IT Employees due to the desktop operating system delivery over the internet.

5. **What is meant by Network Virtualization?**

The Network virtualization is the ability to create virtual networks that are decoupled from the underlying network hardware. This ensures the network can better integrate with and support increasingly virtual environments. It has capability to combine multiple physical networks into one virtual, or it can divide one physical network into separate, independent virtual networks.

6. **List out the advantages of network virtualization.**

The benefits of network virtualization are

- It consolidates the physical hardware of a network into a single virtual network that reduce the management overhead of network resources.
- It gives better scalability and flexibility in network operations.
- It provides automated provisioning and management of network resources.

7. **What is meant by Storage Virtualization?**

Storage virtualization is the process of grouping multiple physical storages using software to appear as a single storage device in a virtual form.

It pools the physical storage from different network storage devices and makes it appear to be a single storage unit that is handled from a single console. Storage virtualization helps to address the storage and data management issues by facilitating easy backup, archiving and recovery tasks in less time.

8. **List out the advantages of storage virtualization.**

The benefits provided by storage virtualization are

- Automated management of storage mediums with estimated of down time.
- Enhanced storage management in heterogeneous IT environment.
- Better storage availability and optimum storage utilization.
- It gives scalability and redundancy in storage.
- It provides consummate features like disaster recovery, high availability, consistency, replication & re-duplication of data.

9. **What is meant by Server Virtualization?**

A Server virtualization is the process of dividing a physical server into multiple unique and isolated virtual servers by means of software. It partitions a single physical server into the multiple virtual servers; each virtual server can run its own operating system and applications independently.

10. **List out the advantages of server virtualization.**

The benefits of server virtualization are

- It gives quick deployment and provisioning of virtual operating system.
- It has reduced the capital expenditure due to consolidation of multiple servers into a single physical server which eliminate the cost of multiple physical hardware.
- It provides ease in development & testing.
- It makes optimum resource utilization of physical server.

11. **What is meant by Application Virtualization**

Application virtualization is a technology that encapsulates an application from the underlying operating system on which it is executed. It enables access to an application without needing to install it on the local or target device. From the user's perspective, the application works and interacts like it's native on the device.

It allows to use any cloud client which supports BYOD like Thin client, Thick client, Mobile client, PDA and so on.

12. **List out the advantages of Application virtualization.**

- It allows for cross-platform operations like running Windows applications on Linux or android and vice versa.
- It allows to run applications that have legacy issues like supported on older Operating systems.
- It avoids conflict between the other virtualized applications
- It allows a user to run more than one instance of an application at same time
- It reduces system integration and administration costs by maintaining a common software baseline across multiple diverse computers in an organization.

13. **List out the Live VM Migration Steps.**

Live migration of a VM consists of the following six steps:

Steps 0 and 1: Start migration.

Steps 2: Transfer memory.

Step 3: Suspend the VM and copy the last portion of the data.

Steps 4 and 5: Commit and activate the new host.

14. **What is File Migration?**

- To support VM migration, a system must provide each VM with a consistent, location-independent view of the file system that is available on all hosts.
- A simple way to achieve this is to provide each VM with its own virtual disk, which the file system is mapped to, and transport the contents of this virtual disk along with the other states of the VM.

15. **What is Memory Migration?**

- Memory Migration is one of the most important aspects of VM migration. Moving the memory instance of a VM from one physical host to another can be approached in any number of ways.
- Memory migration can be in a range of hundreds of megabytes to a few gigabytes in a typical system today, and it needs to be done in an efficient manner.

16. **What is Network Migration ?**

A migrating VM should maintain all open network connections without relying on forwarding mechanisms on the original host or on support from mobility or redirection mechanisms.

- To enable remote systems to locate and communicate with a VM, each VM must be assigned a virtual IP address known to other entities.

17. **What is Dynamic Deployment of Virtual Clusters?**

The Cellular Disco at Stanford is a virtual cluster built in a shared-memory multiprocessor system. The INRIA virtual cluster was built to test parallel algorithm performance. The COD (Cluster-on-Demand) project is a virtual cluster management system for dynamic allocation of servers from a computing pool to multiple virtual clusters.

18. **What is Docker?**

Docker is an open-source centralized platform designed to create, deploy, and run applications. Docker uses container on the host's operating system to run applications. It allows applications to use the same Linux kernel as a system on the host computer, rather than creating a whole virtual operating system. Containers ensure that our application works in any environment like development, test, or production.

Docker includes components such as Docker client, Docker server, Docker machine, Docker hub, Docker composes, etc.

19. **List out the Advantages of Docker.**

There are the following advantages of Docker –

It runs the container in seconds instead of minutes.

- It uses less memory.
- It provides lightweight virtualization.
- It does not a require full operating system to run applications.

20. **List out the Disadvantages of Docker.**

There are the following disadvantages of Docker -

- It increases complexity due to an additional layer.
- In Docker, it is difficult to manage large amount of containers.
- Some features such as container self -registration, containers self-inspects, copying files form host to the container, and more are missing in the Docker.

21. **Define Docker Engine.**

It is a client server application that contains the following major components.

- A server which is a type of long-running program called a daemon process.
- The REST API is used to specify interfaces that programs can use to talk to the daemon and instruct it what to do.
- A command line interface client.

22. **What are Docker components?**

Let's look at the core components that compose Docker:

• The Docker client and server

• Docker Images

• Registries

• Docker Containers

23. **Define Docker images.**

Images are the building blocks of the Docker world. You launch your containers from images. Images are the "build" part of Docker's life cycle. They are a layered format, using Union file systems, that are built step-by-step using a series of instructions. For example:

• Add a file.

• Run a command.

• Open a port.

24. **Define Containers.**

Docker helps you build and deploy containers inside of which you can package your applications and services. As we've just learnt, containers are launched from images and can contain one or more running processes. You can think about images as the building or packing aspect of Docker and the containers as the running or execution aspect of Docker.

A Docker container is:

• An image format.

• A set of standard operations.

• An execution environment.

25. **What is a Docker image?**

A Docker image is made up of filesystems layered over each other. At the base is a boot filesystem, bootfs, which resembles the typical Linux/Unix boot filesystem. A Docker user will probably never interact with the boot filesystem. Indeed, when a container has booted, it is moved into memory, and the boot filesystem is unmounted to free up the RAM used by the initrd disk image.

26. **Difference between Docker Vs VM (Virtual Machine).**

Virtual Machines	Docker Containers
Need more resources	Less resources are used
Process isolation is done at the hardware level	Process Isolation is done at Operating System-level
Separate Operating System for each VM	Operating System resources can be shared within Docker
VMs can be customized	Custom container setup is easy
Takes time to create a Virtual Machine	The creation of docker is very quick
Booting takes minutes	Booting is done within seconds.

27. **What is a Docker container?**

This is a very important question so just make sure you don't deviate from the topic and I will advise you to follow the below mentioned format:

- Docker containers include the application and all of its dependencies, but share the kernel with other containers, running as isolated processes in user space on the host operating system. Docker containers are not tied to any specific infrastructure: they run on any computer, on any infrastructure, and in any cloud.
- Now explain how to create a Docker container, Docker containers can be created by either creating a Docker image and then running it or you can use Docker images that are present on the Dockerhub.
- Docker containers are basically runtime instances of Docker images.

28. **What is a Docker hub?**

Docker hub is a cloud-based registry service that allows you to link to code repositories, build your images and test them, store manually pushed images, and link to the Docker cloud so you can deploy images to your hosts. It provides a centralized resource for container image discovery, distribution and change management, user and team collaboration, and workflow automation throughout the development pipeline.

29. **What command can you run to export a docker image as an archive?**

This can be done using the docker save command and the syntax is: docker save -o <exported_name>.tar <container-name>

30. **What command can be run to import a pre-exported Docker image into another Docker host?**

This can be done using the docker load command and the syntax is docker load -i <export_image_name>.tar

31. **Can a paused container be removed from Docker?**

No, it is not possible! A container MUST be in the stopped state before we can remove it.

UNIT IV

CLOUD DEPLOYMENT ENVIRONMENT

Google App Engine – Amazon AWS – Microsoft Azure; Cloud Software Environments – Eucalyptus– OpenStack.

GOOGLE App Engine:

1. **Discuss in detail about the Google App engine and its architecture(or)Discuss in detail about GAE Applications. Nov/Dec 2020(Or)Explain the functional modules of GAE with an example(May-2022)(Or) Demonstrate the programming environment of Google APP Engine.(May-2023)**

Google App Engine

Google App Engine (GAE) is a Platform-as-a-Service cloud computing model that supports many programming languages.

GAE is a scalable runtime environment mostly devoted to execute Web applications. In fact, it allows developers to integrate third-party frameworks and libraries with the infrastructure still being managed by Google.

It allows developers to use readymade platform to develop and deploy web applications using development tools, runtime engine, databases and middleware solutions. It supports languages like Java, Python, .NET, PHP, Ruby, Node.js and Go in which developers can write their code and deploy it on available google infrastructure with the help of Software Development Kit (SDK).

In GAE, SDKs are required to set up your computer for developing, deploying, and managing your apps in App Engine. GAE enables users to run their applications on a large number of data centers associated with Google's search engine operations. Presently, Google App Engine uses fully managed, serverless platform that allows to choose from several popular languages, libraries, and frameworks to develop user applications and then uses App Engine to take care of provisioning servers and scaling application instances based on demand. The functional architecture of the Google cloud platform for app engine is shown in Fig. 4.1.

The infrastructure for google cloud is managed inside datacenter. All the cloud services and applications on Google runs through servers inside datacenter. Inside each data center, there are thousands of servers forming different clusters. Each cluster can run multipurpose servers.

The infrastructure for GAE composed of four main components like Google File System (GFS), MapReduce, BigTable, and Chubby. The GFS is used for storing large amounts of data on google storage clusters. The MapReduce is used for application program development with data processing on large clusters. Chubby is used as a distributed application locking services while BigTable offers a storage service for accessing structured as well as unstructured data. In this architecture, users can interact with Google applications via the web interface provided by each application.

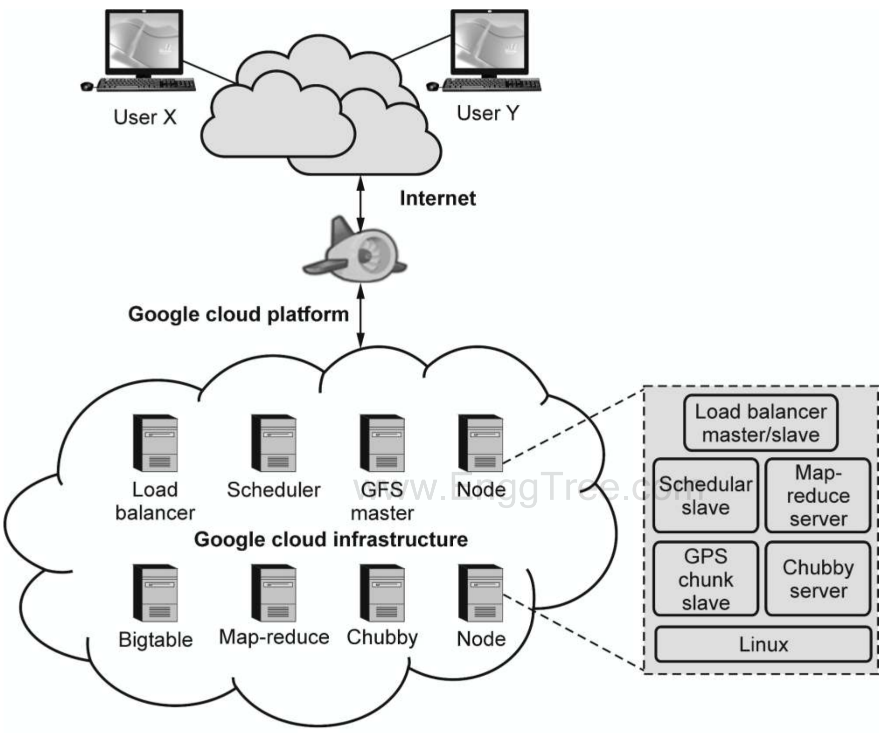


Fig. 4.1 : Functional architecture of the Google cloud platform for app engine

The GAE platform comprises five main components like

- Application runtime environment offers a platform that has built-in execution engine for scalable web programming and execution.
- Software Development Kit (SDK) for local application development and deployment over google cloud platform.
- Datastore to provision object-oriented, distributed, structured data storage to store application and data. It also provides secures data management operations based on BigTable techniques.

- Admin console used for easy management of user application development and resource management
- GAE web service for providing APIs and interfaces.

Programming Environment for Google App Engine

The Google provides programming support for its cloud environment, that is, Google Apps Engine, through Google File System (GFS), Big Table, and Chubby. The following sections provide a brief description about GFS, Big Table, Chubby and Google APIs.

The Google File System (GFS)

Google has designed a distributed file system, named GFS, for meeting its exacting demands off processing a large amount of data. Most of the objectives of designing the GFS are similar to those of the earlier designed distributed systems. Some of the objectives include availability, performance, reliability, and scalability of systems. GFS has also been designed with certain challenging assumptions that also provide opportunities for developers and researchers to achieve these objectives. Some of the assumptions are listed as follows :

a) Automatic recovery from component failure on a routine basis

b) Efficient storage support for large - sized files as a huge amount of data to be processed is stored in these files. Storage support is provided for small - sized files without requiring any optimization for them.

c) With the workloads that mainly consist of two large streaming reads and small random reads, the system should be performance conscious so that the small reads are made steady rather than going back and forth by batching and sorting while advancing through the file.

d) The system supports small writes without being inefficient, along with the usual large and sequential writes through which data is appended to files.

e) Semantics that are defined well are implemented.

f) Atomicity is maintained with the least overhead due to synchronization.

g) Provisions for sustained bandwidth is given priority rather than a reduced latency. Google takes the aforementioned assumptions into consideration, and supports its cloud platform, Google Apps Engine, through GFS. Fig. 4.2 shows the architecture of the GFS clusters.

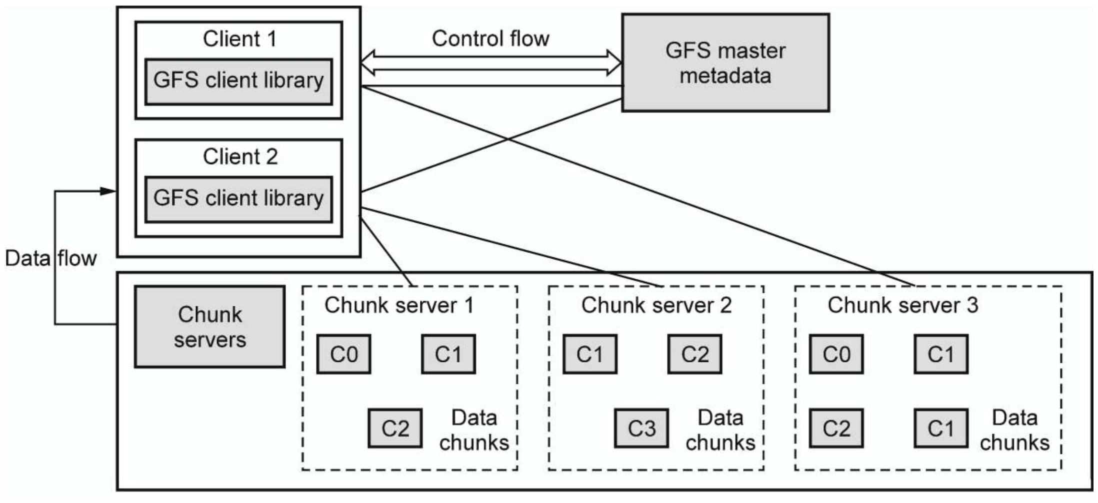


Fig. 4.2 : Architecture of GFS clusters

GFS provides a file system interface and different APIs for supporting different file operations such as *create* to create a new file instance, *delete* to delete a file instance, *open* to open a named file and return a handle, *close* to close a given file specified by a handle, *read* to read data from a specified file and *write* to write data to a specified file.

A single GFS Master and three chunk servers are serving to two clients comprise a GFS cluster. These clients and servers, as well as the Master, are Linux machines, each running a server process at the user level. These processes are known as user-level server processes.

Applications contain a specific file system, Application Programming Interface (APIs) that are executed by the code that is written for the GFS client. Further, the communication with the GFS Master and chunk servers are established for performing the read and write operations on behalf of the application.

The clients interact with the Master only for metadata operations. However, data-bearing communications are forwarded directly to chunk servers. POSIX API, a feature that is common to most of the popular file systems, is not included in GFS, and therefore, Linux vnode layer hook-in is not required.

Clients or servers do not perform the caching of file data. Due to the presence of the streamed workload, caching does not benefit clients, whereas caching by servers has the least consequence as a buffer cache that already maintains a record for frequently requested files locally.

The GFS provides the following features :

- Large - scale data processing and storage support
- Normal treatment for components that stop responding
- Optimization for large-sized files (mostly appended concurrently and read sequentially)
- Fault tolerance by constant monitoring, data replication, and automatic recovering
- Data corruption detections at the disk or Integrated Development Environment
- (IDE) subsystem level through the checksum method
- High throughput for concurrent readers and writers
- Simple designing of the Master that is centralized and not bottlenecked

GFS provides caching for the performance and scalability of a file system and logging for debugging and performance analysis.

Big Table

Googles Big table is a distributed storage system that allows storing huge volumes of structured as well as unstructured data on storage mediums.

Google created Big Table with an aim to develop a fast, reliable, efficient and scalable storage system that can process concurrent requests at a high speed.

Millions of users access billions of web pages and many hundred TBs of satellite images. A lot of semi-structured data is generated from Google or web access by users.

This data needs to be stored, managed, and processed to retrieve insights. This required data management systems to have very high scalability.

Google's aim behind developing Big Table was to provide a highly efficient system for managing a huge amount of data so that it can help cloud storage services.

It is required for concurrent processes that can update various data pieces so that the most recent data can be accessed easily at a fast speed. The design requirements of Big Table are as follows :

1. High speed

2. Reliability
3. Scalability
4. Efficiency
5. High performance
6. Examination of changes that take place in data over a period of time.

Big Table is a popular, distributed data storage system that is highly scalable and self-managed. It involves thousands of servers, terabytes of data storage for in-memory operations, millions of read/write requests by users in a second and petabytes of data stored on disks. Its self-managing services help in dynamic addition and removal of servers that are capable of adjusting the load imbalance by themselves.

It has gained extreme popularity at Google as it stores almost all kinds of data, such as Web indexes, personalized searches, Google Earth, Google Analytics, and Google Finance. It contains data from the Web is referred to as a Web table. The generalized architecture of Big table is shown in Fig. 4.3

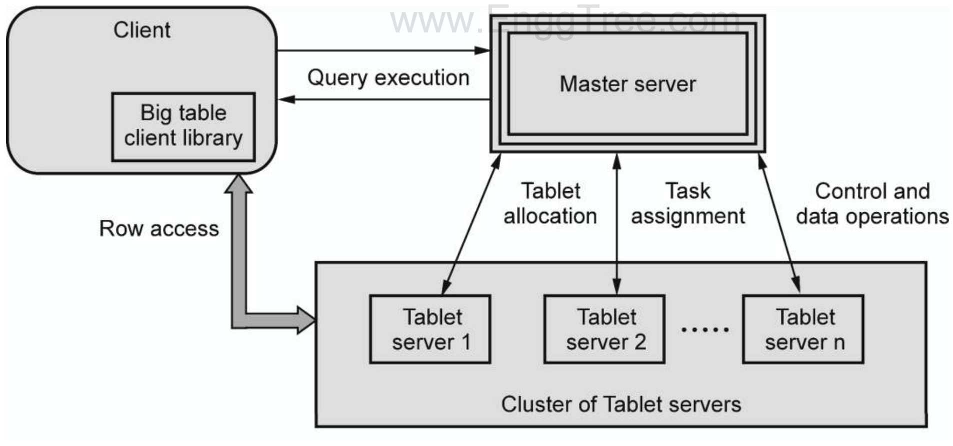


Fig. 4..3 : Generalized architecture of Big table

It is composed of three entities, namely Client, Big table master and Tablet servers. Big tables are implemented over one or more clusters that are similar to GFS clusters. The client application uses libraries to execute Big table queries on the master server. Big table is initially broken up into one or more slave servers called tablets for the execution of secondary tasks. Each tablet is 100 to 200 MB in size.

The master server is responsible for allocating tablets to tasks, clearing garbage collections and monitoring the performance of tablet servers. The master server splits tasks and executes them over tablet servers. The master server is also responsible for maintaining a centralized view of the system to support optimal placement and load- balancing decisions.

It performs separate control and data operations strictly with tablet servers. Upon granting the tasks, tablet servers provide row access to clients. Fig. 4.6.3 shows the structure of Big table :

Big Table is arranged as a sorted map that is spread in multiple dimensions and involves sparse, distributed, and persistence features. The Big Table's data model primarily combines three dimensions, namely *row, column, and time*stamp. The first two dimensions are string types, whereas the time dimension is taken as a 64-bit integer. The resulting combination of these dimensions is a string type.

Each row in Big table has an associated row key that is an arbitrary string of up to

64 KB in size. In Big Table, a row name is a string, where the rows are ordered in a lexicological form. Although Big Table rows do not support the relational model, they offer atomic access to the data, which means you can access only one record at a time. The rows contain a large amount of data about a given entity such as a web page. The row keys represent URLs that contain information about the resources that are referenced by the URLs.

The other important dimension that is assigned to Big Table is a timestamp. In Big table, the multiple versions of data are indexed by timestamp for a given cell. The timestamp is either related to real-time or can be an arbitrary value that is assigned by a programmer. It is used for storing various data versions in a cell.

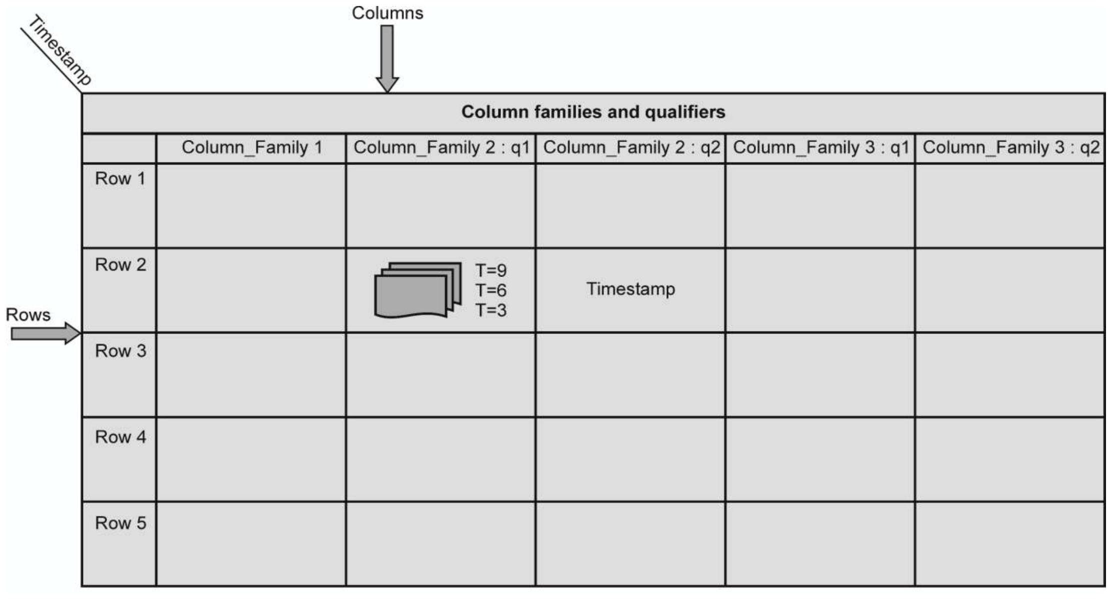


Fig. 4.4 : Structure of Big table

By default, any new data that is inserted into Big Table is taken as current, but you can explicitly set the timestamp for any new write operation in Big Table. Timestamps provide the Big Table lookup option that returns the specified number of the most recent values. It can be used for marking the attributes of the column families.

The attributes either retain the most recent values in a specified number or keep the values for a particular time duration.

Big Table supports APIs that can be used by developers to perform a wide range of operations such as metadata operations, read/write operations, or modify/update operations. The commonly used operations by APIs are as follows:

- Creation and deletion of tables
- Creation and deletion of column families within tables
- Writing or deleting cell values
- Accessing data from rows
- Associate metadata such as access control information with tables and column families

The functions that are used for atomic write operations are as follows :

- Set () is used for writing cells in a row.

- DeleteCells () is used for deleting cells from a row.
- DeleteRow() is used for deleting the entire row, i.e., all the cells from a row are deleted.

It is clear that Big Table is a highly reliable, efficient, and fan system that can be used for storing different types of semi-structured or unstructured data by users.

Chubby

Chubby is the crucial service in the Google infrastructure that offers storage and coordination for other infrastructure services such as GFS and Bigtable. It is a coarse - grained distributed locking service that is used for synchronizing distributed activities in an asynchronous environment on a large scale. It is used as a name service within Google and provides reliable storage for file systems along with the election of coordinator for multiple replicas. The Chubby interface is similar to the interfaces that are provided by
distributed systems with advisory locks. However, the aim of designing Chubby is to provide reliable storage with consistent availability.

It is designed to use with loosely coupled distributed systems that are connected in a high-speed network and contain several small-sized machines. The lock service enables the synchronization of the activities of clients and permits the clients to reach a consensus about the environment in which they are placed. Chubby's main aim is to efficiently handle a large set of clients by providing them a highly reliable and available system. Its other important characteristics that include throughput and storage capacity are secondary. Fig. 4.5 shows the typical structure of a Chubby system :

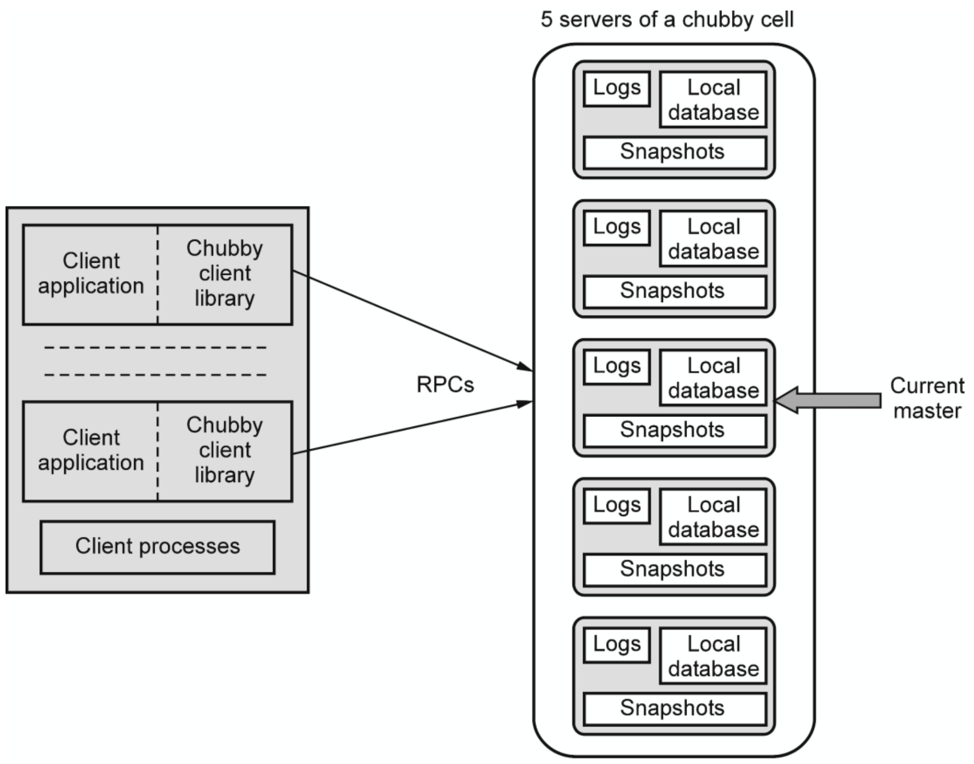


Fig. 4.5: Structure of a Chubby system

The chubby architecture involves two primary components, namely server and client library. Both the components communicate through a Remote Procedure Call (RPC). However, the library has a special purpose, i.e., linking the clients against the chubby cell. A Chubby cell contains a small set of servers. The servers are also called replicas, and usually, five servers are used in every cell. The Master is elected from the five replicas through a distributed protocol that is used for consensus. Most of the replicas must vote
for the Master with the assurance that no other Master will be elected by replicas that have once voted for one Master for a duration. This duration is termed as a Master lease.

Chubby supports a similar file system as Unix. However, the Chubby file system is simpler than the Unix one. The files and directories, known as nodes, are contained in the Chubby namespace. Each node is associated with different types of metadata. The nodes are opened to obtain the Unix file descriptors known as handles. The specifiers for handles include check digits for preventing the guess handle for clients, handle sequence numbers, and mode information for recreating the lock state when the Master changes. Reader and writer locks are implemented by Chubby using files and directories. While exclusive permission for a lock in the writer mode can be obtained by a single client, there can be any number of clients who share a lock in the reader's mode.

Another important term that is used with Chubby is an event that can be subscribed by clients after the creation of handles. An event is delivered when the action that corresponds to it is completed. An event can be :

a. Modification in the contents of a file

b. Addition, removal, or modification of a child node c. Failing over of the Chubby Master

d. Invalidity of a handle

e. Acquisition of lock by others

f. Request for a conflicting lock from another client

In Chubby, caching is done by a client that stores file data and metadata to reduce the traffic for the reader lock. Although there is a possibility for caching of handles and files locks, the Master maintains a list of clients that may be cached. The clients, due to caching, find data to be consistent. If this is not the case, an error is flagged. Chubby maintains sessions between clients and servers with the help of a keep-alive message, which is required every few seconds to remind the system that the session is still active.

If the server failure has indeed occurred, the Master does not respond to a client about the keep-alive message in the local lease timeout. This incident sends the session in jeopardy. It can be recovered in a manner as explained in the following points:

- The cache needs to be cleared.
- The client needs to wait for a grace period, which is about 45 seconds.
- Another attempt is made to contact the Master.

If the attempt to contact the Master is successful, the session resumes and its jeopardy is over. However, if this attempt fails, the client assumes that the session is lost. Fig.4.6 shows the case of the failure of the Master :

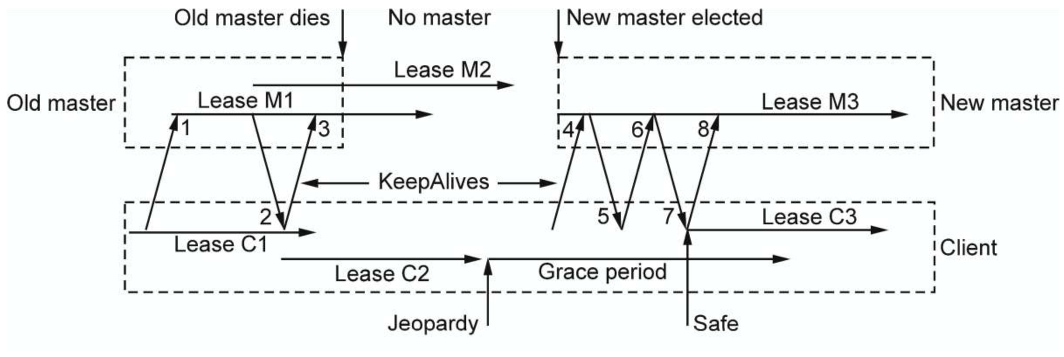


Fig. 4.6.: Case of failure of Master server

Chubby offers a decent level of scalability, which means that there can be any (unspecified) number of the Chubby cells. If these cells are fed with heavy loads, the lease timeout increases. This increment can be anything between 12 seconds and 60 seconds. The data is fed in a small package and held in the Random-Access Memory (RAM) only. The Chubby system also uses partitioning mechanisms to divide data into smaller packages. All of its excellent services and applications included, Chubby has proved to be a great innovation when it comes to storage, locking, and program support services.

The Chubby is implemented using the following APIs :

1. Creation of handles using the open() method
2. Destruction of handles using the close() method

The other important methods include GetContentsAndStat(), GetStat(), ReadDir(), SetContents(), SetACl(), Delete(), Acquire(), TryAcquire(), Release(), GetSequencer(), SetSequencer(), and CheckSequencer(). The commonly used APIs in chubby are listed in Table 4.7 :

API	Descriptio
Open	Opens the file or directory and returns a handle
Close	Closes the file or directory and returns the associated
Delete	Deletes the file or directory
ReadDir	Returns the contents of a directory
SetContents	Writes the contents of a file
GetStat	Returns the metadata
GetContentsAndSt	Writes the file contents and return metadata associated
Acquire	Acquires a lock on a file

Release	Releases a lock on a file

Table 4.7 : APIs in Chubby

Google APIs

Google developed a set of Application Programming Interfaces (APIs) that can be used to communicate with Google Services. This set of APIs is referred as Google APIs. and their integration to other services. They also help in integrating Google Services to other services. Google App Engine help in deploying an API for an app while not being aware about its infrastructure. Google App Engine also hosts the endpoint APIs which are created by Google Cloud Endpoints. A set of libraries, tools, and capabilities that can be used to generate client libraries and APIs from an App Engine application is known as Google Cloud Endpoints. It eases the data accessibility for client applications. We can also save the time of writing the network communication code by using Google Cloud Endpoints that can also generate client libraries for accessing the backend API.

AWS:

2. **Explain in detail about AWS EC2 and EB with an example.**

Programming on Amazon EC2

- ✓ Amazon was the first company to introduce VMs in application hosting. Customers can rent VMs instead of physical machines to run their own applications. By using VMs, customers can load any software of their choice.
- ✓ The elastic feature of such a service is that a customer can create, launch, and terminate server instances as needed, paying by the hour for active servers. Amazon provides several types of preinstalled VMs.
- ✓ Instances are often called Amazon Machine Images (AMIs) which are preconfigured with operating systems based on Linux or Windows, and additional software.

Table 6.12 defines three types of AMI. Figure 6.24 shows an execution environment. AMIs are the templates for instances, which are running VMs. The workflow to create a VM is

Create an AMI → Create Key Pair → Configure Firewall → Launch (6.3)

This sequence is supported by public, private, and paid AMIs shown in Figure 6.24. The AMIs are formed from the virtualized compute, storage, and server resources shown at the bottom of Figure 6.23.

Table 6.12 Three Types of AMI

Image Type	AMI Definition
Private AMI	Images created by you, which are private by default. You can grant access to other users to launch your private images.
Public AMI	Images created by users and released to the AWS community, so anyone can launch instances based on them and use them any way they like. AWS lists all public images at http://developer.amazonwebservices.com/connect/kbcategory.jspa?categoryID=171.
Paid QAMI	You can create images providing specific functions that can be launched by anyone willing to pay you per each hour of usage on top of Amazon's charges.

- ✓ Standard instances are well suited for most applications.
- ✓ Micro instances provide a small number of consistent CPU resources and allow you to burst CPU capacity when additional cycles are available. They are well suited for lower throughput applications and web sites that consume significant compute cycles periodically.
- ✓ High-memory instances offer large memory sizes for high-throughput applications, including database and memory caching applications.
- ✓ High-CPU instances have proportionally more CPU resources than memory (RAM) and are well suited for compute-intensive applications.
- ✓ Cluster compute instances provide proportionally high CPU resources with increased network performance and are well suited for high-performance computing (HPC) applications and other demanding network-bound applications. They use 10 Gigabit Ethernet interconnections.

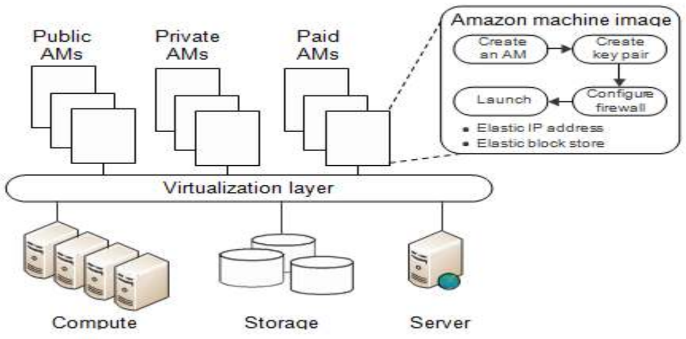


FIGURE 6.23
Amazon EC2 execution environment.

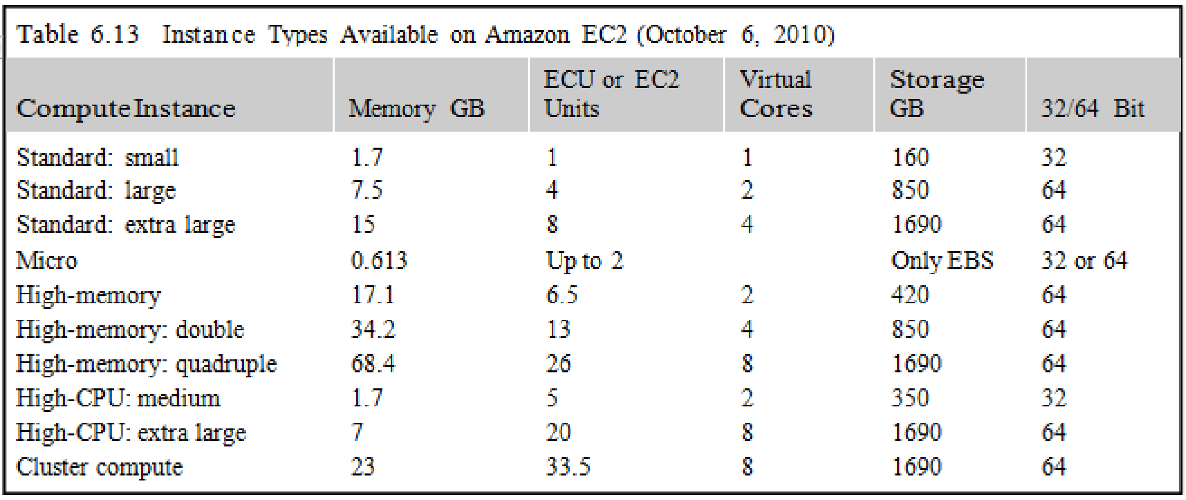


Table 6.13 Instance Types Available on Amazon EC2 (October 6, 2010)

Compute Instance	Memory GB	ECU or EC2 Units	Virtual Cores	Storage GB	32/64 Bit
Standard: small	1.7	1	1	160	32
Standard: large	7.5	4	2	850	64
Standard: extra large	15	8	4	1690	64
Micro	0.613	Up to 2		Only EBS	32 or 64
High-memory	17.1	6.5	2	420	64
High-memory: double	34.2	13	4	850	64
High-memory: quadruple	68.4	26	8	1690	64
High-CPU: medium	1.7	5	2	350	32
High-CPU: extra large	7	20	8	1690	64
Cluster compute	23	33.5	8	1690	64

Amazon Elastic Block Store (EBS) and SimpleDB

- ✓ The Elastic Block Store (EBS) provides the volume block interface for saving and restoring the virtual images of EC2 instances.
- ✓ Traditional EC2 instances will be destroyed after use. The status of EC2 can now be saved in the EBS system after the machine is shut down. Users can use EBS to save persistent data and mount to the running instances of EC2.
- ✓ Note that S3 is "Storage as a Service" with a messaging interface. EBS is analogous to a distributed file system accessed by traditional OS disk access mechanisms. EBS allows you to create storage volumes from 1 GB to 1 TB that can be mounted as EC2 instances.
- ✓ Multiple volumes can be mounted to the same instance. These storage volumes behave like raw, unformatted block devices, with user-supplied device names and a block device interface.
- ✓ You can create a file system on top of Amazon EBS volumes, or use them in any other way you would use a block device (like a hard drive). Snapshots are provided so that the data can be saved incrementally. This can improve performance when saving and restoring data. In terms of pricing, Amazon provides a similar pay-per-use schema as EC2 and S3.

Amazon SimpleDB Service

- ✓ SimpleDB provides a simplified data model based on the relational database data model. Structured data from users must be organized into domains. Each domain can be considered a table. The items are the rows in the table.
- ✓ A cell in the table is recognized as the value for a specific attribute (column name) of the corresponding row. This is similar to a table in a relational database. However, it is possible to assign multiple values to a single cell in the table.

- ✓ This is not permitted in a traditional relational database which wants to maintain data consistency.
- ✓ Many developers simply want to quickly store, access, and query the stored data. SimpleDB removes the requirement to maintain database schemas with strong consistency.
- ✓ SimpleDB is priced at $0.140 per Amazon SimpleDB Machine Hour consumed with the first 25 Amazon SimpleDB Machine Hours consumed per month free (as of October 6, 2010).
- ✓ SimpleDB, like Azure Table, could be called "LittleTable," as they are aimed at managing small amounts of information stored in a distributed table; one could say BigTable is aimed at basic big data, whereas LittleTable is aimed at metadata.

3. Explain in detail about Cloud Storage Providers -S3 with an example.

Simple Storage Service (S3)

Amazon S3 offers a simple web services interface that can be used to store and retrieve any amount of data from anywhere, at any time on the web. It gives any developer access to the same scalable, secure, fast, low - cost data storage infrastructure that Amazon uses to operate its own global website network. S3 is an online backup and storage system. The high - speed data transfer feature known as AWS Import/Export will exchange data to and from AWS using Amazon's own internal network to another portable device.

Amazon S3 is a cloud - based storage system that allows storage of data objects in the range of 1 byte up to 5 GB in a flat namespace. The storage containers in S3 have predefined buckets, and buckets serve the function of a directory, though there is no object hierarchy to a bucket, and the user can save objects to it but not files. Here it is important to note that the concept of a file system is not associated with S3 because file systems are not supported, only objects are stored. In addition to this, the user is not required to mount a bucket, as opposed to a file system. Fig. 4.7 shows an S3 diagrammatically.

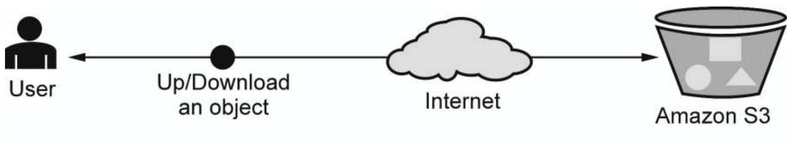


Fig. 4.7 : AWS S3

S3 system allows buckets to be named (Fig. 4.8), but the name must be unique in the S3 namespace across all consumers of AWS. The bucket can be accessed through the S3 web API (with SOAP or REST), which is similar to a normal disk storage system.

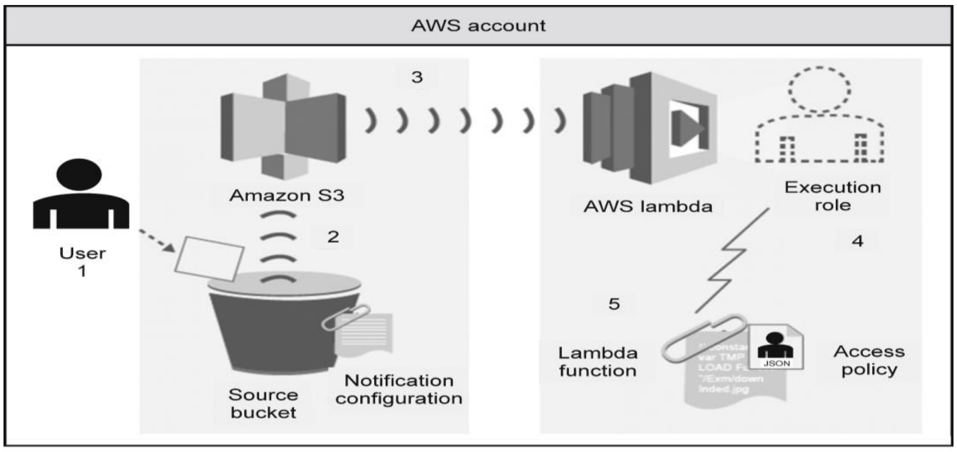


Fig. 4.8: Source bucket

The performance of S3 is limited for use with non-operational functions such as data archiving, retrieval and disk backup. The REST API is more preferred to SOAP API because it is easy to work with large binary objects in REST.

Amazon S3 offers large volumes of reliable storage with high protection and low bandwidth access. S3 is most ideal for applications that need storage archives. For example, S3 is used by large storage sites that share photos and images.

The APIs to manage the bucket has the following features :

- Create new, modify or delete existing buckets.
- Upload or download new objects to a bucket.
- Search and identify objects in buckets.
- Identify metadata associated with objects and buckets.
- Specify where the bucket is stored.
- Provide public access to buckets and objects.

The S3 service can be used by many users as a backup component in a 3-2-1 backup method. This implies that your original data is 1, a copy of your data is 2 and an off-site copy of data is 3. In this method, S3 is the 3rd level of backup. In addition to this, Amazon S3 provides the feature of versioning.

In versioning, every version of the object stored in an S3 bucket is retained, but for this, the user must enable the versioning feature. Any HTTP or REST operation, namely PUT, POST, COPY or DELETE will create a new object that is stored along with the older version. A GET operation retrieves the new version of the object, but the ability to recover and undo actions are also available. Versioning is a useful method for reserving and archiving data.

Amazon Glacier

Amazon glacier is very low - price online file storage web service which offer secure, flexible and durable storage for online data backup and archiving. This web service is specially designed for those data which are not accessed frequently. That data which is allowed to be retrieved within three to five hours can use amazon glacier service.

You can virtually store any type of data, any format of data and any amount of data using amazon glacier. The file in ZIP and TAR format are the most common type of data stored in amazon glacier.

Some of the common use of amazon glacier are :

- Replacing the traditional tape solutions with backup and archive which can last longer.
- Storing data which is used for the purposes of compliance.

Glacier Vs S3

Both amazon S3 and amazon glacier work almost the same way. However, there are certain important aspects that can reflect the difference between them. Table 6.10.1 shows the comparison of amazon glacier and amazon S3 :

Amazon Glacier	Amazon S3
It supports 40 TB archives	It supports 5 TB objects
It is recognised by archive IDs which are system generated	It can use "friendly" key names

It encrypts the archives automatically	It is optional to encrypt the data
It is extremely low - cost storage	Its cost is much higher than Amazon Glacier

Table 4.9 : Amazon Glacier Vs Amazon S3

You can also use amazon S3 interface for availing the offerings of amazon glacier with no need of learning a new interface. This can be done by utilising Glacier as S3 storage class along with object lifecycle policies.

Azure:

4. **Explain in detail about Azure Architecture and its Components with an example.**

Microsoft Windows Azure(Azure)

In 2008, Microsoft launched a Windows Azure platform to meet the challenges in cloud computing. This platform is built over Microsoft data centers. Figure 4.22 shows the overall architecture of Microsoft's cloud platform. The platform is divided into three major component platforms. Windows Azure offers a cloud platform built on Windows OS and based on Microsoft virtualization technol- ogy. Applications are installed on VMs deployed on the data-center servers. Azure manages all servers, storage, and network resources of the data center. On top of the infrastructure are the var- ious services for building different cloud applications.

Live service Users can visit Microsoft Live applications and apply the data involved across multiple machines concurrently.

- .NET service This package supports application development on local hosts and execution on cloud machines.

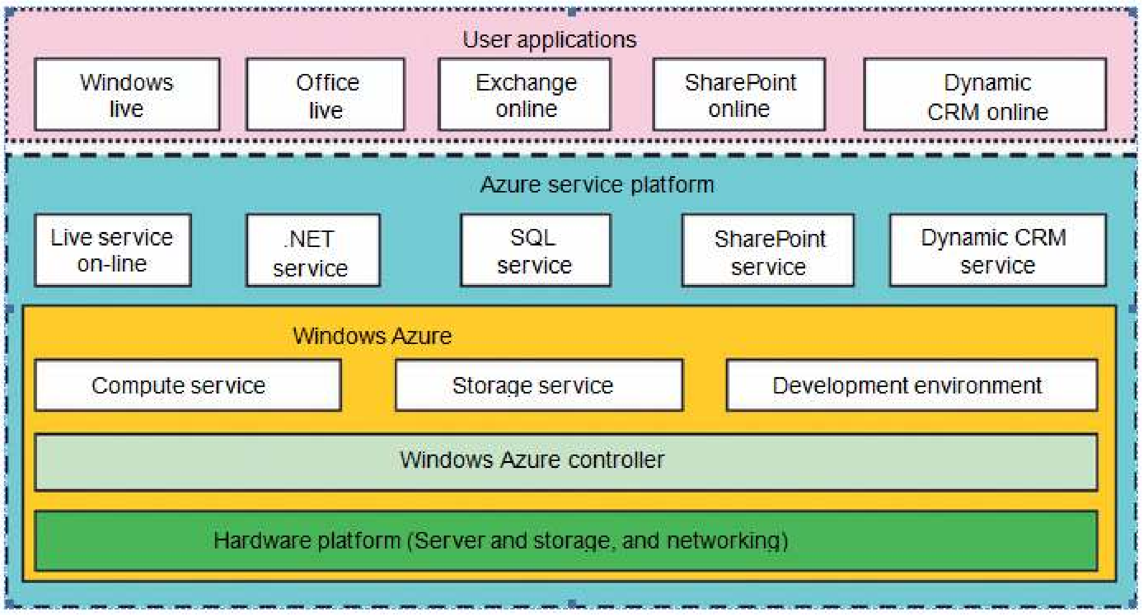


FIGURE 4.22

Microsoft Windows Azure platform for cloud computing.

SQL Azure This function makes it easier for users to visit and use the relational database associated with the SQL server in the cloud.

- SharePoint service This provides a scalable and manageable platform for users to develop their special business applications in upgraded web services.

- Dynamic CRM service This provides software developers a business platform in managing CRM applications in financing, marketing, and sales and promotions.

- ✓ All these cloud services in Azure can interact with traditional Microsoft software applications, such as Windows Live, Office Live, Exchange online, SharePoint online, and dynamic CRM online.
- ✓ The Azure platform applies the standard web communication protocols SOAP and REST. The Azure service applications allow users to integrate the cloud application with other platforms or third-party clouds.
- ✓ You can download the Azure development kit to run a local version of Azure. The powerful SDK allows Azure applications to be developed and debugged on the Windows hosts.

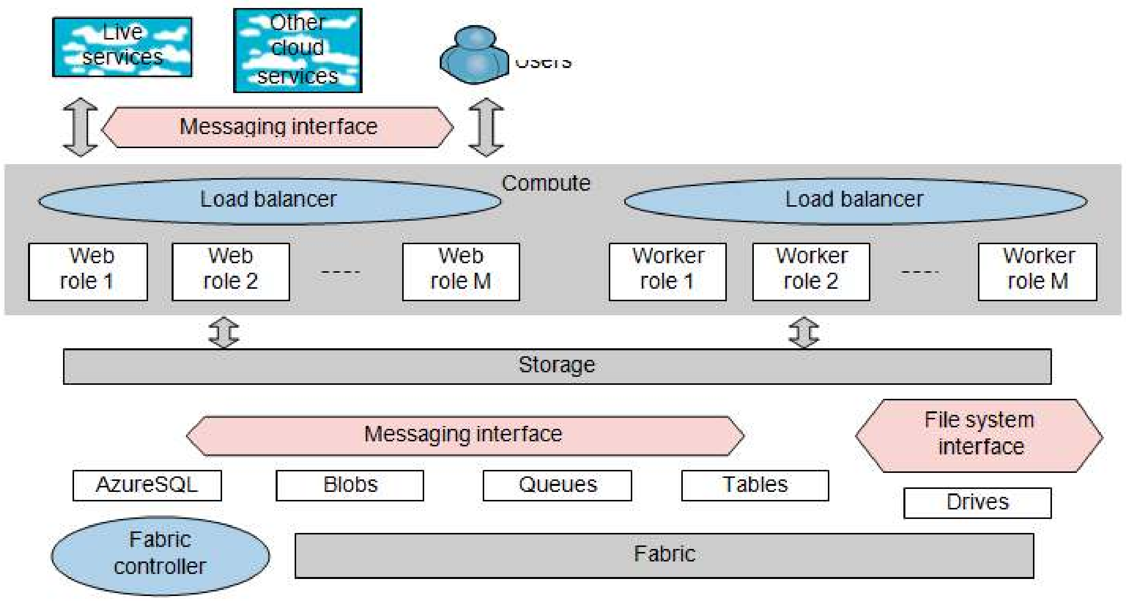


Figure:4.10 Features of the Azure cloud platform.

SQLAzure

Azure offers a very rich set of storage capabilities, as shown in Figure 6.25. All the storage modalities are accessed with REST interfaces except for the recently introduced Drives that are analogous to Amazon EBS discussed in above (AWS Methods), and offer a file system interface as a durable NTFS volume backed by blob storage. The REST interfaces are automatically associated with URLs and all storage is replicated three times for fault tolerance and is guaranteed to be consistent in access.

The basic storage system is built from blobs which are analogous to S3 for Amazon. Blobs are arranged as a three-level hierarchy: Account → Containers → Page or Block Blobs.

Containers are analogous to directories in traditional file systems with the account acting as the root. The block blob is used for streaming data and each such blob is made up as a sequence of blocks of up to 4 MB each, while each block has a 64 byte ID.

Block blobs can be up to 200 GB in size. Page blobs are for random read/write access and consist of an array of pages with a maximum blob size of 1 TB. One can associate metadata with blobs as <name, value> pairs with up to 8 KB per blob.

Azure Tables

The Azure Table and Queue storage modes are aimed at much smaller data volumes. Queues provide reliable message delivery and are naturally used to support work spooling between web and worker roles. Queues consist of an unlimited number of messages which can be retrieved and pro- cessed at least once with an 8 KB limit on message size.

Azure supports PUT, GET, and DELETE message operations as well as CREATE and DELETE for queues. Each account can have any number of Azure tables which consist of rows called entities and columns called properties.

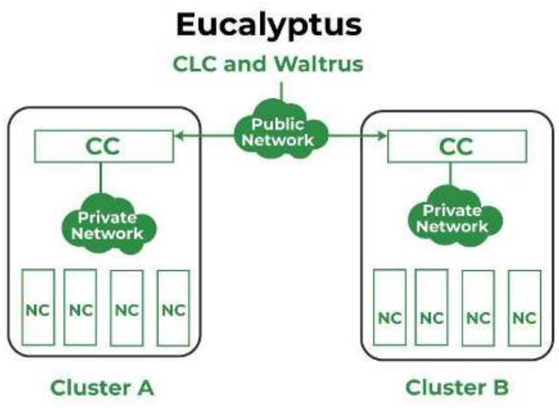
There is no limit to the number of entities in a table and the technology is designed to scale well to a large number of entities stored on distributed computers. All entities can have up to 255 general properties which are <name, type, value> triples.

Two extra properties, PartitionKey and RowKey, must be defined for each entity, but otherwise, there are no constraints on the names of properties—this table is very flexible! RowKey is designed to give each entity a unique label while PartitionKey is designed to be shared and entities with the same PartitionKey are stored next to each other; a good use of PartitionKey can speed up search performance. An entity can have, at most, 1 MB storage; if you need large value sizes, just store a link to a blob store in the Table property value. ADO.NET and LINQ support table queries.

Cloud Software Environments:

5. Explain in detail about Eucalyptus and its operation.

Eucalyptus is a Linux-based open-source software architecture for cloud computing and also a storage platform that implements Infrastructure a Service (IaaS). It provides quick and efficient computing services. Eucalyptus was designed to provide services compatible with Amazon's EC2 cloud and Simple Storage Service(S3).



Eucalyptus Architecture

Eucalyptus CLIs can handle Amazon Web Services and their own private instances. Clients have the independence to transfer cases from Eucalyptus to Amazon Elastic Cloud. The virtualization layer oversees the Network, storage, and Computing. Occurrences are isolated by hardware virtualization.

Important Features are:-

1. **Images**: A good example is the Eucalyptus Machine Image which is a module software bundled and uploaded to the Cloud.
2. **Instances**: When we run the picture and utilize it, it turns into an instance.
3. **Networking**: It can be further subdivided into three modes: Static mode(allocates IP address to instances), System mode (assigns a MAC address and imputes the instance's network interface to the physical network via NC), and Managed mode (achieves local network of instances).
4. **Access Control:** It is utilized to give limitations to clients.
5. **Elastic Block Storage**: It gives block-level storage volumes to connect to an instance.
6. **Auto-scaling and Load Adjusting**: It is utilized to make or obliterate cases or administrations dependent on necessities.

Components of Architecture

- **Node Controller** is the lifecycle of instances running on each node. Interacts with the operating system, hypervisor, and Cluster Controller. It controls the working of VM instances on the host machine.

- **Cluster Controller** manages one or more Node Controller and Cloud Controller simultaneously. It gathers information and schedules VM execution.
- **Storage Controller (Walrus)** Allows the creation of snapshots of volumes. Persistent block storage over VM instances. Walrus Storage Controller is a simple file storage system. It stores images and snapshots. Stores and serves files using S3(Simple Storage Service) APIs.
- **Cloud Controller** Front-end for the entire architecture. It acts as a Complaint Web Services to client tools on one side and interacts with the rest of the components on the other side.

Operation Modes Of Eucalyptus

- **Managed Mode**: Numerous security groups to users as the network is large. Each security group is assigned a set or a subset of IP addresses. Ingress rules are applied through the security groups specified by the user. The network is isolated by VLAN between Cluster Controller and Node Controller. Assigns two IP addresses on each virtual machine.
- **Managed (No VLAN) Node:** The root user on the virtual machine can snoop into other virtual machines running on the same network layer. It does not provide VM network isolation.
- **System Mode:** Simplest of all modes, least number of features. A MAC address is assigned to a virtual machine instance and attached to Node Controller's bridge Ethernet device.
- **Static Mode**: Similar to system mode but has more control over the assignment of IP address. MAC address/IP address pair is mapped to static entry within the DHCP server. The next set of MAC/IP addresses is mapped.

Advantages Of The Eucalyptus Cloud

1. Eucalyptus can be utilized to benefit both the eucalyptus private cloud and the eucalyptus public cloud.
2. Examples of Amazon or Eucalyptus machine pictures can be run on both clouds.
3. Its API is completely similar to all the Amazon Web Services.
4. Eucalyptus can be utilized with DevOps apparatuses like Chef and Puppet.
5. Although it isn't as popular yet but has the potential to be an alternative to OpenStack and CloudStack.
6. It is used to gather hybrid, public and private clouds.
7. It allows users to deliver their own data centers into a private cloud and hence, extend the services to other organizations.

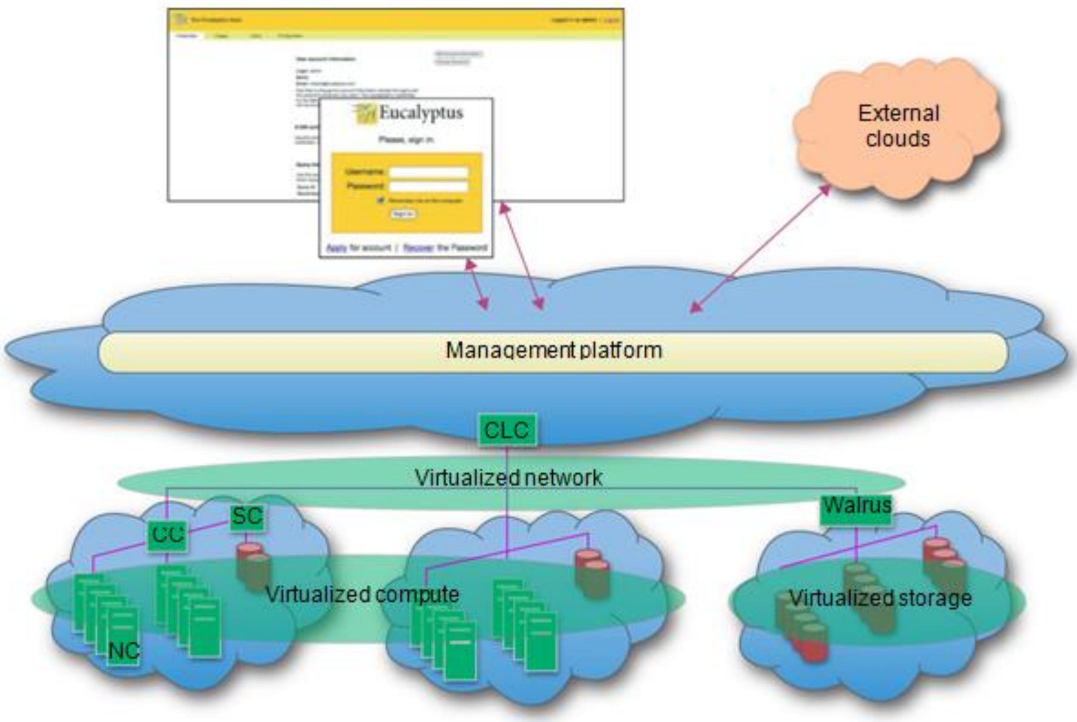


Fig:4.10The Eucalyptus architecture for VM image management.

Nimbus:

Nimbus Nimbus is a toolkit that, once included in the collection, provides infrastructure such as a cloud of service to its client through the WSRF-based web service APIs or Amazon EC2 WSDL. Nimbus is a free and open source software, subject to the requirements of the Apache License, version 2.

Nimbus supports both the Xen and KVM hypervisors as well as the portable device organizers Portable Batch System and Oracle Grid Engine. Allows the submission of customized visual clusters for content. It is adjustable in terms of planning, network rental, and accounting usage.

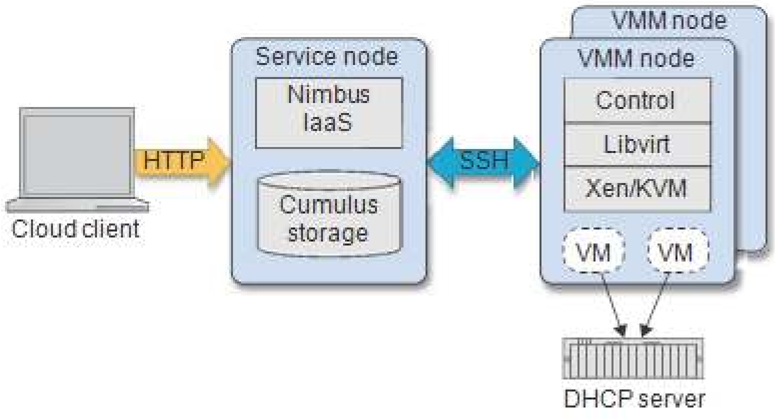


Fig:4.11 Nimbus cloud infrastructure.

OPEN STACK:

6. **Give the anatomy/pictorial representation of open stack in cloud environment)(Nov/Dec 2021)(May-2022)(or) Open Stack is Open Cloud Ecosystem.(May-2023)**

Open Stack

OpenStack is an open - source cloud operating system that is increasingly gaining admiration among data centers. This is because OpenStack provides a cloud computing platform to handle enormous computing, storage, database and networking resources in a data center. In simple way we can say, OpenStack is an opensource highly scalable cloud computing platform that provides tools for developing private, public or hybrid clouds, along with a web interface for users to access resources and admins to manage those resources.

Put otherwise, OpenStack is a platform that enables potential cloud providers to create, manage and bill their custom-made VMs to their future customers. OpenStack is free and open, which essentially means that everyone can have access to its source code and can suggest or make changes to it and share it with the OpenStack community. OpenStack is an open-source and freely available cloud computing platform that enables its users to create, manage and deploy virtual machines and other instances. Technically, OpenStack provides Infrastructure-as-a-Service (IaaS) to its users to enable them to manage virtual private servers in their data centers.

OpenStack provides the required software tools and technologies to abstract the underlying infrastructure to a uniform consumption model. Basically, OpenStack allows various organisations to provide cloud services to the user community by leveraging the organization's pre-existing infrastructure. It also provides options for scalability so that resources can be scaled whenever organisations need to add more resources without hindering the ongoing processes.

The main objective of OpenStack is to provide a cloud computing platform that is :

- Global
- Open-source
- Freely available
- Easy to use
- Highly and easily scalable
- Easy to implement

- Interoperable

OpenStack is for all. It satisfies the needs of users, administrators and operators of private clouds as well as public clouds. Some examples of open-source cloud platforms already available are Eucalyptus, OpenNebula, Nimbus, CloudStack and OpenStack, which are used for infrastructure control and are usually implemented in private clouds.

Components of OpenStack

OpenStack consists of many different components. Because OpenStack cloud is open - source, developers can add components to benefit the OpenStack community. The following are the core components of OpenStack as identified by the OpenStack community:

- **Nova :** This is one of the primary services of OpenStack, which provides numerous tools for the deployment and management of a large number of virtual machines. Nova is the compute service of OpenStack.
- **Swift :** Swift provides storage services for storing files and objects. Swift can be equated with Amazon's Simple Storage System (S3).
- **Cinder :** This component provides block storage to Nova Virtual Machines. Its working is similar to a traditional computer storage system where the computer is able to access specific locations on a disk drive. Cinder is analogous to AWS's EBS.
- **Glance :** Glace is OpenStack's image service component that provides virtual templates (images) of hard disks. These templates can be used for new VMs. Glance may use either Swift or flat files to store these templates.
- **Neutron (formerly known as Quantum) :** This component of OpenStack provides Networking-as-a- Service, Load-Balancer-as-a-Service and Firewall- as-a-Service. It also ensures communication between other components.
- **Heat :** It is the orchestration component of OpenStack. It allows users to manage infrastructural needs of applications by allowing the storage of requirements in files.
- **Keystone :** This component provides identity management in OpenStack
- **Horizon :** This is a dashboard of OpenStack, which provides a graphical interface.
- **Ceilometer :** This component of OpenStack provisions meters and billing models for users of the cloud services. It also keeps an account of the resources used by each individual user of the OpenStack cloud. Let us also discuss some of the non- core components of OpenStack and their offerings.

- **Trove :** Trove is a component of OpenStack that provides Database-as-a- service. It provisions relational databases and big data engines.
- **Sahara :** This component provisions Hadoop to enable the management of data processors.
- **Zaqar :** This component allows messaging between distributed application components.
- **Ironic :** Ironic provisions bare-metals, which can be used as a substitute to VMs.

The basic architectural components of OpenStack, shown in Fig:4.12, includes its core and optional services/ components. The optional services of OpenStack are also known as Big Tent services, and OpenStack can be used without these components or they can be used as per requirement.

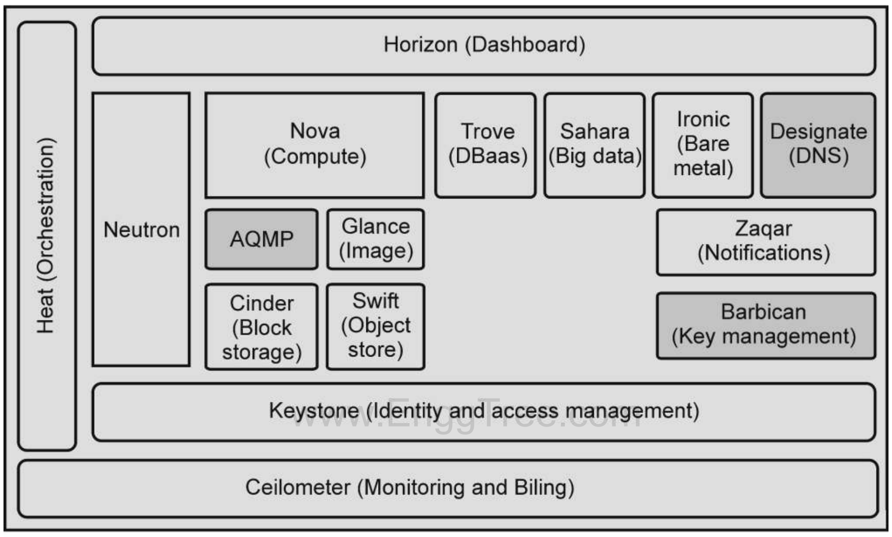


Fig. 4.12 : Components of open stack architecture

We have already discussed the core services and the four optional services. Let us now discuss the rest of the services.

- **Designate :** This component offers DNS services analogous to Amazon's Route 53.
  The following are the subsystems of Designate :
  Mini DNS Server
  Pool Manager
  Central Service and APIs
- **Barbican :** Barbican is the key management service of OpenStack that is comparable to KMS from AWS. This provides secure storage, retrieval, and provisioning and management of various types of secret data, such as keys, certificates, and even binary data.
- **AMQP :** AMQP stands for Advanced Message Queue Protocol and is a messaging mechanism

used by OpenStack. The AQMP broker lies between two components of Nova and enables communication in a slackly coupled fashion.

Further, OpenStack uses two architectures - Conceptual and Logical, which are discussed in the next section.

Features and Benefits of OpenStack

OpenStack helps build cloud environments by providing the ability to integrate various technologies of your choice. Apart from the fact that OpenStack is open-source, there are numerous benefits that make it stand out. Following are some of the features and benefits of OpenStack Cloud :

- **Compatibility :** OpenStack supports both private and public clouds and is very easy to deploy and manage. OpenStack APIs are supported in Amazon Web Services. The compatibility eliminates the need for rewriting applications for AWS, thus enabling easy portability between public and private clouds.
- **Security :** OpenStack addresses the security concerns, which are the top- most concerns for most organisations, by providing robust and reliable security systems.
- **Real-time Visibility :** OpenStack provides real-time client visibility to administrators, including visibility of resources and instances, thus enabling administrators and providers to track what clients are requesting for.
- **Live Upgrades :** This feature allows upgrading services without any downtime. Earlier, for upgradations, the was a need for shutting-down complete systems, which resulted in loss of performance. Now, OpenStack has enabled upgrading systems while they are running by requiring only individual components to shut- down.

Apart from these, OpenStack offers other remarkable features, such as networking, compute, Identity Access Management, orchestration, etc.

Conceptual OpenStack Architecture

Fig. 4.13, depicting a magnified version of the architecture by showing relationships among different services and between the services and VMs. This expanded representation is also known as the Conceptual architecture of OpenStack.

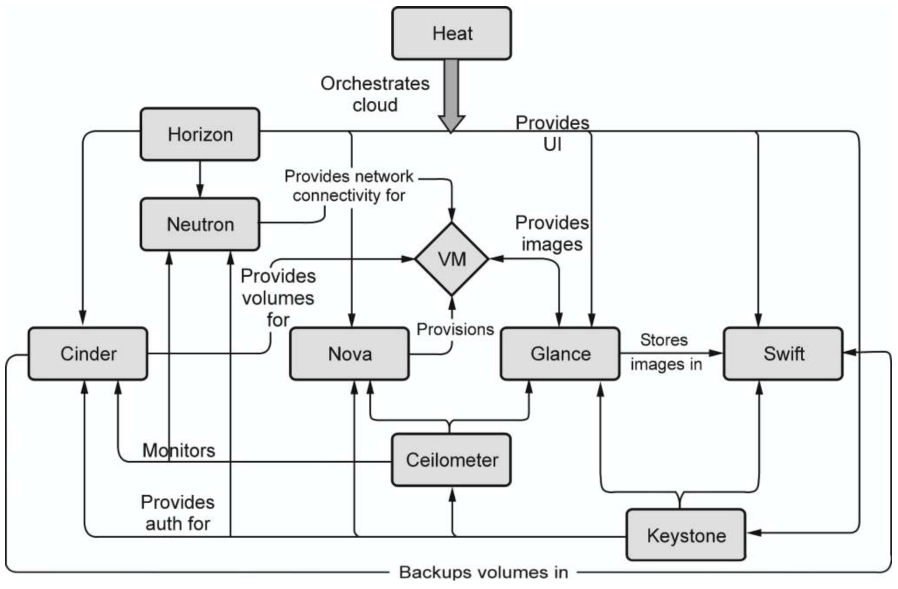


Fig. 4.13 : Conceptual architecture of OpenStack

From Fig. 5.7.2, we can see that every service of OpenStack depends on other services within the systems, and all these services exist in a single ecosystem working together to produce a virtual machine. Any service can be turned on or off depending on the VM required to be produced. These services communicate with each other through APIs and in some cases through privileged admin commands.

Let us now discuss the relationship between various components or services specified in the conceptual architecture of OpenStack. As you can see in Figure 4.2, three components, Keystone, Ceilometer and Horizon, are shown on top of the OpenStack platform.

Here, **Horizon** is providing user interface to the users or administrators to interact with underlying OpenStack components or services, **Keystone** is providing authentication to the user by mapping the central directory of users to the accessible OpenStack services, and **Ceilometer** is monitoring the OpenStack cloud for the purpose of scalability, billing, benchmarking, usage reporting and other telemetry services. Inside the OpenStack platform, you can see that various processes are handled by different OpenStack services; **Glance** is registering Hadoop images, providing image services to OpenStack and allowing retrieval and storage of disk images. Glance stores the images in **Swift**, which is responsible for providing reading service and storing data in the form of objects and files. All other OpenStack components also store data in Swift, which also stores data or job binaries. **Cinder**, which offers permanent block storage or volumes to VMs, also stores backup volumes in Swift. **Trove** stores backup databases in Swift and boots databases instances via **Nova**, which is the main computing engine that provides and manages virtual machines using disk images.

Neutron enables network connectivity for VMs and facilitates PXE Network for Ironic that fetches images via Glance. **VMs** are used by the users or administrators to avail and provide the benefits of cloud services. All the OpenStack services are used by VMs in order to provide best services to the users. The infrastructure required for running cloud services is managed by **Heat**, which is the orchestration component of OpenStack that orchestrates clusters and stores the necessarys resource requirements of a cloud application. Here, **Sahara** is used to offer a simple means of providing a data processing framework to the cloud users.

Table 4.14 shows the dependencies of these services.

Code Name	Dependent on	Optional
Nova (Compute)	Keystone, Horizon, Glance	Cinder, Neutron
Swift (Object Storage)	Keystone	-
Cinder (Block Storage)	Keystone	-
Glance (Image Service)	Swift, Keystone, Horizon	-
Neutron (Network)	Keystone, Nova	-
Keystone (Identity)	-	-
Horizon (Dashboard)	Keystone	-

Table 4.14: Service Dependencies

Modes of Operations of OpenStack

OpenStack majorly operates in two modes - single host and multi host. A single host mode of operation is that in which the network services are based on a central server, whereas a multi host operation mode is that in which each compute node has a duplicate copy of the network running on it and the nodes act like Internet gateways that are running on individual nodes. In addition to this, in a multi host operation mode, the compute nodes also individually host floating IPs and security groups. On the other hand, in a single host mode of operation, floating

IPs and security groups are hosted on the cloud controller to enable communication.

Both single host and multi host modes of operations are widely used and have their own set of advantages and limitations. A single host mode of operation has a major limitation that if the cloud controller goes down, it results in the failure of the entire system because instances stop communicating. This is overcome by a multi host operation mode where a copy of the network is provisioned to every node. Whereas, this limitation is overcome by the multi host mode, which requires a unique public IP address for each compute node to enable communication. In case public IP addresses are not available, using the multi host mode is not possible.

PART-A

1. ***Is Google App Engine built over PaaS? Justify.***

   Google App Engine is a Platform as a Service (PaaS) product that provides Web app developers and enterprises with access to Google's scalable hosting and tier 1 Internet service. The App Engine requires that apps be written in Java or Python, store data in Google BigTable and use the Google query language.

2. **What is Google Cloud Platform?**

   Google Cloud Platform (GCP), offered by Google, is a suite of cloud computing services that runs on the same infrastructure that Google uses internally for its end-user products, such as Google Search, Gmail, file storage, and YouTube.Google Cloud Platform provides infrastructure as a service, platform as a service, and serverless computing environments

3. **What is Open stack?**

   OpenStack is a cloud operating system that controls large pools of compute, storage, and networking resources throughout a datacenter, all managed and provisioned through APIs with common authentication mechanisms.

4. **How is Open Stack used in a cloud environment?**

   OpenStack is an open source platform that uses pooled virtual resources to build and manage private and public clouds. The tools that comprise the OpenStack platform, called "projects," handle the core cloud-computing services of compute, networking, storage, identity, and image services.

5. **Identify the development technologies currently supported by AppEngine. Nov/Dec 2020.**

   Google App Engine primarily supports Go, PHP, Java, Python, Node. js, . NET, and Ruby applications, although it can also support other languages via "custom runtimes".

6. **Outline the main Services that are offered by AWS.Nov/Dec 2020.**

   Amazon Web Services offers a broad set of global cloud-based products including compute, storage, databases, analytics, networking, mobile, developer tools, management tools, IoT, security and enterprise applications.

7. **Define GAE.**

   Google App Engine (GAE) is a platform-as-a-service cloud computing model that supports many programming languages. GAE is a scalable runtime environment mostly devoted to execute Web applications. In fact, it allows developers to integrate third-party frameworks and libraries with the infrastructure still being managed by Google. It allows developers to use readymade platform to develop and deploy web applications using development tools, runtime engine, databases and middleware solutions. It supports languages like Java, Python, .NET, PHP, Ruby, Node.js and Go in which developers can write their code and deploy it on available google infrastructure with the help of Software Development Kit (SDK). In GAE, SDKs are required to set up your computer for developing, deploying, and managing your apps in App Engine.

8. **What are the core components of Google app engine architecture ?**

   The infrastructure for GAE composed of four main components like Google File System (GFS), MapReduce, BigTable, and Chubby. The GFS is used for storing large amounts of data on google storage clusters. The MapReduce is used for application program development with data processing on large clusters. Chubby is used as a distributed application locking services while BigTable offers a storage service for accessing structured as well as unstructured data.

9. Enlist the advantages of GFS.

Google has designed a distributed file system, named GFS, for meeting its exacting demands off processing a large amount of data. GFS provides a file system interface and different APIs for supporting different file operations such as create to create a new file instance, delete to delete a file instance, open to open a named file and return a handle, close to close a given file specified by a handle, read to read data from a specified file and write to write data to a specified file.

The advantages of GFS are :

- ❖ Automatic recovery from component failure on a routine basis.
- ❖ Efficient storage support for large - sized files as a huge amount of data to be processed is stored in these files. Storage support is provided for small - sized files without requiring any optimization for them.
- ❖ With the workloads that mainly consist of two large streaming reads and small random reads, the system should be performance conscious so that the small reads are made steady rather than going back and forth by batching and sorting while advancing through the file.
- ❖ The system supports small writes without being inefficient, along with the usual large and sequential writes through which data is appended to files.
- ❖ Semantics that are defined well are implemented.
- ❖ Atomicity is maintained with the least overhead due to synchronization.
- ❖ Provisions for sustained bandwidth is given priority rather than a reduced latency.

10. What is the role of chubby in Google app engine ?

Ans. : Chubby is the crucial service in the Google infrastructure that offers storage and coordination for other infrastructure services such as GFS and Bigtable. It is a coarse - grained distributed locking service that is used for synchronizing distributed activities in an asynchronous environment on a large scale. It is used as a name service within Google and provides reliable storage for file systems along with the election of coordinator for multiple replicas. The Chubby interface is similar to the interfaces that are provided by distributed systems with advisory locks. However, the aim of designing Chubby is to provide reliable storage with consistent availability.

11. What is Openstack ? Enlist its important components.

OpenStack is an open source highly scalable cloud computing platform that provides tools for developing private, public or hybrid clouds, along with a web interface for users to access resources and admins to manage those resources.

The different components of Openstack architecture are :

a. Nova (Compute)

b. Swift (Object storage)

c. Cinder (Block level storage)

d. Neutron (Networking)

e. Glance (Image Management)

f. Keystone (Key management) g. Horizon (Dashboard)

h. Ceilometer (Metering)

i. Heat (Orchestration)

12. Enlist the pros and cons of storage as a service.

The key advantages or pros of storage as a service are given as follows :

- Cost - Storage as a service reduces much of the expense of conventional backup methods, by offering ample cloud storage space at a small monthly charge.
- Invisibility - Storage as a service is invisible, as no physical presence can be seen in its deployment, and therefore does not take up valuable office space.
- Security - In this type of service, data is encrypted both during transmission and during rest, ensuring no unauthorized access to files by the user.
- Automation - Storage as a service makes the time-consuming process of backup easier to accomplish through automation. Users can simply select what and when they want to backup, and the service does the rest of it.
- Accessibility - By using storage as a service, users can access data from smartphones, netbooks to desktops, and so on.
- Syncing - Syncing in storage as a service ensures that your files are updated automatically across all of your devices. This way, the latest version of a user file stored on their desktop is available on your smartphone.
- Sharing - Online storage services make it easy for users to share their data with just a few clicks.
- Collaboration - Cloud storage services are also ideal for collaborative purposes. They allow multiple people to edit and collaborate in a single file or document. So, with

this feature, users don't need to worry about tracking the latest version or who made any changes.

- Data Protection - By storing data on cloud storage services, data is well protected against all kinds of disasters, such as floods, earthquakes and human error.
- Disaster Recovery - Data stored in the cloud is not only protected from disasters by having the same copy at several locations, but can also favor disaster recovery in order to ensure business continuity.

The disadvantages or cons of storage as a service are given as follows

- Potential downtimes : Due to failure in cloud, vendors may go through periods of downtime where the service is not available, which may be a major issue for mission-critical data.
- Limited customization : As the cloud infrastructure is owned and managed by the service provider, it is less customizable.
- Vendor lock-in : Due to Potential for vendor lock-in, it may be difficult to migrate from one service provider to another.
- Unreliable : In some cases, there is still a possibility that the system could crash and leave consumers with no means of accessing their stored data. The small service provider becomes unreliable in that case. Therefore, when a cloud storage system is unreliable, it becomes a liability. No one wants to save data on an unstable platform or trust an organization that is unstable. Most cloud storage providers seek to resolve the issue of reliability through redundancy.

13. Difference between Amazon Glacier and Amazon S3.

Amazon Glacier	Amazon
It supports 40 TB archives	It supports 5 TB objects
It is recognised by archive IDs which are system	It can use "friendly" key names
It encrypts the archives	It is optional to encrypt the data
It is extremely low - cost storage	Its cost is much higher than Amazon

14. What are different risks in cloud storages ?

The following are the risks in cloud storage :

- **Dependency :** It is also known as "vendor-lock-in". The term alludes to the difficulties in moving from one cloud specialist organisation to other. This is because of the movement of information. Since administrations keep running over a remote virtual condition, the client is furnished with restricted access over the product and equipment, which gives rise to concerns about control.
- **Unintended Permanence :** There have been scenarios when cloud users complain that specific pictures have been erased in the current 'iCloud hack'. In this way, the specialist organisations are in full commitment that the client's information ought not be damaged or lost. Consequently, clients are urged to make full utilisation of cloud backup offices. Subsequently, the duplicates of documents might be recovered from the servers, regardless of the possibility that the client loses its records.
- **Insecure Interfaces and APIs :** To manage and interact with cloud services, various interfaces and APIs are used by customers. Two categories of web-based APIs are SOAP (based on web services) and REST (based on HTTP). These APIs are easy targets for man-in-the-middle or replay attacks. Therefore, secure authentication, encryption and access control must be used to provide protection against these malicious attacks.

- **Compliance Risks :** It is a risk for organisations that have earned certifications to either meet industry standards or to gain the competitive edge when migrating to clouds. This is a risk when cloud provider does not follow their own compliance requirements or when the cloud provider does not allow the audit by the cloud customer.

15. Enlist the different cloud storage providers.

The description about popular cloud storage providers are given as follows :

- Amazon S3 : Amazon S3 (Simple Storage Service) offers a simple cloud services interface that can be used to store and retrieve any amount of data from anywhere on the cloud at any time. It gives every developer access to the same highly scalable data storage infrastructure that Amazon uses to operate its own global website network. The goal of the service is to optimize the benefits of scale and to pass those benefits on to the developers.
- Google Bigtable Datastore : Google defines Bigtable as a fast and highly scalable datastore. The google cloud platform allows Bigtable to scale through thousands of

commodity servers that can store petabytes of data together. Bigtable has been designed with very high speed, versatility and extremely high scalability in mind.

- The size of the Bigtable database can be petabytes, spanning thousands of distributed servers. Bigtable is now open to developers as part of the Google App Engine, their cloud computing platform.
- Microsoft Live Mesh : Windows Live Mesh was a free-to-use Internet-based file synchronization application designed by Microsoft to enable files and directories between two or more computers to be synchronized on Windows or Mac OS platforms. It has support of mesh objects that consists of data feeds, which can be represented in Atom, RSS, JSON, or XML. It uses Live Framework APIs to share any data item between devices that recognize the data.
- Nirvanix : Nirvanix offers public, hybrid and private cloud storage services with usage-based pricing. It supports Cloud-based Network Attached Storage (CloudNAS) to store data in premises. Nirvanix CloudNAS is intended for businesses that manage archival, backup, or unstructured archives that need long-term, secure storage, or organizations that use automated processes to migrate files to mapped drives. The CloudNAS has built-in disaster data recovery and automatic data replication feature for up to three geographically distributed storage nodes.

16. What is Amazon S3 ?

Amazon S3 is a cloud-based storage system that allows storage of data objects in the range of 1 byte up to 5 GB in a flat namespace. The storage containers in S3 have predefined buckets, and buckets serve the function of a directory, though there is no object hierarchy to a bucket, and the user can save objects to it but not files. Amazon S3 offers a simple web services interface that can be used to store and retrieve any amount of data from anywhere, at any time on the web. It gives any developer access to the same scalable, secure, fast, low-cost data storage infrastructure that Amazon uses to operate its own global website network.

17. What is meant by Amazon Elastic Block Store (EBS) and SimpleDB?

- The Elastic Block Store (EBS) provides the volume block interface for saving and restoring the virtual images of EC2 instances.

- Traditional EC2 instances will be destroyed after use. The status of EC2 can now be saved in the EBS system after the machine is shut down. Users can use EBS to save persistent data and mount to the running instances of EC2.

18. What is Amazon SimpleDB Service?

- SimpleDB provides a simplified data model based on the relational database data model. Structured data from users must be organized into domains. Each domain can be considered a table. The items are the rows in the table.
- A cell in the table is recognized as the value for a specific attribute (column name) of the corresponding row. This is similar to a table in a relational database. However, it is possible to assign multiple values to a single cell in the table.

19. What is Microsoft Windows Azure(Azure)?

In 2008, Microsoft launched a Windows Azure platform to meet the challenges in cloud computing. This platform is built over Microsoft data centers.

Windows Azure offers a cloud platform built on Windows OS and based on Microsoft virtualization technology. Applications are installed on VMs deployed on the data-center servers. Azure manages all servers, storage, and network resources of the data center.

20. What is meant by SQL Azure?

SQL Azure This function makes it easier for users to visit and use the relational database associated with the SQL server in the cloud.

21. Define Dynamic CRM Service.

• Dynamic CRM service This provides software developers a business platform in managing CRM applications in financing, marketing, and sales and promotions.

22. Define Azure Table.

The Azure Table and Queue storage modes are aimed at much smaller data volumes. Queues provide reliable message delivery and are naturally used to support work spooling between web and worker roles. Queues consist of an unlimited number of messages which can be retrieved and pro- cessed at least once with an 8 KB limit on message size.

23. What is Eucalyptus?

Eucalyptus is a Linux-based open-source software architecture for cloud computing and also a storage platform that implements Infrastructure a Service (IaaS). It provides quick and efficient computing services. Eucalyptus was designed to provide services compatible with Amazon's EC2 cloud and Simple Storage Service(S3).

24. List out the important features of eucalyptus.

Important Features are:-

Images: A good example is the Eucalyptus Machine Image which is a module software bundled and uploaded to the Cloud.

Instances: When we run the picture and utilize it, it turns into an instance.

Networking: It can be further subdivided into three modes: Static mode(allocates IP address to instances), System mode (assigns a MAC address and imputes the instance's network interface to the physical network via NC), and Managed mode (achieves local network of instances).

Access Control: It is utilized to give limitations to clients.

Elastic Block Storage: It gives block-level storage volumes to connect to an instance.

Auto-scaling and Load Adjusting: It is utilized to make or obliterate cases or administrations dependent on necessities.

25. List out advantages of eucalyptus Cloud.

Advantages Of The Eucalyptus Cloud

- Eucalyptus can be utilized to benefit both the eucalyptus private cloud and the eucalyptus public cloud.
- Examples of Amazon or Eucalyptus machine pictures can be run on both clouds.
- Its API is completely similar to all the Amazon Web Services.
- Eucalyptus can be utilized with DevOps apparatuses like Chef and Puppet.
- Although it isn't as popular yet but has the potential to be an alternative to OpenStack and CloudStack.
- It is used to gather hybrid, public and private clouds.
- It allows users to deliver their own data centers into a private cloud and hence, extend the services to other organizations.

26. What is meant by Nimbus?

Nimbus:

Nimbus is a toolkit that, once included in the collection, provides infrastructure such as a cloud of service to its client through the WSRF-based web service APIs or Amazon EC2 WSDL. Nimbus is a free and open source software, subject to the requirements of the Apache License, version 2.

Nimbus supports both the Xen and KVM hypervisors as well as the portable device organizers Portable Batch System and Oracle Grid Engine. Allows the submission of customized visual clusters for content. It is adjustable in terms of planning, network rental, and accounting usage.

UNIT V

CLOUD SECURITY

Virtualization System-Specific Attacks: Guest hopping – VM migration attack – hyperjacking. Data Security and Storage; Identity and Access Management (IAM) - IAM Challenges - IAM Architecture and Practice.

GUEST HOPPING:

1.Explain in detail about Guest Hopping with an example.

Guest Hopping:

Guest hopping in the context of virtual machines and cloud computing typically refers to an attack scenario where an unauthorized user gains access to multiple virtual machines within a cloud environment. This type of attack can potentially compromise the security and integrity of the virtual machines and the data they contain.

Here are some considerations related to guest hopping attacks on virtual machines in a cloud computing environment:

Shared Infrastructure: Cloud computing often involves the sharing of physical resources among multiple virtual machines. If an attacker successfully compromises one virtual machine, they may attempt to leverage that access to gain unauthorized access to other virtual machines within the same infrastructure.

Hypervisor Vulnerabilities: The hypervisor is the software layer that manages and orchestrates virtual machines in a cloud environment. Exploiting vulnerabilities in the hypervisor could allow an attacker to break the isolation between virtual machines, enabling guest hopping attacks.

Misconfigurations: Misconfigurations in virtual machine settings or the cloud infrastructure can create security weaknesses that attackers can exploit to move laterally between virtual machines. This includes weak authentication mechanisms, insecure network configurations, or inadequate access controls.

Privilege Escalation: Once inside a virtual machine, an attacker may attempt to escalate their privileges to gain administrative or root access. This can be achieved by exploiting vulnerabilities in the guest operating system or misconfigurations within the virtual machine.

Data Exfiltration and Malware Propagation: Once an attacker gains access to multiple virtual machines, they may exfiltrate sensitive data from those machines or propagate malware to further compromise the cloud infrastructure or launch attacks on other targets.

To mitigate guest hopping attacks in a cloud computing environment, it is crucial to follow security best practices:

Regularly patch and update the hypervisor, virtual machine software, and guest operating systems to address known vulnerabilities.

Implement strong access controls, including robust authentication mechanisms, to prevent unauthorized access to virtual machines.Employ network segmentation and isolation techniques to limit lateral movement between virtual machines.

Use intrusion detection and prevention systems to monitor and detect suspicious activities within the cloud environment.

Regularly audit and review system logs for any signs of unauthorized access or anomalous behavior. Educate users and administrators about secure configuration practices and the importance of adhering to security guidelines.

By implementing these measures, organizations can reduce the risk of guest hopping attacks and enhance the security of their cloud computing environments.

Application-level security issues (or cloud service provider CSP level attacks) refer to intrusion from the malicious attackers due to vulnerabilities of the shared nature of the cloud. Some companies host their applications in shared environments used by multiple users, without considering the possibilities of exposure to security breaches, such as:

1. SQL Injection

An unauthorized user gains access to the entire database of an application by inserting malicious code into a standard SQL code. Often used to attack websites, SQL injection can be avoided by the usage of dynamically generated SQL in the code. It is also necessary to remove all stored procedures that are rarely used and assign the least possible privileges to users who have permission to access the database.

2. Guest-Hopping Attack

In guest-hopping attacks, due to the separation failure between shared infrastructures, an attacker gets access to a virtual machine by penetrating another virtual machine hosted in the same hardware. One possible mitigation of guest-hopping attack is the Forensics and VM debugging tools to observe any attempt to compromise the virtual machine. Another solution is to use the High Assurance Platform (HAP), which provides a high degree of isolation between virtual machines.

3. Side-Channel Attack

An attacker opens a side-channel attack by placing a malicious virtual machine on the same physical machine as the victim machine. Through this, the attacker gains access to all confidential information on the victim machine. The countermeasure to eliminate the risk of side-channel attacks in a virtualized cloud environment is to ensure that no legitimate user VMs reside on the same hardware of other users.

4. Malicious Insider

A malicious insider can be a current or former employee or business associate who maliciously and intentionally abuses system privileges and credentials to access and steal sensitive customer information within the network of an organization. Strict privilege planning and security auditing can minimize this security risk that originates from within an organization.

5. Cookie Poisoning

Cookie poisoning means to gain unauthorized access into an application or a webpage by modifying the contents of the cookie. In a SaaS model, cookies contain user identity credential information that allows the applications to authenticate the user identity. Cookies are forged to impersonate an authorized user. A solution is to clean up the cookie and encrypt the cookie data.

6. Backdoor And Debug Option

The backdoor is a hidden entrance to an application, which was created intentionally or unintentionally by developers while coding. Debug option is also a similar entry point, often used by developers to facilitate troubleshooting in applications. But the problem is that the hackers can use these hidden doors to bypass security policies and enter the website and access the sensitive information. To prevent this kind of attack, developers should disable the debugging option.

7. Cloud Browser Security

A web browser is a universal client application that uses Transport Layer Security (TLS) protocol to facilitate privacy and data security for Internet communications. TLS encrypts the connection between web applications and servers, such as web browsers loading a website. Web browsers only use TLS encryption and TLS signature, which are not secure enough to defend malicious attacks. One of the solutions is to use TLS and at the same time XML based cryptography in the browser core.

8. Cloud Malware Injection Attack

A malicious virtual machine or service implementation module such as SaaS or IaaS is injected into the cloud system, making it believe the new instance is valid. If succeeded, the user requests are redirected automatically to the new instance where the malicious code is executed. The mitigation is to perform an integrity check of the service instance before using it for incoming requests in the cloud system.

9. ARP Poisoning

Address Resolution Protocol (ARP) poisoning is when an attacker exploits some ARP protocol weakness to map a network IP address to one malicious MAC and then update the ARP cache with this malicious MAC address. It is better to use static ARP entries to minimize this attack. This tactic can work for small networks such as personal clouds, but it is easier to use other strategies such as port security features on large-scale clouds to lock a single port (or network device) to a particular IP address.

VM MIGRATION ATTACK

2. Explain in detail about VM migration attack with an example.

VM migration is the process of moving a virtual machine from one physical host to another within a cloud infrastructure. It allows for workload balancing, resource optimization, and maintenance activities in a dynamic cloud environment. However, if the migration process is compromised, it can lead to security risks and potential unauthorized access to VMs and their data.

Here are the steps an attacker might take in a VM migration attack:

1. Interception of Migration Traffic: Attackers may attempt to intercept the migration traffic between the source and destination hosts. By positioning themselves as a "man-in-the-middle," they can eavesdrop on sensitive information, such as the VM's contents, network communications, and credentials exchanged during the migration process.

2. Unauthorized VM Migration: An attacker might try to initiate unauthorized VM migrations within the cloud environment. This can involve moving VMs to their control or to compromised hosts under their

influence. Once the VM is under their control, the attacker can gain access to sensitive data, manipulate the VM's behavior, or disrupt the cloud infrastructure.

3. Exploiting Migration Channel Vulnerabilities: The migration process relies on communication channels and protocols between the source and destination hosts. Attackers may exploit vulnerabilities or weaknesses in these channels to gain unauthorized access, inject malicious code into the VM, or manipulate the migration process to their advantage.

4. Resource Exhaustion: Attackers can target the resources involved in the migration process, such as network bandwidth or storage, to cause resource exhaustion. By overwhelming these resources, the attacker can disrupt VM migrations, cause denial-of-service conditions, or impact the availability and performance of other cloud services.

5. VM Rollback Attacks: During VM migration, checkpoints or snapshots are often created to ensure data consistency. Attackers might attempt to tamper with these snapshots or manipulate the rollback process. By doing so, they can compromise the integrity of the VM or introduce unauthorized changes to the VM's state or data.

To mitigate VM migration attacks in a cloud computing environment, the following security measures are recommended:

1. Secure Communication Channels: Encrypting the migration traffic using secure protocols, such as SSL/TLS, ensures the confidentiality and integrity of data during transit. This protects against interception, eavesdropping, and tampering.

2. Authentication and Authorization: Implement strong authentication mechanisms and access controls to ensure that only authorized users can initiate or participate in VM migration processes. This prevents unauthorized individuals from manipulating or intercepting the migration.

3. Intrusion Detection and Prevention: Employ intrusion detection and prevention systems (IDPS) to monitor migration traffic and detect any anomalous or malicious activities. IDPS can identify suspicious patterns, detect unauthorized access attempts, and raise alerts for further investigation.

4. Network Segmentation: Segmenting the network infrastructure separates migration traffic from other data traffic. This reduces the attack surface and limits the impact of potential compromises. Additionally, network segmentation helps in applying more granular security controls and monitoring specifically for migration-related activities.

5. Resource Monitoring and Protection: Implement mechanisms to monitor resource utilization during VM migrations. This helps in detecting and mitigating resource exhaustion attacks, ensuring that sufficient resources are available to complete migrations successfully.

6. Regular Updates and Patching: Keep the migration infrastructure, including the hypervisor and migration software, up to date with the latest security patches. Regular updates address known vulnerabilities and minimize the risk of exploitation.

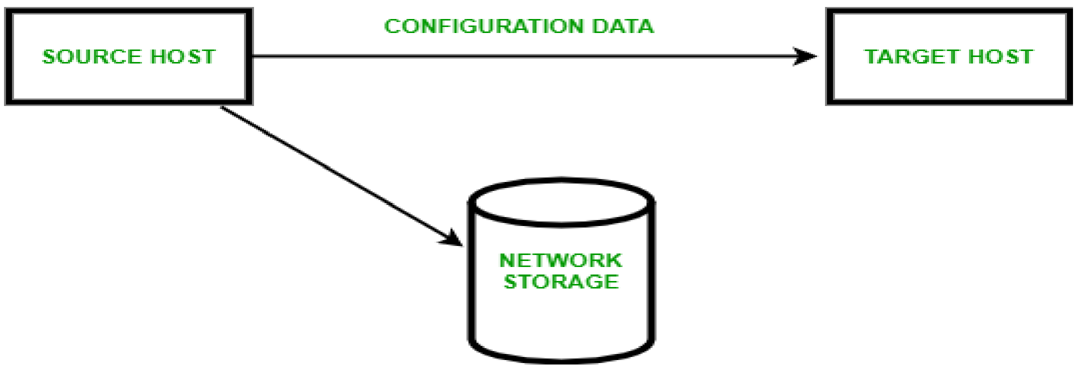
7. Auditing and Logging: Enable comprehensive logging of migration activities, including the source and destination hosts, migration timestamps, and user information. Regularly review the logs to identify any suspicious or unauthorized activities. Logging is crucial for forensic analysis and investigations in case of a security incident.

By implementing these security measures, organizations can strengthen the security and integrity of VM migration processes within their cloud computing

1. Cold Migration :

A powered down Virtual Machine is carried to separate host or data store. Virtual Machine's power state is OFF and there is no need of common shared storage. There is a lack of CPU check and there is long shortage time. Log files and configuration files are migrated from the source host to the destination host.

2. The first host's Virtual Machine is shut down and again started on next host. Applications and OS are terminated on Virtual Machines before moving them to physical devices. User is given choice of movement of disks associated from one data store to another one.



Host's Virtual Machine

3. Hot Migrations :

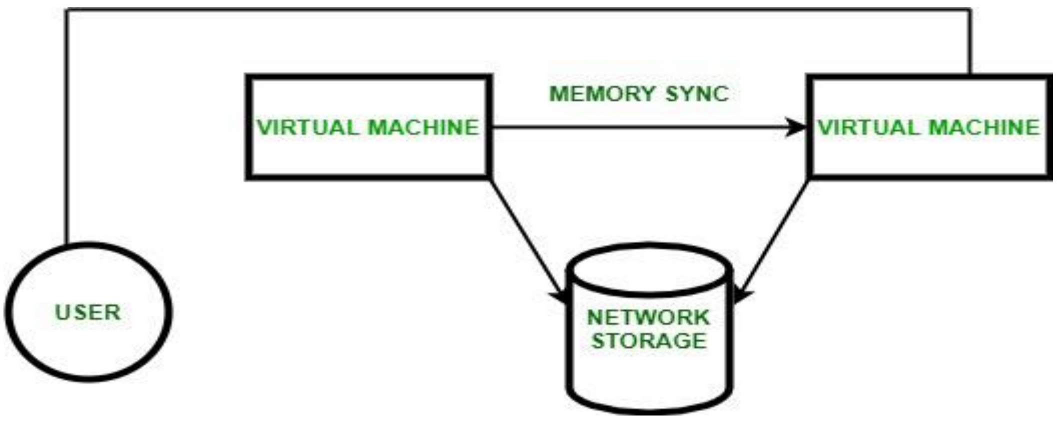
A powered on Virtual Machine is moved from one physical host to another physical host. A source host state is cloned to destination host and then that source host state is discarded. Complete state is shifted to the destination host. Network is moved to destination Virtual Machine.

A common shared storage is needed and CPU checks are put into use. Shortage time is very little. Without stoppage of OS or applications, they are shifted from Virtual Machines to physical machines. The physical server is freed for maintenance purposes and workloads (which are among physical servers) are dynamically balanced so as to run at optimized levels. Downtime of clients is easily avoidable.

Suspend first host's Virtual Machine and then clone it across registers of CPU and RAM and again resume some time later on second host. This migration runs when source system is operative.

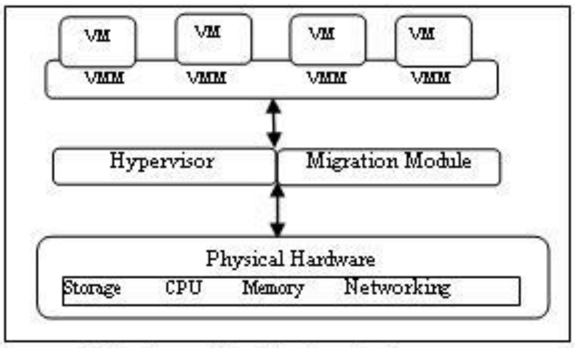
Stage-0:
Is Pre-Migration stage having functional Virtual Machine on primary host.
Stage-1:
Is Reservation stage initializing container on destination host.
Stage-2:
Is Iterative pre-copy stage where shadow paging is enabled and all dirty pages are cloned in succession rounds.
Stage-3:

Is Stop and copy where first host's Virtual Machine is suspended and all remaining Virtual Machine state are synchronized on second host.
Stage-4:
Is Commitment where there is minimization of Virtual Machine state on first host.
Stage-5:
Is Activation stage where second host's Virtual Machine start and establishes connection to all local computers resuming all normal activities.



Host's Virtual Machine start and establishes connection

provides the technologies of virtual machine migration with its execution procedures. Basically it is the process of migrate a virtual machine from one host to another. It also has the capability to move workload of multiple running virtual machines on a single physical machine. The main difference between virtualization and virtual machine migration is that only migration module is inculcate with hypervisor. The architecture of virtual machine migration virtualized platform is shown in figure:



Migration – Virtualization Architecture

Although, the process of migration has been initiated in 1980, but it was used often-ally, due to its main limitation i.e. how to handle interaction between various modules of operating system. But it overcomes in virtual machine migration because it moves the whole operating system along with running processes. VM migration becomes this process simplified and efficient. It also takes care of load balancing, energy consumption, workload consolidation etc. Henceforth, it becomes more popular and wide adoption in industry. Below table describes the types of VM migrations.

VM Migration Type	Description
Cold Migration	Before migration, the virtual machine must be powered off, after doing this task. The old one should be deleted from source host. Moreover, the virtual machine need not to be on shared storage.
Warm Migration	Whenever transfer OS and any application, there is no need to suspend the source host. Basically it has high demand in public cloud.
Live Migration	It is the process of moving a running virtual machine without stopping the OS and other applications from source host to destination host.

Techniques of VM Migration

This subsection describes the types of virtual machine migration techniques. It is basically of two types:- copy Migration

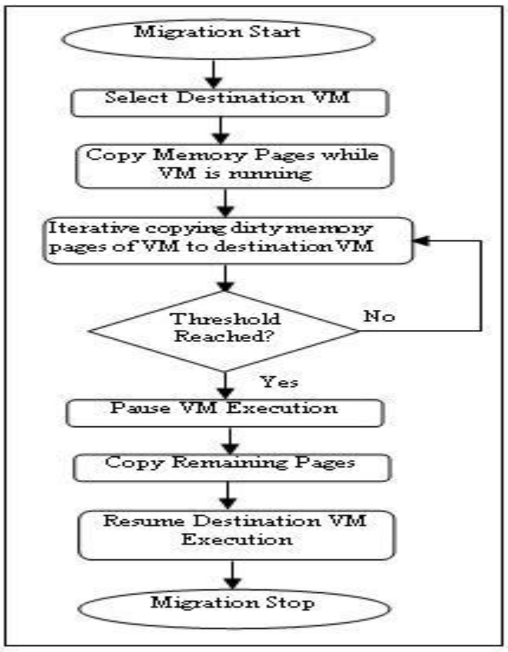
Po-Copy Migration

Pre- Copy Migration: In this migration, the hypervisor copies all memory page from source machine to destination machine while the virtual machine is running. It has two phases: Warm- up Phase and stop and copy phase.

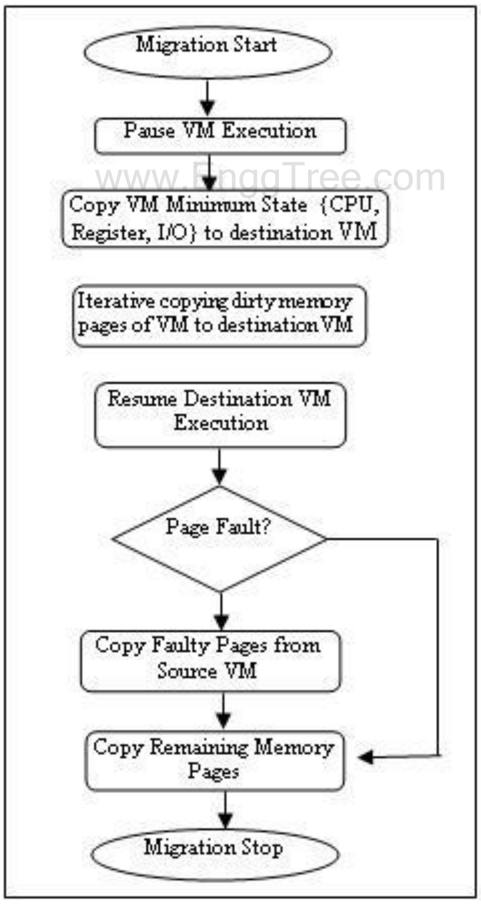
Warm Up Phase: During copying all memory pages from source to destination, some memory pages changed because of source machine CPU is active. All the changed memory paged known as dirty pages. All these dirty pages are required to recopy on destination machine; this phase is called as warm up phase.

Stop & Copy Phase: Warm up phase is repeated until all the dirty pages recopied on destination machine. This time CPU of source machine is deactivated till all memory pages will transfer another machine. Ultimately at this time CPU of both source and destination is suspended, this is known as down time phase. This is the main thing that has to explore in migration for its optimization.

Post- Copy Migration: In this technique, VM at the source is suspended to start post copy VM migration. When VM is suspended, execution state of the VM (i.e. CPU state, registers, non-pageable memory) is transferred to the target. In parallel the sources actively send the remaining memory pages of the VM to the target. This process is known as pre-paging. At the target, if the VM tries to access a page that has not been transferred yet, it generates a page fault, also known as network faults. These faults are redirect to the source, which responds with the faulted pages. Due to this, the performance of applications is degrading with number of network faults. To overcome this, pre-paging scheme is used to push pages after the last fault by dynamically using page transmission order. Figure 3 & 4 shows the pre copy and post copy migration technique respectively.


Migration Start
Select Destination VM
Copy Memory Pages while VM is running
Iterative copying dirty memory pages of VM to destination VM
Threshold Reached?
No
Yes
Pause VM Execution
Copy Remaining Pages
Resume Destination VM Execution
Migration Stop

Pre- Copy Migration Technique


Migration Start
Pause VM Execution
Copy VM Minimum State {CPU, Register, I/O} to destination VM
Iterative copying dirty memory pages of VM to destination VM
Resume Destination VM Execution
Page Fault?
Copy Faulty Pages from Source VM
Copy Remaining Memory Pages
Migration Stop

Post- Copy Migration Technique

Hyperjacking:

3**. Explain in detail about Hyperjacking with an example.**

Hyperjacking refers to a type of cyber attack where an attacker gains unauthorized access and control over a hypervisor in a virtualized environment. The hypervisor is a critical component that manages and orchestrates multiple virtual machines (VMs) on a physical host. By compromising the hypervisor, the attacker can potentially control and manipulate the entire virtualized infrastructure, including the VMs running on it.

Here are some key points to understand about hyperjacking:

1. Hypervisor Exploitation: Hyperjacking involves exploiting vulnerabilities in the hypervisor software to gain unauthorized access. This can be achieved through various attack vectors, such as exploiting software vulnerabilities, misconfigurations, or weak authentication mechanisms associated with the hypervisor.

2. Privilege Escalation: Once an attacker gains access to the hypervisor, they can attempt to escalate their privileges to gain administrative control. This allows them to control the VMs, modify their configurations, access sensitive data, or even launch further attacks within the virtualized environment.

3. Lateral Movement: Hyperjacking attacks may involve lateral movement within the virtualized environment. Once the hypervisor is compromised, the attacker can attempt to move laterally across different VMs, potentially compromising their security and integrity.

4. Data Exfiltration and Manipulation: With control over the hypervisor and potentially the VMs, an attacker can exfiltrate sensitive data from the VMs or manipulate their contents. This can lead to unauthorized access to critical information or tampering with the operations of virtualized systems.

To prevent hyperjacking attacks and enhance the security of virtualized environments, consider the following measures:

1. Patching and Updates: Keep the hypervisor software and virtualization platform up to date with the latest security patches and updates. Regularly applying patches helps mitigate known vulnerabilities that could be exploited by attackers.

2. Secure Configuration: Follow best practices for securing the hypervisor and virtualized environment, such as disabling unnecessary services, implementing strong authentication mechanisms, and applying appropriate access controls.

3. Hypervisor Hardening: Employ hypervisor hardening techniques to reduce the attack surface and strengthen the security of the hypervisor. This can include measures like disabling unused features, configuring secure network settings, and employing intrusion detection systems for monitoring.

4. Network Segmentation: Implement network segmentation within the virtualized environment to isolate different VMs and limit the lateral movement of an attacker who gains access to the hypervisor. This helps contain the impact of a potential hyperjacking attack.

5. Hypervisor Security Monitoring: Implement robust monitoring and logging mechanisms to detect suspicious activities and potential signs of hyperjacking. Monitor hypervisor logs, network traffic, and other relevant indicators to identify any unauthorized access or abnormal behavior.

6. Access Control and Authentication: Implement strong access control measures for the hypervisor, including multi-factor authentication, role-based access control, and regular review of access privileges. This helps minimize the risk of unauthorized access to the hypervisor.

By implementing these security measures, organizations can reduce the risk of hyperjacking attacks and strengthen the overall security posture of their virtualized environments. Regular security assessments and audits can also help identify and address potential vulnerabilities before they are exploited by attackers.

DATA AND ITS SECURITY

4. **Discuss in detail about provider data and its security.(May-2023)**

In addition to the security of your own customer data, customers should also be concerned about what data the provider collects and how the CSP protects that data. Specifically with regard to your customer data, what metadata does the provider have about your data

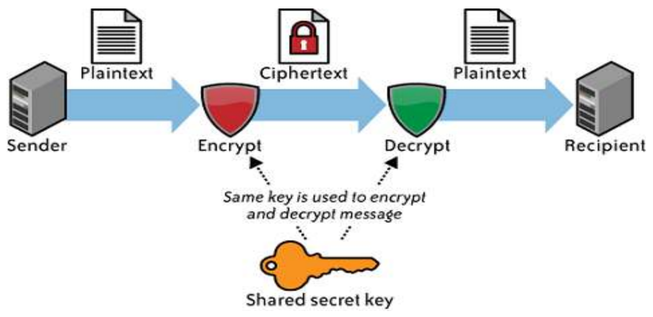
Storage

For data stored in the cloud (i.e., storage-as-a-service), we are referring to IaaS and not data associated with an application running in the cloud on PaaS or SaaS. The same three information security concerns are associated with this data stored in the cloud (e.g., Amazon's S3) as with data stored elsewhere: confidentiality, integrity, and availability.

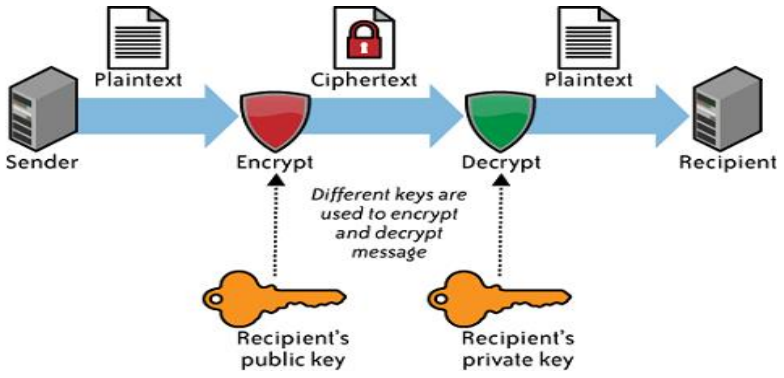
Confidentiality

For large organizations, this coarse authorization presents significant security concerns unto itself. Often, the only authorization levels cloud vendors provide are administrator authorization and user authorization with no levels in between. Again, these access control issues are not unique to CSPs

Symmetric Encryption Diagram



Asymmetric Encryption Diagram



<u>IAM</u>

5. Draw the architecture of IAM and explain in detail.

IAM- Identity Access Management

The protection of enterprise information assets is critical to improve and sustain the business, which is one of the core security aspects of architecture.

- **Identity Governance:** The ability in making sure the right people are granted the right access rights, making sure the wrong ones are not and managing the lifecycle through organization structure, processes and enabling technology.
- **Directory Services:** The ability in enforcing access rights, within specified policy, when users attempt to access a desired application, system or platform.
- **Access Management:** The ability to provide ways to control storage of identity information about users and access rights.

Why IAM?

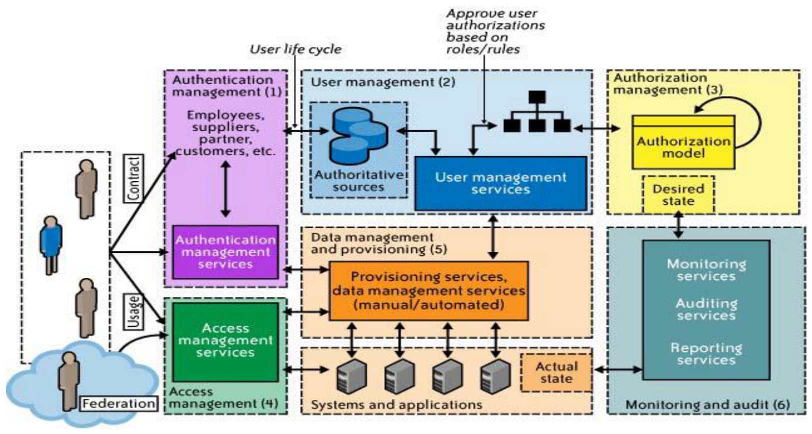
- *Improve operational efficiency*
- Properly architected IAM technology and processes can improve efficiency by automating user on-boarding and other repetitive tasks
- *Regulatory compliance management*

– disgruntled employees deleting sensitive files

– HIPAA, SOX

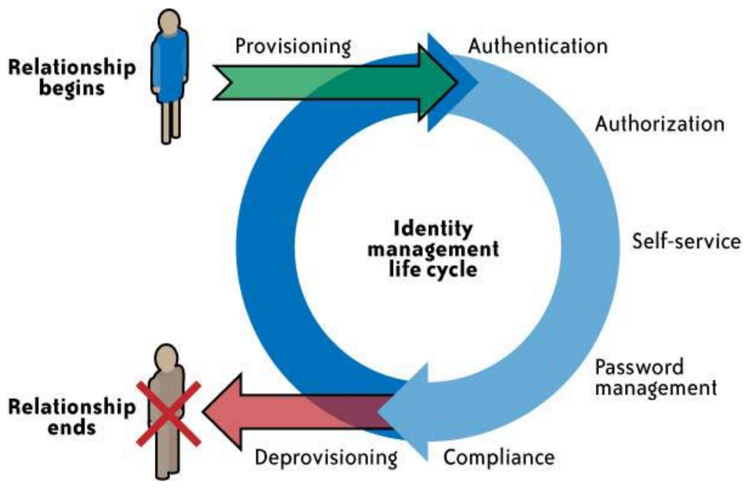
IAM Architecture and Practice

- User management
- Authentication management
- Authorization management
- Access management
- Data management and provisioning
- Monitoring and auditing

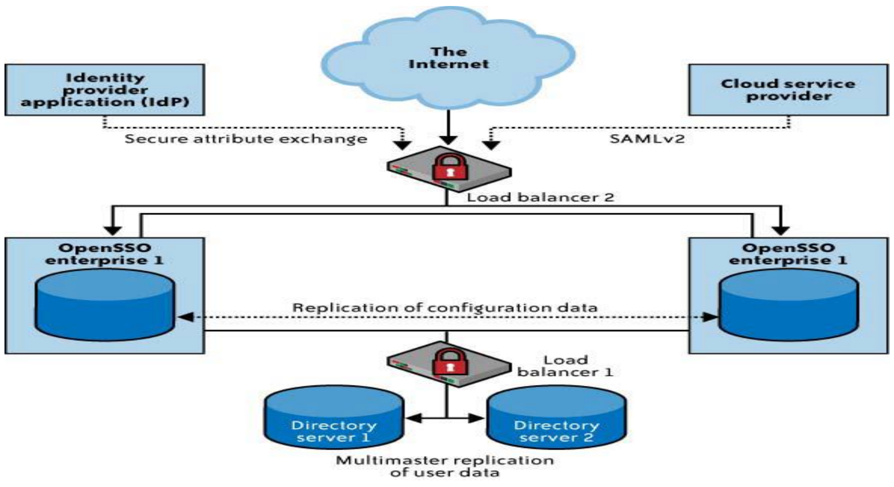


Approve user authorizations based on roles/rules
User life cycle
Authentication management (1)
Employees, suppliers, partner, customers, etc.
Authentication management services
User management (2)
Authoritative sources
User management services
Authorization management (3)
Authorization model
Desired state
Contract
Usage
Federation
Access management services
Access management (4)
Data management and provisioning (5)
Provisioning services, data management services (manual/automated)
Actual state
Systems and applications
Monitoring services
Auditing services
Reporting services
Monitoring and audit (6)

Enterprise IAM functional architecture



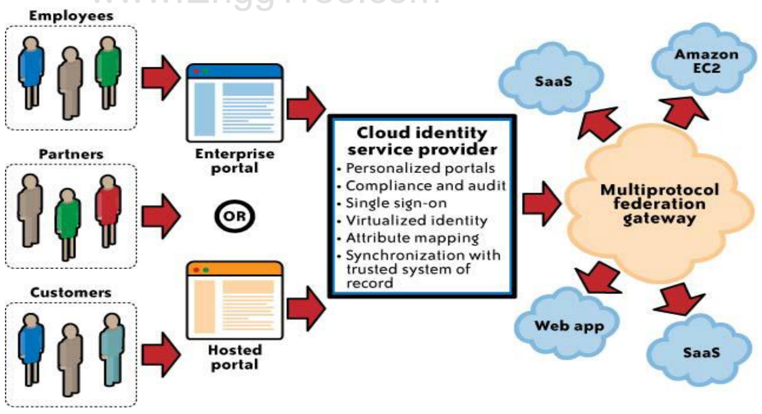
Identity life cycle



Identity provider deployment architecture

User management functions

- Cloud identity administration
- Federation or SSO
- Authorization management
- Compliance management



IDaaS- IDentity management as a Service

SaaS

- Two major challenges for identity management

– Is the organization ready to provision and manage the user life cycle by extending its established IAM practice to the SaaS service?

– Are the SaaS provider capabilities sufficient to automate user provisioning and life cycle management without implementing a custom solution for the SaaS service?

Customer responsibilities

- User provisioning
- Profile management
- SaaS IAM capability evaluation
- Investigation support
- Compliance management

PaaS

- Customer Responsibility

– PaaS platform service levels

– Third-party web services provider service levels

– Network connectivity parameters for the network (Internet)

- PaaS Health Monitoring

Health Monitoring Services

- Service health dashboard published by the CSP
- CCID (this database is generally community-supported, and may not reflect all CSPs and all incidents that have occurred)
- CSP customer mailing list that notifies customers of occurring and recently occurred outages
- RSS feed for RSS readers with availability and outage information

IaaS availability in cloud

- Availability of a CSP network, host, storage, and support application infrastructure
- Availability of your virtual servers and the attached storage for compute services.
- Availability of virtual storage that your users and virtual server depend on for storage service.
- Availability of your network connectivity to the Internet or virtual network connectivity to IaaS services.
- Availability of network services, including a DNS, routing services, and authentication services required to connect to the IaaS service.

IaaS Health Monitoring

- Service health dashboard published by the CSP.
- CCID (this database is generally community-supported, and may not reflect all CSPs and all incidents that have occurred).
- CSP customer mailing list that notifies customers of occurring and recently occurred outages.

• Internal or third-party-based service monitoring tools (e.g., Nagios) that periodically check the health of your IaaS virtual server.

Key privacy issues in the cloud

• Typical issues with regard to the dependence on the Cloud Computing provider are

– Cloud Computing provider were to go bankrupt and stopped providing services, the customer could experience problems in accessing data and therefore potentially in business continuity

– Some widely used Cloud Computing services (e.g. GoogleDocs) do not include any contract between the customer and Cloud Computing provider.

The IAM architecture is made up of several processes and activities (see Fig. 4.9.2). The processes supported by IAM are given as follows.

a) **User management -** It provides processes for managing the identity of different entities.

b) **Authentication management -** It provides activities for management of the process for determining that an entity is who or what it claims to be.

c) **Access management -** It provides policies for access control in response to request for resource by entity.

d) **Data management -** It provides activities for propagation of data for authorization to resources using automated processes.

e) **Authorization management -** It provides activities for determining the rights associated with entities and decide what resources an entity is permitted to access in accordance with the organization's policies.

f) **Monitoring and auditing -** Based on the defined policies, it provides monitoring, auditing, and reporting compliance by users regarding access to resources.

The activities supported by IAM are given as follows.

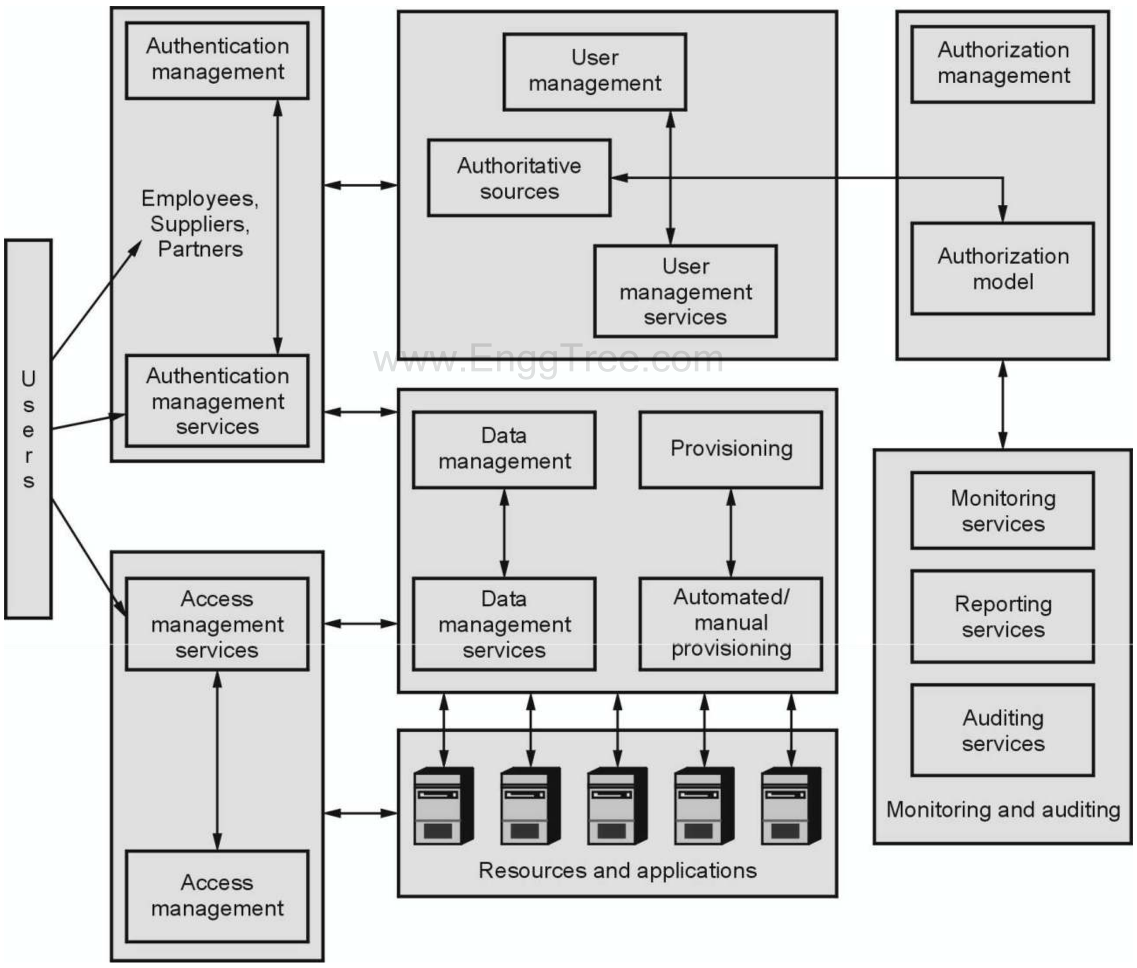
a) **Provisioning -** The provisioning has essential processes that provide users with necessary access to data and resources. It supports management of all user account operations like add, modify, suspend, and delete users with password management. By provisioning the users are given access to data, systems, applications, and databases based on a unique user identity. The deprovisioning does the reverse of provisioning which deactivate of delete the users identity with privileges.

b) **Credential and attribute management -** The Credential and attribute management prevents identity impersonation and inappropriate account use. It deals with management of credentials and user attributes such as create, issue, manage and revoke users to minimize the business risk associated with it. The individuals' credentials are verified during the authentication process. The Credential and attribute management processes include provisioning of static or dynamic attributes that comply with a password standard, encryption management of credentials and handling access policies for user attributes.

c) **Compliance management -** The Compliance management is the process used for monitoring the access rights and privileges and tracked to ensure the security of an enterprise's resources. It also helpful to auditors to verify the compliance to various access control policies, and standards. It includes practices like access monitoring, periodic auditing, and reporting.

Identity federation management - Identity federation management is the process of managing the trust relationships beyond the network boundaries where organizations come together to exchange the information about their users and entities.

e) Entitlement management - In IAM, entitlements are nothing but authorization policies. The Entitlement management provides processes for provisioning and deprovisioning of privileges needed for the users to access the resources
including systems, applications, and databases.



IAM

Security Standards

Security standards are needed to define the processes, measures and practices required to implement the

security program in a web or network environment. These standards also apply to cloud-related IT exercises and include specific actions to ensure that a secure environment is provided for cloud services along with privacy for confidential information. Security standards are based on a set of key principles designed to protect a trusted environment of this kind. The following sections explain the different security
standards used in protecting cloud environment.

Security Assertion Markup Language (SAML)

Security Assertion Markup Language (SAML) is a security standard developed by OASIS Security Services Technical Committee that enables Single Sign-On technology (SSO) by offering a way of authenticating a user once and then communicating authentication to multiple applications. It is an open standard for exchanging authentication and authorization data between parties, in particular, between an identity provider and a service provider.

It enables Identity Providers (IdPs) to pass permissions and authorization credentials to Service Providers (SP). A range of existing standards, including SOAP, HTTP, and XML, are incorporated into SAML. An SAML transactions use Extensible Markup Language (XML) for standardized communications between the identity provider and service providers. SAML is the link between the authentication of a user's identity and the authorization to use a service. The majority of SAML transactions are in a standardized XML form. The XML schema is mainly used to specify SAML assertions and protocols. For authentication and message integrity, both SAML 1.1 and SAML 2.0 use digital signatures based on the XML Signature Standard. XML encryption is supported in SAML 2.0 but not by SAML 1.0 as it doesn't support encryption capabilities. SAML defines assertions, protocol, bindings and profiles based on XML.

Open Authentication (OAuth)

OAuth is a standard protocol which allows secure API authorization for various types of web applications in a simple, standard method. OAuth is an open standard for delegating access and uses it as a way of allowing internet users to access their data on websites and applications without passwords. It is a protocol that enables secure authorization from web, mobile or desktop applications in a simple and standard way. It is a publication and interaction method with protected information. It allows developers access to their data while securing credentials of their accounts. OAuth enables users to access their information which is shared by service providers and consumers without sharing all their identities. This mechanism is used by companies such as Amazon, Google, Facebook, Microsoft and Twitter to permit the users to share information about their accounts with third party applications or websites. It specifies a process for resource owners to authorize third-party access to their server resources without

sharing their credentials. Over secure Hypertext Transfer Protocol (HTTPs), OAuth essentially allows access tokens to be issued to third-party clients by an authorization server, with the approval of the resource owner. The third party then uses the access token to access the
protected resources hosted by the resource server.

Secure Sockets Layer and Transport Layer Security

Secure Sockets Layer (SSL) and Transport Layer Security (TLS) are cryptographically secure protocols to provide security and data integrity for TCP/IP based communications. The network connections segments in the transport layer are encrypted by the TLS and SSL. In web browsers, e-mail, instant messaging and voice over IP, many implementations of these protocols are widely used. TLS is the latest updated IETF standard protocol for RFC 5246. The TLS protocol allows client/server applications to communicate across a network in a way that avoids eavesdropping, exploitation, tampering and message forgery. TLS uses cryptography to ensure endpoint authentication and data confidentiality. A more secure bilateral connection mode is also supported by TLS ensuring that both ends of the connection communicate with the individual they believe is connected. This is called mutual authentication. The TLS client side must also keep a certificate for mutual authentication. Three basic phases involve TLS are Algorithm support for pair negotiation involves cipher suites that are negotiated between the client and the server to determine the ciphers being used; Authentication and key exchange involves decisions on authentication algorithms and key exchange to be used. Here key exchange and authentication algorithms are public key algorithms; and Message authentication using Symmetric cipher encryption determines the message authentication codes. The Cryptographic hash functions are used for message authentication codes. Once these decisions are made, the transfer of data can be commenced.

PART-A

1. **How to perform virtual machine security?**

Firewalls, intrusion detection and prevention, integrity monitoring, and log inspection can all be deployed as software on virtual machines to increase protection and maintain compliance integrity of servers and applications as virtual resources move from on-premises to public cloud environments.

Integrity monitoring and log inspection software must be applied at the virtual machine level.

2. Define IAM.

Identity and access management is a critical function for every organization, and a fundamental expectation of SaaS customers is that the principle of least privilege is granted to their data.

3. Why cloud requires security standards?

Security standards define the processes, procedures, and practices necessary for implementing a security program.

These standards also apply to cloud related IT activities and include specific steps that should be taken to ensure a secure environment is maintained that provides privacy and security of confidential information in a cloud environment.

4. What is SAML?

Security Assertion Markup Language (SAML) is an XML-based standard for communicating authentication, authorization, and attribute information among online partners.

It allows businesses to securely send assertions between partner organizations regarding the identity and entitlements of a principal.

5. List the types of statements are provided by SAML.

- Authentication statements
- Attribute statements
- Authorization decision statements

6. Describe about SAML protocol.

- A SAML protocol describes how certain SAML elements (including assertions) are packaged within SAML request and response elements SAML protocol is a simple request–response protocol.
- The most important type of SAML protocol request is a query.

7. List the types of SAML queries.

- Authentication query
- Attribute query
- Authorization decision query.

8. What is OAuth?

- OAuth (Open authentication) is an open protocol, initiated by Blaine Cook and Chris Messina, to allow secure API authorization in a simple, standardized method forvarious types of web applications.
- OAuth is a method for publishing and interacting with protected data.
- OAuth allows users to grant access to their information, which is shared by the
- service provider and consumers without sharing all of their identity.

9. What is the purpose of OpenID?

- OpenID is an open, decentralized standard for user authentication and access control that allows users to log onto many services using the same digital identity.
- It is a single-sign-on (SSO) method of access control.
- An OpenID is in the form of a unique URL and is authenticated by the entity hosting the OpenID URL.

10. Why cloud environment need SSL/TLS?

- SSL/TLS Transport Layer Security (TLS) and its predecessor, Secure Sockets Layer (SSL), are cryptographically secure protocols designed to provide security and data integrity for communications over TCP/IP.
- TLS and SSL encrypt the segments of network connections at the transport layer.

11. What is mutual authentication?

- TLS also supports a more secure bilateral connection mode whereby both ends of the connection can be assured that they are communicating with whom they believe they are connected. This is known as mutual authentication.
- Mutual authentication requires the TLS client side to also maintain a certificate