Title: Cloud Computing

Authors Name: Er. Karan Kumar

CLOUD COMPUTING

Course Outcome (CO)

At the end of course, the student will be able to understand

CO 1 Articulate the main concepts, key technologies, strengths and limitations of cloud computing.

CO 2 Learn the key and enabling technologies that help in the development of cloud.

CO 3 Develop the ability to understand and use the architceture of compute and storage cloud, service and delivery models.

CO 4 Explain the core issues of cloud computing such as resource managcment and security

CO 5 To appreciate the emergence of cloud as the next generation computing paradigm.

Syllabus

UNIT-I: INTRODUCTION

Introduction to Cloud Computing – Definition of Cloud – Evolution of Cloud Computing – Underlying Principles of Parallel and Distributed Computing – Cloud Characteristics – Elasticity in Cloud – On-demand Provisioning.

UNIT-II: CLOUD ENABLING TECHNOLOGIES

Service Oriented Architecture – REST and Systems of Systems – Web Services – Publish-Subscribe Model – Basics of Virtualization – Types of Virtualization – Implementation Levels of Virtualization – Virtualization Structures – Tools and Mechanisms – Virtualization of CPU – Memory – I/O Devices –Virtualization Support and Disaster Recovery.

UNIT-III: CLOUD ARCHITECTURE, SERVICES AND STORAGE

Layered Cloud Architecture Design – NIST Cloud Computing Reference Architecture – Public, Private and Hybrid Clouds – laaS – PaaS – SaaS – Architectural Design Challenges – Cloud Storage – Storage-as-a Service – Advantages of Cloud Storage – Cloud Storage Providers – S3.

UNIT-IV: RESOURCE MANAGEMENT AND SECURITY IN CLOUD

Inter Cloud Resource Management – Resource Provisioning and Resource Provisioning Methods – Global Exchange of Cloud Resources – Security Overview – Cloud Security Challenges – Software-as-a-Service Security – Security Governance – Virtual Machine Security – IAM – Security Standards.

UNIT-V: CLOUD TECHNOLOGIES AND ADVANCEMENTS

Hadoop – MapReduce – Virtual Box — Google App Engine – Programming Environment for Google App Engine –– Open Stack – Federation in the Cloud – Four Levels of Federation – Federated Services and Applications – Future of Federation.

Index

UNIT 1

INTRODUCTION

1.1 Introduction to Cloud-Computing

Cloud-Computing is the delivery of computing services- including server, storage, databases, networking, software, analytics and intelligence-over the internet ("the cloud") to offer faster innovation, flexible resources, and economies of scale. User typically pay only for cloud services that they use. It helps in lowering operating cost, run the infrastructure more efficiently and scale business needs change.

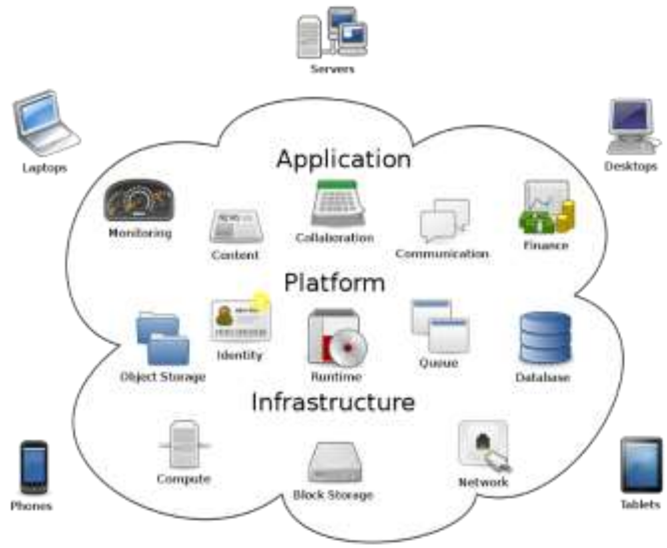


Fig 1.1 Cloud-Computing Environment

1.1.1 The NIST Definition of Cloud Computing:

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models (SaaS, PaaS, IaaS), and four deployment models (Public, Private, Hybrid and Community Cloud). The various essential characteristics have been given below.

1.1.2 Essential Characteristics:

- **On-demand self-service:** A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service provider.
- **Broad network access:** Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g., mobile phones, tablets, laptops, and workstations).
- **Resource pooling:** The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. There is a sense of location independence in that the customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction (e.g., country, state, or datacenter). Examples of resources include storage, processing, memory, and network bandwidth.
- **Rapid elasticity:** Capabilities can be elastically provisioned and released, in some cases automatically, to scale rapidly outward and inward commensurate with demand. To the consumer, the capabilities available for provisioning often appear to be unlimited and can be appropriated in any quantity at any time.
- **Measured service:** Cloud systems automatically control and optimize resource use by leveraging a metering capability1 at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts). Resource usage can be monitored, controlled, and reported, providing transparency for both the provider and consumer of the utilized service.

1.1.3 Examples of Cloud Computing services:

- Dropbox
- Gmail
- Facebook

- Google-Drive
- Google App Engine
- Amazon Web Services (AWS) etc.

1.2 Definition of “Cloud”:

- The word "**cloud**" often refers to the Internet, which more precisely means a data center full of servers connected to the Internet performing a service.
- A cloud can be a wide area network (WAN) like the public Internet or a private network of any size, local or global.

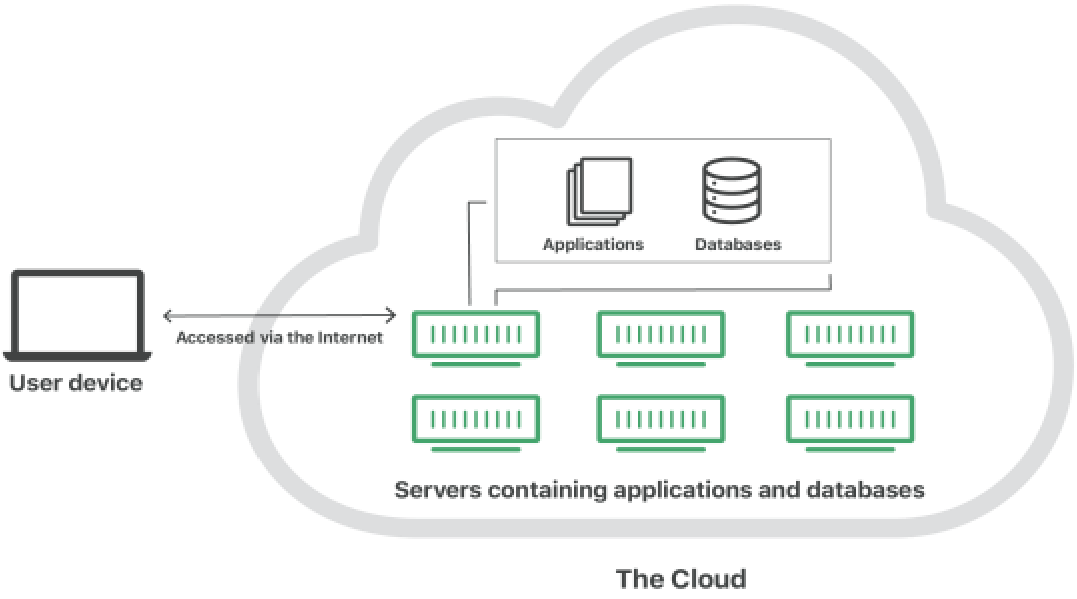

Applications
Databases
User device
Accessed via the Internet
Servers containing applications and databases
The Cloud

Fig 1.2 The “Cloud”

1.3 Evolution of Cloud-Computing:

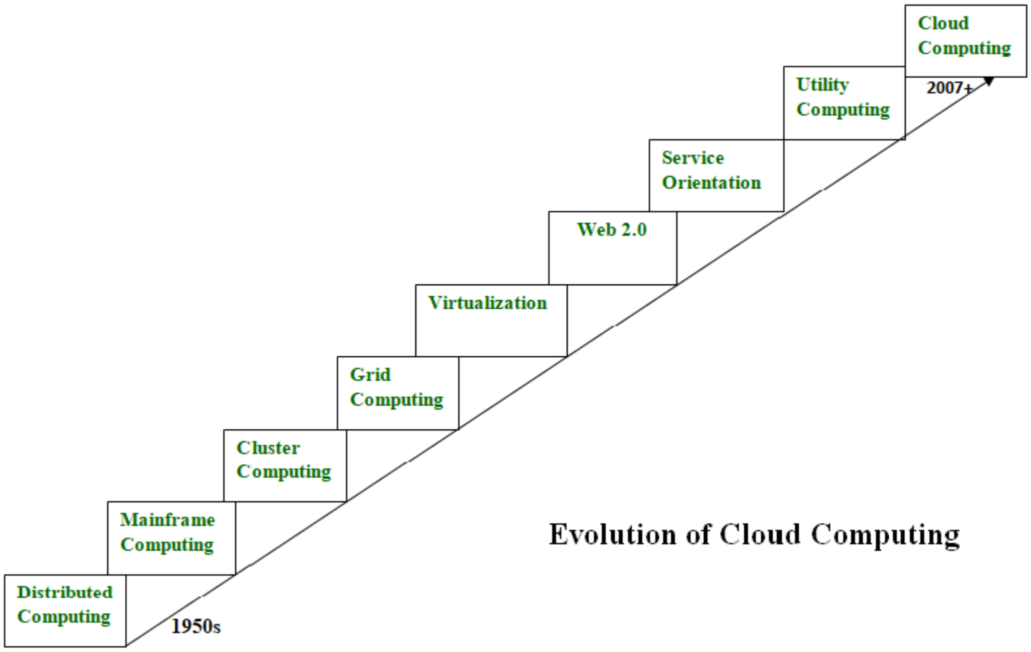


Fig 1.3 Evolution of Cloud-Computing

- **Distributed Systems:**

  It is a composition of multiple independent systems but all of them are depicted as a single entity to the users. The purpose of distributed systems is to share resources and also use them effectively and efficiently. Distributed systems possess characteristics such as scalability, concurrency, continuous availability, heterogeneity, and independence in failures.

  Problems: The main problem with this system was that all the systems were required to be present at the same geographical location.

  Solution: Thus, to solve this problem, distributed computing led to three more types of computing and they were-Mainframe computing, cluster computing, and grid computing.

- **Mainframe computing:**

  Mainframes which first came into existence in 1951 are highly powerful and reliable computing machines. These are responsible for handling large data such as massive input-output operations. Even today these are used for bulk processing tasks such as online transactions etc.

  Problems: These systems have almost no downtime with high fault tolerance. After distributed computing, these increased the processing capabilities of the system. But these were very expensive.

  Solution: To reduce this cost, cluster computing came as an alternative to mainframe technology.

- **Cluster computing:**

  In 1980s, cluster computing came as an alternative to mainframe computing. Each machine in the cluster was connected to each other by a network with high bandwidth. These were way cheaper than those mainframe systems.

  Problems: These were equally capable of high computations. Also, new nodes could easily be added to the cluster if it were required. Thus, the problem of the cost was solved to some extent, but the problem related to geographical restrictions still pertained.

  Solution: To solve this, the concept of grid computing was introduced.

- **Grid computing:**

  In 1990s, the concept of grid computing was introduced. It means that different systems were placed at entirely different geographical locations and these all were connected via the internet. These systems belonged to different organizations and thus the grid consisted of heterogeneous nodes.

Problems: Although it solved some problems, but new problems emerged as the distance between the nodes increased.

The main problem which was encountered was the low availability of high bandwidth connectivity and with-it other network associated issues.

Solution: Thus. cloud computing is often referred to as ***"Successor of grid computing".***

- **Virtualization:**

  It was introduced nearly 40 years back. It refers to the process of creating a virtual layer over the hardware which allows the user to run multiple instances simultaneously on the hardware. It is a key technology used in cloud computing. It is the base on which major cloud computing services such as Amazon EC2, VMware vCloud, etc work on. Hardware virtualization is still one of the most common types of virtualization.

- **Web 2.0:**

  It is the interface through which the cloud computing services interact with the clients. It is because of Web 2.0 that we have interactive and dynamic web pages. It also increases flexibility among web pages. Popular examples of web 2.0 include Google Maps, Facebook, Twitter, etc. Needless to say, social media is possible because of this technology only. In gained major popularity in 2004.

- **Service orientation:**

  It acts as a reference model for cloud computing. It supports low-cost, flexible, and evolvable applications. Two important concepts were introduced in this computing model. These were Quality of Service (QoS) which also includes the SLA (Service Level Agreement) and Software as a Service (SaaS).

- **Utility computing:**

  It is a computing model that defines service provisioning techniques for services such as compute services along with other major services such as storage, infrastructure, etc which are provisioned on a pay-per-use basis.

  Thus, the above technologies contributed to the making of cloud computing.

1.4 Underlying Principles of Parallel and Distributed Computing

1.4.1 Introduction to Serial Computing:

Initially, software has been written for serial computation.

- To be run on a single computer having a single Central Processing Unit.
- A problem is broken into a discrete series of instructions.
- Instructions are executed one after another.
- Only one instruction may execute at any moment in time

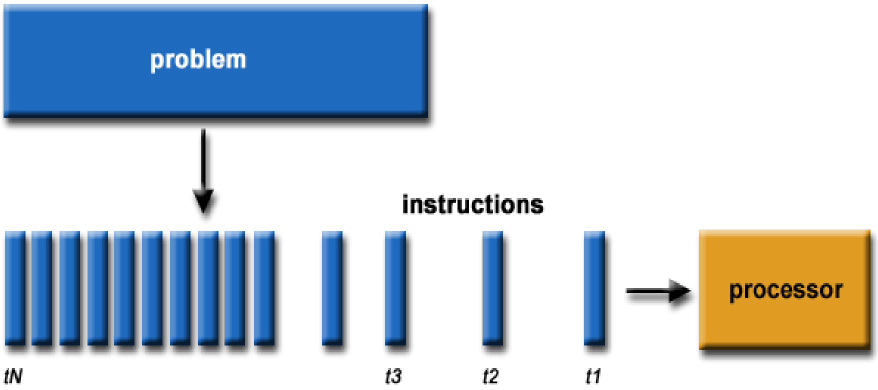


Fig 1.4 Serial Computing

1.4.2 Parallel Computing:

In the simplest sense, **Parallel Computing** is the simultaneous use of multiple compute resources to solve a computational problem:

- A problem is broken into discrete parts that can be solved concurrently

- Each part is further broken down to a series of instructions
- Instructions from each part execute simultaneously on different processors
- An overall control/coordination mechanism is employed

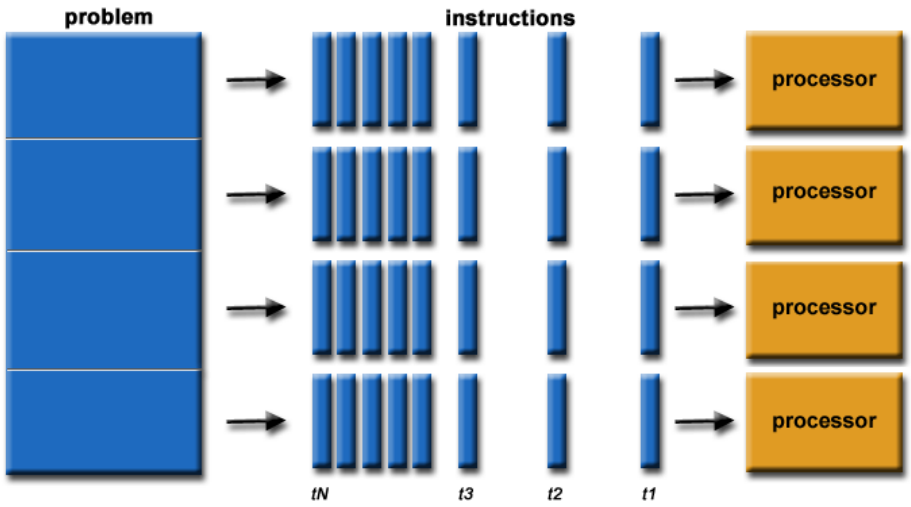


Fig 1.5 Parallel Computing

1.4.3 Why we use Parallel Computing:

- Save time and/or money:
- Solve larger / more complex problems:
- Provide concurrency:
- Take advantage of non-local resources:
- Make better use of underlying parallel hardware

1.4.4 Terminologies used in Parallel Computing:

Some of the more commonly used terms associated with parallel computing are listed below.

a) Supercomputing / High Performance Computing (HPC):

Using the world's fastest and largest computers to solve large problems.

b) Node:

A standalone "computer in a box". Usually comprised of multiple CPUs/processors/cores, memory, network interfaces, etc. Nodes are networked together to comprise a supercomputer.

c) CPU / Socket / Processor / Core:

- This varies, depending upon the task.
- In the past, a CPU (Central Processing Unit) was a singular execution component for a computer.
- Then, multiple CPUs were incorporated into a node.
- Then, individual CPUs were subdivided into multiple "cores", each being a unique execution unit.

d) Task:

- A logically discrete section of computational work.
- A task is typically a program or program-like set of instructions that is executed by a processor.
- A parallel program consists of multiple tasks running on multiple processors.

e) Pipelining:

Breaking a task into steps performed by different processor units, with inputs streaming through, much like an assembly line;

f) Shared Memory:

- From a strictly hardware point of view, describes a computer architecture where all processors have direct (usually bus based) access to common physical memory.

- In a programming sense, it describes a model where parallel tasks all have the same "picture" of memory and can directly address and access the same logical memory locations regardless of where the physical memory actually exists.

g) Symmetric Multi-Processor (SMP):

Shared memory hardware architecture where multiple processors share a single address space and have equal access to all resources.

h) Distributed Memory:

In hardware, refers to network-based memory access for physical memory that is not common. As a programming model, tasks can only logically "see" local machine memory and must use communications to access memory on other machines where other tasks are executing.

i) Communications:

Parallel tasks typically need to exchange data. There are several ways this can be accomplished, such as through a shared memory bus or over a network, however the actual event of data exchange is commonly referred to as communications regardless of the method employed.

j) Synchronization:

- The coordination of parallel tasks in real time, very often associated with communications.
- Often implemented by establishing a synchronization point within an application where a task may not proceed further until another task(s) reaches the same or logically equivalent point.
- Synchronization usually involves waiting by at least one task and can therefore cause a parallel application's wall clock execution time to increase.

k) Parallel Overhead:

The amount of time required to coordinate parallel tasks, as opposed to doing useful work. Parallel overhead can include factors such as:

- Task start-up time
- Synchronizations
- Data communications
- Software overhead imposed by parallel languages, libraries, operating system, etc.
- Task termination time

1.5 Cloud Characteristics

Cost: Cloud computing eliminates the capital expense of buying hardware and software and sing up and running on-site datacenters—the racks of servers, the round-the-clock electricity for power and cooling, the IT experts for managing the infrastructure. It adds up fast.

Speed: Most cloud computing services are provided self service and on demand, so even vast amounts of computing resources can be provisioned in minutes, typically with just a few mouse clicks, giving businesses a lot of flexibility and taking the pressure off capacity planning.

Global scale: The benefits of cloud computing services include the ability to scale elastically. In cloud speak, that means delivering the right amount of IT resources—for example, computing power, storage, bandwidth—right when it is needed and from the right geographic location.

Productivity: On-site data centres typically require a lot of “racking and stacking”—hardware setup, software patching, and other time-consuming IT management chores. Cloud computing removes the need for many of these tasks, so IT teams can spend time on achieving more important business goals.

Performance: The biggest cloud computing services run on a worldwide network of secure data centers, which are regularly upgraded to the latest generation of fast and efficient computing hardware. This offers several benefits over a single corporate datacenter, including reduced network latency for applications and greater economies of scale.

Reliability: Cloud computing makes data backup, disaster recovery and business continuity easier and less expensive because data can be mirrored at multiple redundant sites on the cloud provider’s network.

Security: Many cloud providers offer a broad set of policies, technologies and controls that strengthen your security posture overall, helping protect your data, apps and infrastructure from potential threats.

1.6 Elasticity in Cloud:

- In cloud computing, elasticity is defined as **"*the degree to which a system is able to adapt to workload changes by provisioning and de-provisioning resources in an autonomic manner, such that at each point in time the available resources match the current demand as closely as possible".***

- The purpose of **elasticity** is to match the resources allocated with actual amount of resources needed at any given point in time.

1.7 On-Demand provisioning:

1.7.1Cloud-Provisioning:

- In general, **provisioning means** "providing" or making something available.

- Cloud provisioning refers to the **“*Processes for the deployment and integration of cloud computing services within an enterprise IT infrastructure.”***

- Cloud provisioning primarily defines how, what and when an organization will provision (provide) cloud services. These services can be internal, public or hybrid cloud products and solutions.
- For example, the creation of virtual machines, the allocation of storage capacity and/or granting access to cloud software.

Types of Provisioning:

Provisioning can be categorized as follows:

1. **Over-Provisioning:**

   over-**provisioning** of **cloud** resources represents unused resources which represents a zero-ROI (return on investment) expense.

   Over-provisioning of cloud resources has, in the absence of other choices, become an epidemic. The result is that many organizations are investing in cloud resources they simply do not use. Unused resources produce, of course, a return on investment (ROI) of exactly zero.

2. **Under Provisioning:**

   Under-provisioning, i.e., allocating fewer resources than required, must be avoided, otherwise the service cannot serve its users with a good service. Under-provisioning the website may make it seem slow or unreachable.

1.7.3 On-Demand Provisioning:

- On-demand provisioning is a delivery model in which computing resources are made available to the user as needed. The resources may be maintained within the user's enterprise or made available by a cloud service provider.
- The customer or requesting application is provided with resources on run time.

Important Questions:

1. What is cloud and Cloud Computing?
2. Explain Benefits of Cloud Computing.
3. Explain basic applications of Cloud Computing.
4. Give the various characteristics of Cloud Computing.
5. Briefly explain the evolution of Cloud-Computing
6. Explain Cloud Computing strategy and planning's.
7. Differentiate between distributed Computing and Cloud Computing.
8. What are the major challenges faced in cloud?
9. Differentiate between Grid Computing and Cloud Computing.
10. Give the various Principles of Parallel and Distributed Computing.
11. What do you understand by Cloud-Provisioning? Give its Types.
12. What do you understand by On-demand Provisioning?
13. What do you understand by Cloud-Elasticity?

UNIT-2

CLOUD ENABLING TECHNOLOGIES

2.1 Service Oriented Architecture:

Service-Oriented Architecture (SOA) is an architectural approach in which applications make use of services available in the network. In this architecture, services are provided to form applications, through a communication call over the internet.

- SOA allows users to combine a large number of facilities from existing services to form applications.
- SOA encompasses a set of design principles that structure system development and provide means for integrating components into a coherent and decentralized system.
- SOA based computing packages functionalities into a set of interoperable services, which can be integrated into different software systems belonging to separate business domains.

2.1.1 History

- The first report published on SOA by the analysts **Roy W.Schulte** and **Yefim V.Natis** in 1996.
- The term first appeared in 1998, and since then it's grown in popularity. It's also branched into several variants, including microservice architecture.
- Service-oriented architecture's definition has always been a moving target. It evolved for several years before The Open Group published a white paper in 2007.
- In 2005, before The Open Group published its Source Book, Martin Fowler called SOA

2.1.2 Why to use SOA?

- SOA is widely used in market which responds quickly and makes effective changes according to market situations.

- The SOA keep secret the implementation details of the subsystems.
- It allows interaction of new channels with customers, partners and suppliers.
- It authorizes the companies to select software or hardware of their choice as it acts as platform independence

2.1.3 Implementing Service-Oriented Architecture

To implement SOA, you start with the basic service architecture, then provide the infrastructure, meaning protocols and other tools that enable communication and interoperability. Figure 2 shows a diagram of a typical service architecture.

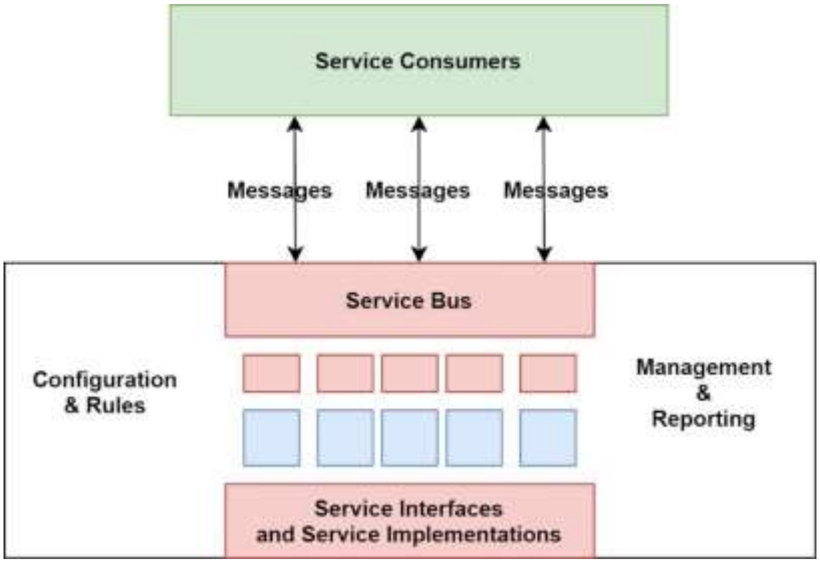


Fig 2.1 Implementing Service-Oriented Architecture

In this diagram, three consumers invoke services by sending messages to an enterprise service bus, which transforms and routes the messages to an appropriate service implementation. A *business rules engine* incorporates business rules in a service or across services. A *service management layer* manages activities like auditing, billing, and logging.

Components in this architecture are loosely coupled, so they can be switched out or updated with relatively minimal impact to the application as a whole. This gives the enterprise flexibility to add or update business processes as needed. For the most part, changes to individual services should not greatly affect other services.

There are two major roles within Service-oriented Architecture:

1. **Service provider:** The service provider is the maintainer of the service and the organization that makes available one or more services for others to use. To advertise services, the provider can publish them in a registry, together with a service contract that specifies the nature of the service, how to use it, the requirements for the service, and the fees charged.

2. **Service consumer:** The service consumer can locate the service metadata in the registry and develop the required client components to bind and use the service.

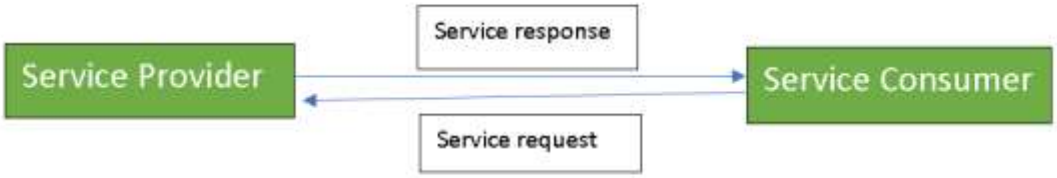


Fig 2.2 Roles within SOA

2.1.4 Practical Examples of SOA:

- Banking Applications
- Designing web pages
- Using location services within mobile
- Summarizing Databases etc.

2.1.5 Guiding Principles of SOA:

1. **Standardized service contract:** Specified through one or more service description documents.

2. **Loose coupling:** Services are designed as self-contained components, maintain relationships that minimize dependencies on other services.

3. **Abstraction:** A service is completely defined by service contracts and description documents. They hide their logic, which is encapsulated within their implementation.

4. **Reusability:** Designed as components, services can be reused more effectively, thus reducing development time and the associated costs.

5. **Autonomy:** Services have control over the logic they encapsulate, and, from a service consumer point of view, there is no need to know about their implementation.

6. **Discoverability:** Services are defined by description documents that constitute supplemental metadata through which they can be effectively discovered. Service discovery provides an effective means for utilizing third-party resources.

7. **Composability:** Using services as building blocks, sophisticated and complex operations can be implemented. Service orchestration and choreography provide solid support for composing services and achieving business goals.

2.1.6 Advantages of SOA:

- **Service reusability:** In SOA, applications are made from existing services. Thus, services can be reused to make many applications.

- **Easy maintenance:** As services are independent of each other they can be updated and modified easily without affecting other services.

- **Platform independent:** SOA allows making a complex application by combining services picked from different sources, independent of the platform.

- **Availability:** SOA facilities are easily available to anyone on request.

- **Reliability:** SOA applications are more reliable because it is easy to debug small services rather than huge codes.
- **Scalability:** Services can run on different servers within an environment, this increases scalability

2.1.7 Disadvantages of SOA:

- **High overhead:** A validation of input parameters of services is done whenever services interact this decreases performance as it increases load and response time.

- **High investment:** A huge initial investment is required for SOA.

- **Complex service management:** When services interact, they exchange messages to tasks. the number of messages may go in millions. It becomes a cumbersome task to handle a large number of messages.

2.2 REST (REpresentational State Transfer):

- Representational state transfer is a software architectural style that defines a set of constraints to be used for creating Web services.
- Web services that conform to the REST architectural style, called RESTful Web services, provide interoperability between computer systems on the internet.

2.2.1 Guiding Principles of REST:

1. **Client–server** – By separating the user interface concerns from the data storage concerns, we improve the portability of the user interface across multiple platforms and improve scalability by simplifying the server components.

2. **Stateless** – Each request from client to server must contain all of the information necessary to understand the request and cannot take advantage of any stored context on the server. Session state is therefore kept entirely on the client.
3. **Cacheable** – Cache constraints require that the data within a response to a request be implicitly or explicitly labelled as cacheable or non-cacheable. If a response is cacheable, then a client cache is given the right to reuse that response data for later, equivalent requests.
4. **Uniform interface** – By applying the software engineering principle of generality to the component interface, the overall system architecture is simplified, and the visibility of interactions is improved. In order to obtain a uniform interface, multiple architectural constraints are needed to guide the behaviour of components. REST is defined by four interface constraints: identification of resources; manipulation of resources through representations; self-descriptive messages; and hypermedia as the engine of application state.
5. **Layered system** – The layered system style allows an architecture to be composed of hierarchical layers by constraining component behaviour such that each component cannot “see” beyond the immediate layer with which they are interacting.
6. **Code on demand (optional)** – REST allows client functionality to be extended by downloading and executing code in the form of applets or scripts. This simplifies clients by reducing the number of features required to be pre-implemented.

2.2.2 Resource

The key abstraction of information in REST is a **resource**. Any information that can be named can be a resource: a document or image, a temporal service, a collection of other resources, a non-virtual object (e.g., a person), and so on. REST uses a **resource identifier** to identify the particular resource involved in an interaction between components.

The state of the resource at any particular timestamp is known as **resource representation**. A representation consists of data, metadata describing the data and **hypermedia** links which can help the clients in transition to the next desired state.

The data format of a representation is known as a **media type**. The media type identifies a specification that defines how a representation is to be processed.

A truly RESTful API looks like *hypertext*. Every addressable unit of information carries an address, either explicitly (e.g., link and id attributes) or implicitly (e.g., derived from the media type definition and representation structure).

Further, **resource representations shall be self-descriptive**: the client does not need to know if a resource is employee or device. It should act on the basis of media-type associated with the resource. So, in practice, you will end up creating lots of **custom media-types** – normally one media-type associated with one resource.

2.2.3 REST and HTTP are not same!!

A lot of people prefer to compare HTTP with REST. **REST and HTTP are not same.**

REST ! = HTTP

Though, because REST also intends to make the web (internet) more streamline and standard, he advocates using REST principles more strictly. And that’s from where people try to start comparing REST with web (HTTP).

In simplest words, in the REST architectural style, data and functionality are considered resources and are accessed using Uniform Resource Identifiers (URIs). The resources are acted upon by using a set of simple, well-defined operations. The clients and servers exchange representations of resources by using a standardized interface and protocol – typically HTTP.

Resources are decoupled from their representation so that their content can be accessed in a variety of formats, such as HTML, XML, plain text, PDF, JPEG, JSON, and others. Metadata about the resource is available and used, for example, to control caching, detect transmission errors, negotiate the appropriate representation format, and perform authentication or access control. And most importantly, every interaction with a resource is stateless.

All these principles help RESTful applications to be simple, lightweight, and fast.

2.3 System of Systems (SoS):

- A **system of systems** (SoS) is "a collection of **systems**, each capable of independent operation, that interoperate together to achieve additional desired capabilities."

- In the data center, independent constituent parts of a large system are connected through SoS-defined software interfaces called middleware. Such programs ensure that constituents do not compete for subtasks within the larger system and provide messaging services so that constituent systems can communicate.

- **Cloud-Computing** is also a System of Systems approach to computing that provides a single platform to access the computing power of many physical machines.

Systems of systems typically exhibit the behaviours of complex systems, but not all complex problems fall in the realm of systems of systems. Inherent to system of systems problems are several combinations of traits, not all of which are exhibited by every such problem:

- Operational Independence of Elements
- Managerial Independence of Elements
- Evolutionary Development
- Emergent Behaviour

- Geographical Distribution of Elements
- Interdisciplinary Study
- Heterogeneity of Systems
- Networks of Systems

The first five traits are known as Maier's criteria for identifying system of systems challenges. The remaining three traits have been proposed from the study of mathematical implications of modeling and analysing system of systems challenges by Dr. Daniel DeLaurentis and his co-researchers at Purdue University.

2.4 Web service:

- **Web service** is a standardized medium to propagate communication between the client and server applications on the World Wide Web. A web service is a software module that is designed to perform a certain set of tasks.
- It is a collection of open protocols and standards used for exchanging data between applications or systems.

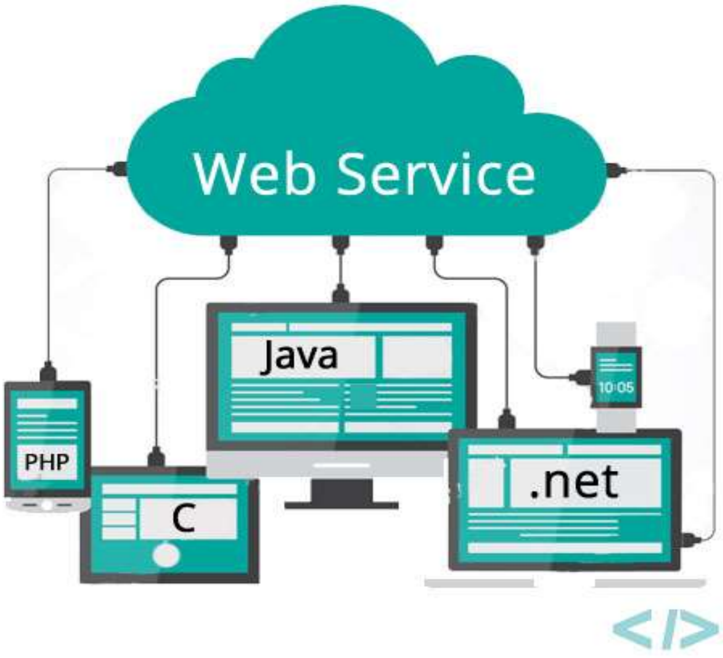


Fig 2.3 Web Services

2.4.1 Web Service Features:

- XML-Based
- Loosely Coupled
- Ability to be Synchronous or Asynchronous
- Supports Remote Procedure Calls (RPCs)
- Supports Document Exchange

2.4.2 Components of Web Services:

1. **Publish():** The first phase is the Publish() phase where a Service Provider feeds all the details about a Web Service in a Service Registry or Repository.

2. **Find():** The second phase is Find() where a Service Request mainly the client application finds the details about Web Service from a repository (also has WSDL XML file).
3. **Bind():** The last phase is Binding() where the client application or the Service Requester synchronizes with the Service Provider for the final implementation of the Web Service.

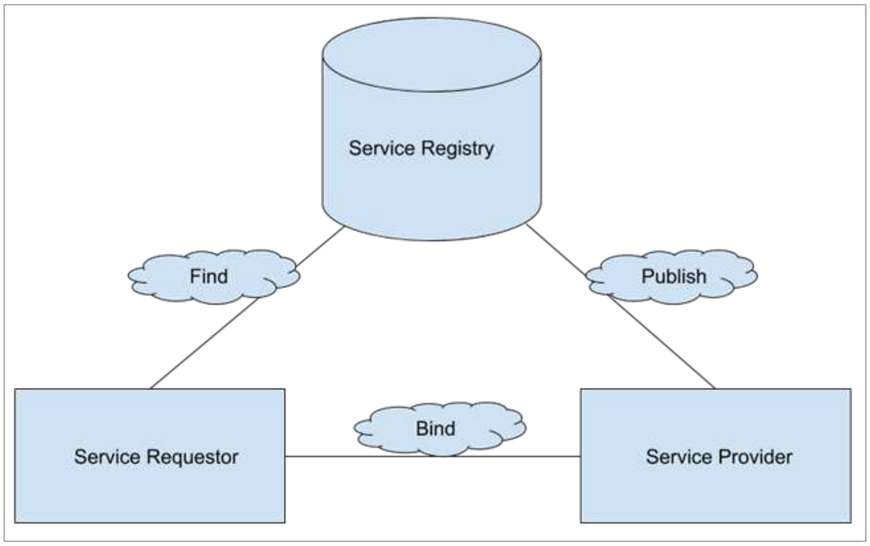
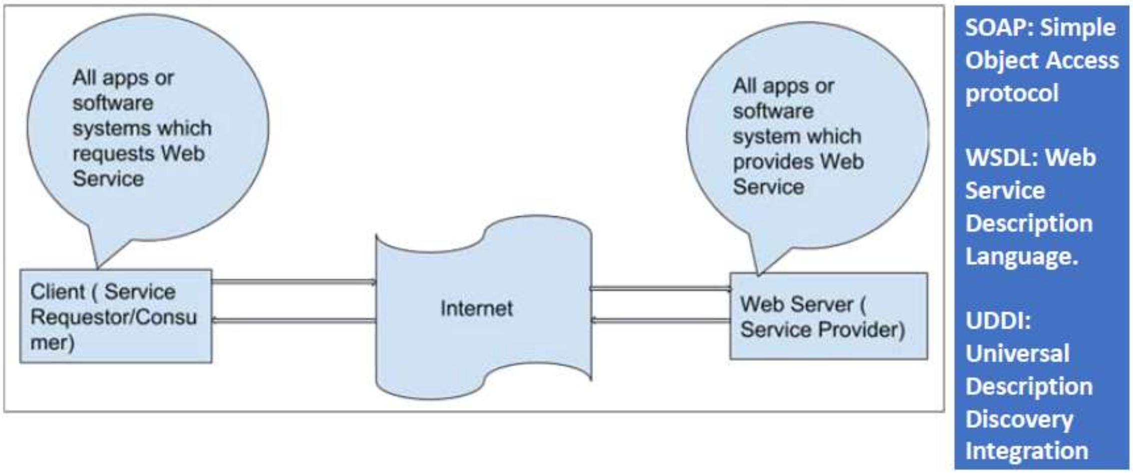


Fig 2.4 Components of web services

2.4.3 Working of Web Services:


All apps or software systems which requests Web Service
Client (Service Requestor/Consumer)
Internet
All apps or software system which provides Web Service
Web Server (Service Provider)

SOAP: Simple Object Access protocol

WSDL: Web Service Description Language.

UDDI: Universal Description Discovery Integration

Fig 2.5 Working of Web Services

- The client would invoke a series of web service calls via requests to a server which would host the actual web service.
- These requests are made through what is known as remote procedure calls. Remote Procedure Calls (RPC) are calls made to methods which are hosted by the relevant web service.
- The main component of a web service is the data which is transferred between the client and the server, and that is XML.
- Web services use something known as SOAP (Simple Object Access Protocol) for sending the XML data between applications.
- The data which is sent from the web service to the application is called a SOAP message. The SOAP message is nothing but an XML document.

2.4.4 Advantages:

- Interoperability
- Usability
- Reusability
- Deployability

2.5 Publisher- Subscriber (Pub/Sub) Model:

In a pub/sub model, any message published to a topic is immediately received by all of the subscribers to the topic.

2.5.1 Context and problem (why use Pub/Sub model):

In cloud-based and distributed applications, components of the system often need to provide information to other components as events happen.

Asynchronous messaging is an effective way to decouple senders from consumers and avoid blocking the sender to wait for a response. However, using a dedicated message queue for each consumer does not effectively scale to many consumers. Also, some of the consumers might be interested in only a subset of the information. How can the sender announce events to all interested consumers without knowing their identities?

2.5.2 Components of Publisher- Subscriber (Pub/Sub) Model:

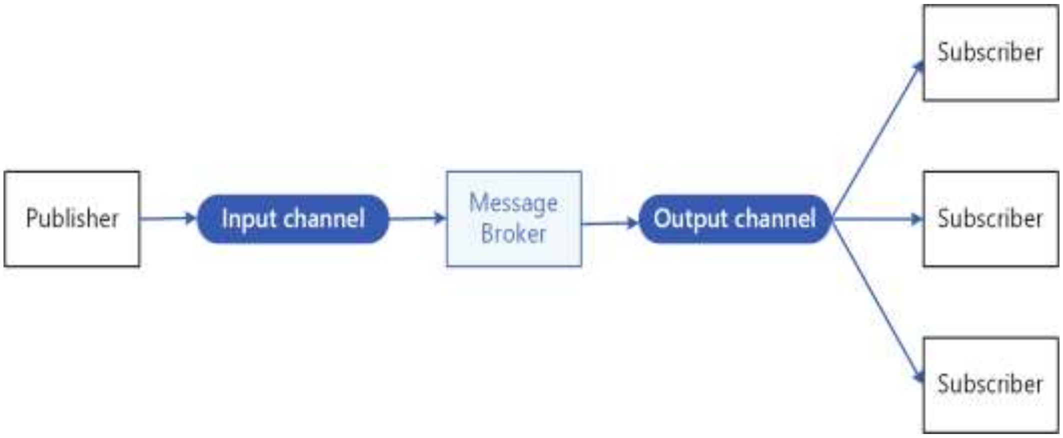


Fig 2.6 Components of Pub/Sub Model:

Publisher: An input messaging channel used by the sender. The sender packages events into messages, using a known message format, and sends these messages via the input channel. The sender in this pattern is also called the *publisher*.

Subscriber: One output messaging channel per consumer. The consumers are known as *subscribers*.

Message Broker: A mechanism for copying each message from the input channel to the output channels for all subscribers interested in that message. This operation is typically handled by an intermediary such as a message broker or event bus.

2.5.3 Benefits of Publisher-Subscriber Model:

- It decouples subsystems that still need to communicate. Subsystems can be managed independently, and messages can be properly managed even if one or more receivers are offline.
- It increases scalability and improves responsiveness of the sender. The sender can quickly send a single message to the input channel, then return to its core processing responsibilities. The messaging infrastructure is responsible for ensuring messages are delivered to interested subscribers.
- It improves reliability. Asynchronous messaging helps applications continue to run smoothly under increased loads and handle intermittent failures more effectively.
- It allows for deferred or scheduled processing. Subscribers can wait to pick up messages until off-peak hours, or messages can be routed or processed according to a specific schedule.
- It enables simpler integration between systems using different platforms, programming languages, or communication protocols, as well as between on-premises systems and applications running in the cloud.
- It facilitates asynchronous workflows across an enterprise.
- It improves testability. Channels can be monitored and messages can be inspected or logged as part of an overall integration test strategy.
- It provides separation of concerns for your applications. Each application can focus on its core capabilities, while the messaging infrastructure handles everything required to reliably route messages to multiple consumers.

2.5.4 Issues and considerations:

Consider the following points when deciding how to implement this pattern:

- **Existing technologies.** It is strongly recommended to use available messaging products and services that support a publish-subscribe model, rather than building your own. In Azure, consider using Service Bus or Event Grid. Other technologies that can be used for pub/sub messaging include Redis, RabbitMQ, and Apache Kafka.
- **Subscription handling.** The messaging infrastructure must provide mechanisms that consumers can use to subscribe to or unsubscribe from available channels.
- **Security.** Connecting to any message channel must be restricted by security policy to prevent eavesdropping by unauthorized users or applications.
- **Subsets of messages.** Subscribers are usually only interested in subset of the messages distributed by a publisher. Messaging services often allow subscribers to narrow the set of messages received by:
- **Topics.** Each topic has a dedicated output channel, and each consumer can subscribe to all relevant topics.
- **Content filtering.** Messages are inspected and distributed based on the content of each message. Each subscriber can specify the content it is interested in.
- **Wildcard subscribers.** Consider allowing subscribers to subscribe to multiple topics via wildcards.
- **Bi-directional communication.** The channels in a publish-subscribe system are treated as unidirectional. If a specific subscriber needs to send acknowledgment or communicate status back to the publisher, consider using the Request/Reply Pattern. This pattern uses one channel to send a message to the subscriber, and a separate reply channel for communicating back to the publisher.
- **Message ordering.** The order in which consumer instances receive messages isn't guaranteed and doesn't necessarily reflect the order in which the messages were created. Design the system to ensure that message processing is idempotent to help eliminate any dependency on the order of message handling.

- **Message priority.** Some solutions may require that messages are processed in a specific order. The Priority Queue pattern provides a mechanism for ensuring specific messages are delivered before others.
- **Poison messages.** A malformed message, or a task that requires access to resources that are not available, can cause a service instance to fail. The system should prevent such messages being returned to the queue. Instead, capture and store the details of these messages elsewhere so that they can be analyzed if necessary.
- **Repeated messages.** The same message might be sent more than once. For example, the sender might fail after posting a message. Then a new instance of the sender might start up and repeat the message. The messaging infrastructure should implement duplicate message detection and removal (also known as de-duping) based on message IDs in order to provide at-most-once delivery of messages.
- **Message expiration.** A message might have a limited lifetime. If it isn't processed within this period, it might no longer be relevant and should be discarded. A sender can specify an expiration time as part of the data in the message. A receiver can examine this information before deciding whether to perform the business logic associated with the message.
- **Message scheduling.** A message might be temporarily embargoed and should not be processed until a specific date and time. The message should not be available to a receiver until this time.

2.6 Basics of Virtualization:

2.6.1 Virtualization:

- Virtualization is the "creation of a virtual (rather than actual) version of something, such as a server, a desktop, a storage device, an operating system or network resources".
- It is a technique, which allows to share a single physical instance of a resource or an application among multiple customers and organizations.

- Virtualization is a large umbrella of technologies and concepts that are meant to provide an abstract environment—whether this is virtual hardware or operating system—to run applications.
- This term is often synonymous with *hardware virtualization*, which plays fundamental role in efficiently delivering *Infrastructure-as-a-Service* solutions for Cloud computing.

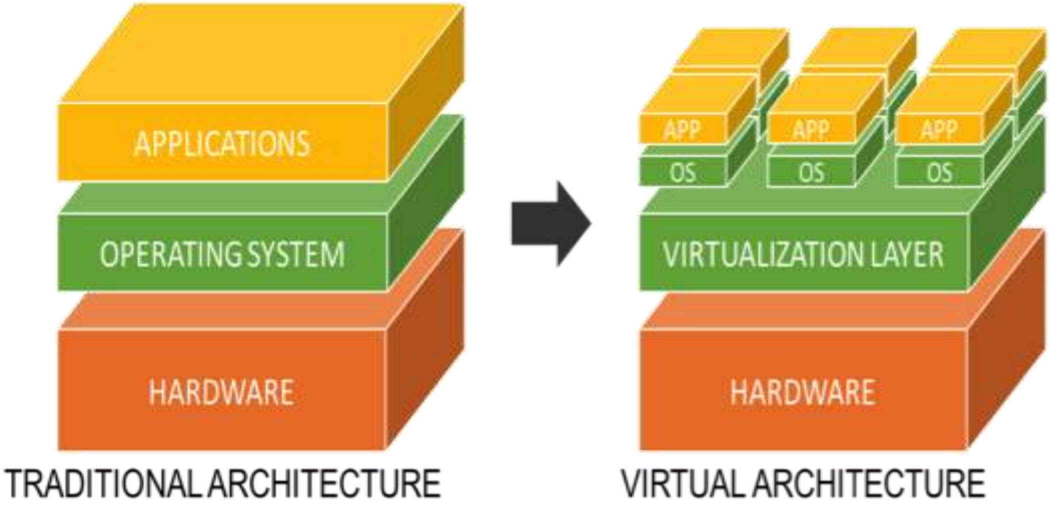


Fig 2.6 Virtualization

2.6.2 Need of Virtualization:

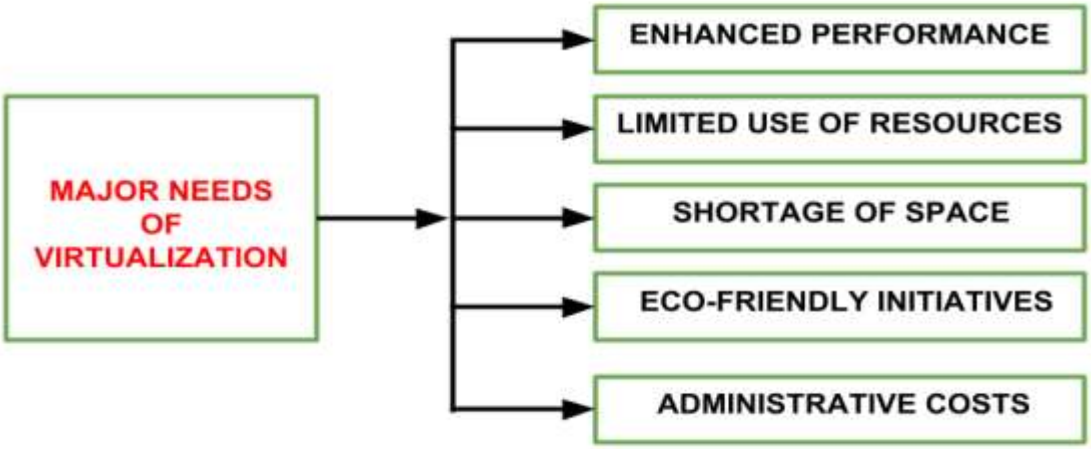


Fig 2.7 Need of Virtualization

Enhanced Performance:

Currently, the end user system i.e., PC is sufficiently powerful to fulfil all the basic computation requirements of the user, with various additional capabilities which are rarely used by the user. Most of their systems have sufficient resources which can host a virtual machine manager and can perform a virtual machine with acceptable performance so far.

Limited use of Hardware and Software Resources:

The limited use of the resources leads to under-utilization of hardware and software resources. As all the PCs of the user are sufficiently capable to fulfill their regular computational needs that’s why many of their computers are used often which can be used 24/7 continuously without any interruption. The efficiency of IT infrastructure could be increase by using these resources after hours for other purposes. This environment is possible to attain with the help of Virtualization.

Shortage of Space:

The regular requirement for additional capacity, whether memory storage or compute power, leads data centers raise rapidly. Companies like Google, Microsoft and Amazon develop their infrastructure by building data centers as per their needs. Mostly, enterprises unable to pay to build any other data center to accommodate additional resource capacity. This heads to the diffusion of a technique which is known as server consolidation.

Eco-Friendly Initiatives:

At this time, corporations are actively seeking for various methods to minimize their expenditures on power which is consumed by their systems. Data centers are main power consumers and maintaining a data center operations needs a continuous power supply as well as a good amount of energy is needed to keep them cool for well-functioning. Therefore, server consolidation drops the power consumed and cooling impact by having a fall in number of servers. Virtualization can provide a sophisticated method of **server consolidation.**

Administrative Costs:

Furthermore, the rise in demand for capacity surplus, that convert into more servers in a data center, accountable for a significant increase in administrative costs. Hardware monitoring, server setup and updates, defective hardware replacement, server resources monitoring, and backups are included in common system administration tasks. These are personnel-intensive operations. The administrative costs is increased as per the number of servers. Virtualization decreases number of required servers for a given workload, hence reduces the cost of administrative employees.

2.6.3 Virtualization Reference Model:

Virtualization is a broad concept, and it refers to the creation of a virtual version of something, whether this is hardware, software environment, storage, or network.

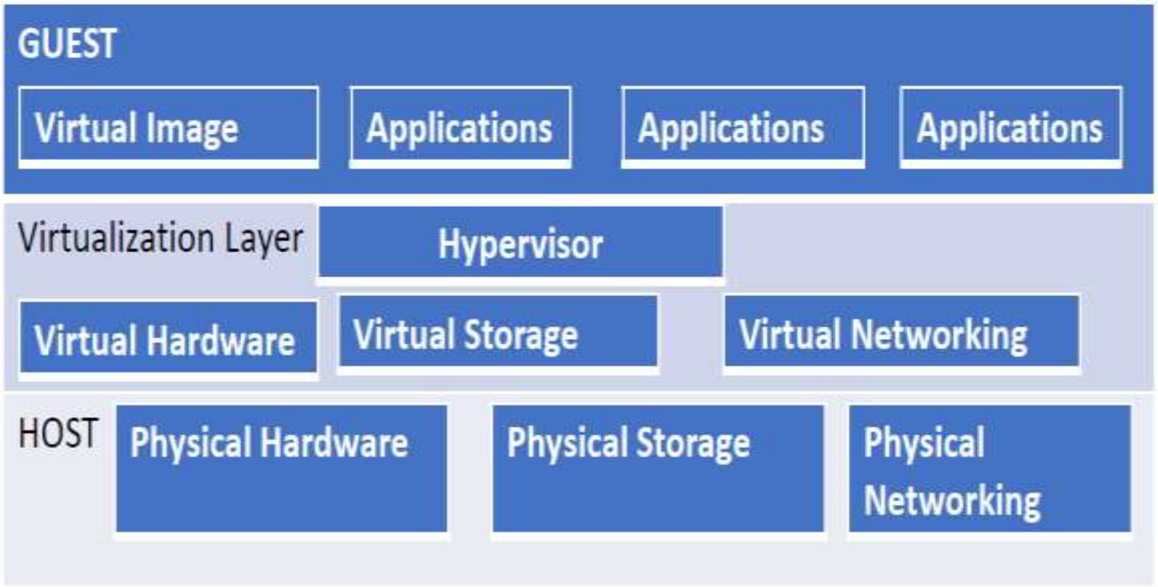


Fig 2.8 Virtualization Reference Model

Four major Components falls under this category in a virtualized environment:

1. **Guest:**

- The guest represents the system component that interacts with the virtualization layer rather than with the host.
- Guests usually consist of one or more virtual disk files, and a VM definition file. Virtual Machines are centrally managed by a host application that sees and manages each virtual machine as a different application.

2. **Host:**

- The host represents the original environment where the guest is supposed to be managed. Each guest runs on the host using shared resources donated to it by the host.
- The operating system works as the host and manages the physical resource management, and the device support.

3. **Virtualization Layer:**

- The virtualization layer is responsible for recreating the same or a different environment where the guest will operate.
- It is an additional abstraction layer between a network and storage hardware, computing, and the application running on it.

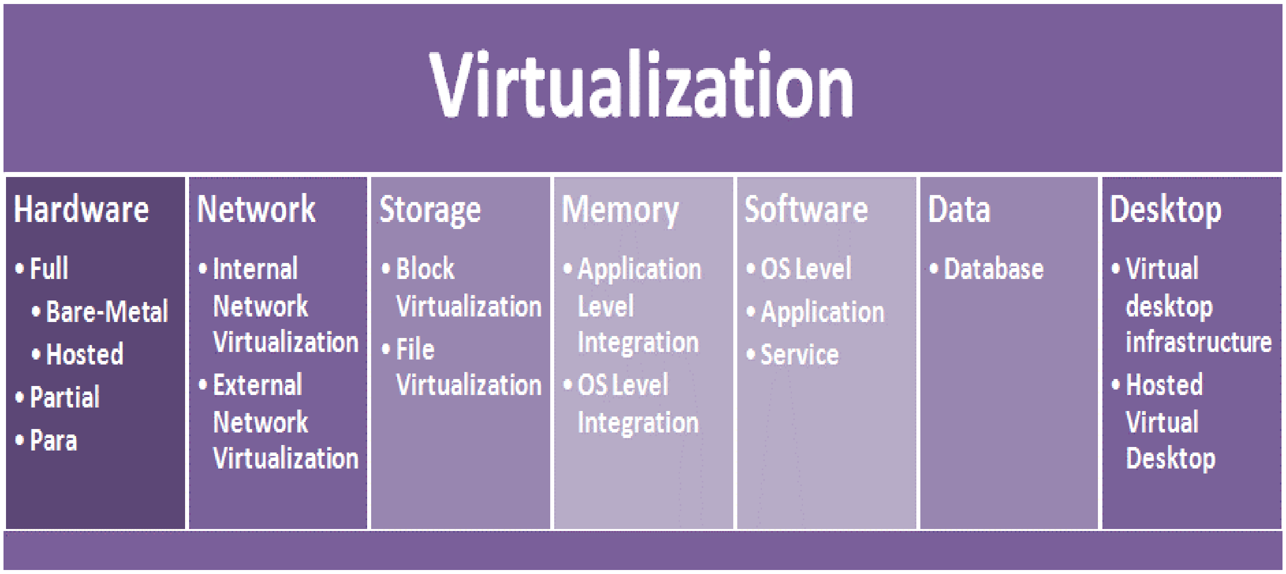
4. Hypervisor (VMM-Virtual Machine Monitor):

- A hypervisor, also known as a virtual machine monitor or VMM, is software that creates and runs virtual machines (VMs).
- A hypervisor allows one host computer to support multiple guest VMs by virtually sharing its resources, such as memory and processing.

2.6.4 Benefits of virtualization:

- More flexible and efficient allocation of resources.
- Enhance development productivity.
- It lowers the cost of IT infrastructure.
- Remote access and rapid scalability.
- High availability and disaster recovery.
- Pay per use of the IT infrastructure on demand.
- Enables running multiple operating system.

2.7 Types of Virtualization:



Virtualization

Hardware	Network	Storage	Memory	Software	Data	Desktop
• Full • Bare-Metal • Hosted • Partial • Para	• Internal Network Virtualization • External Network Virtualization	• Block Virtualization • File Virtualization	• Application Level Integration • OS Level Integration	• OS Level • Application • Service	• Database	• Virtual desktop infrastructure • Hosted Virtual Desktop

Fig 2.9 Types of Virtualization

1. Hardware Virtualization:

- Individual independent segment of hardware or a physical server, may be made up of multiple smaller hardware segments or servers, essentially consolidating multiple physical servers into virtual servers that run on a single primary physical server.

- Each small server can host a virtual machine, but the entire cluster of servers is treated as a single device by any process requesting the hardware.
- The hardware resource allotment is done by the hypervisor.

Types of Hardware Virtualization:

a) **Full Virtualization** – In it, the complete simulation of the actual hardware takes place to allow the software to run an unmodified guest OS.

b) **Para Virtualization** – In this type of virtualization, software unmodified runs in modified OS as a separate system.

c) **Partial Virtualization** – In this type of hardware virtualization, the software may need modification to run.

2. Network Virtualization

- Network virtualization is a method of combining the available resources in a network to consolidate multiple physical networks, divide a network into segments or create software networks between virtual machines (VMs).

Types of Network Virtualization:

a) **Internal:** Provide network-like functionality to a single system.

b) **External:** Combine many networks or parts of networks into a virtual unit.

3. Storage Virtualization:

- In this type of virtualization, multiple physical storage devices are grouped together, which then appear as a single storage device for easier and more efficient management of these resources.

Types of storage virtualization:

a) **Block Virtualization** – Multiple storage devices are consolidated into one

b) **File Virtualization** – Storage system grants access to files that are stored over multiple hosts.

4. Memory Virtualization:

- Physical memory across different servers is aggregated into a single virtualized memory pool.
- It enhances performance by providing greater memory capacity without any addition to the main memory.

Types of Memory Virtualization:

a) **Application-level integration –** Applications running on connected computers directly connect to the memory pool through an API or the file system.

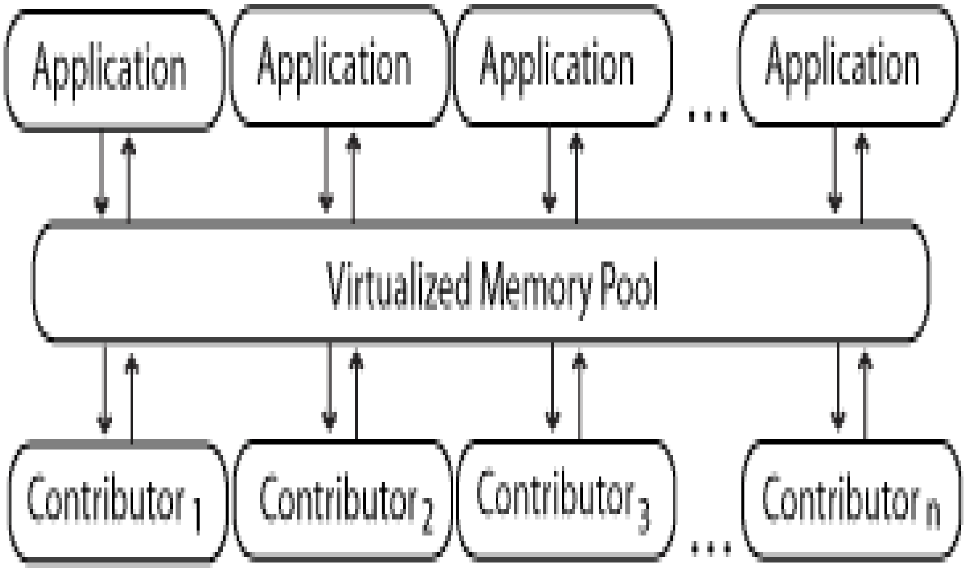


Fig 2.10 Application-level integration

b) **Operating System-Level Integration –** The operating system first connects to the memory pool and makes that pooled memory available to applications.

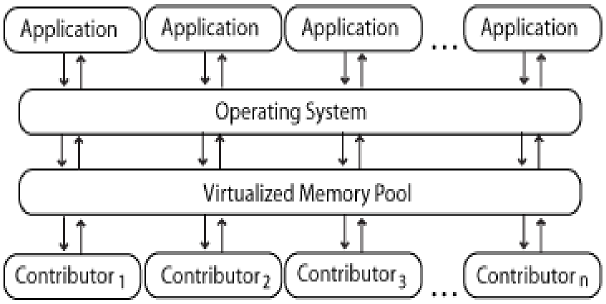


Fig 2.11 Application-level integration

5. Software Virtualization:

- It provides the ability to the main computer to run and create one or more virtual environments. It is used to enable a complete computer system in order to allow a guest OS to run.

Types of Software Virtualization:

a) **Operating System Virtualization** – hosting multiple OSs on the native OS

b) **Application Virtualization** – hosting individual applications in a virtual environment separate from the native OS.

c) **Service Virtualization** – hosting specific processes and services related to a particular application.

6. Data Virtualization:

- Data virtualization is the process of retrieve data from various resources without knowing its type and physical location where it is stored.
- It collects heterogeneous data from different resources and allows data users across the organization to access this data according to their work requirements.

- This heterogeneous data can be accessed using any application such as web portals, web services, E-commerce, Software as a Service (SaaS), and mobile application.

Advantages of Data Virtualization:

- It allows users to access the data without worrying about where it resides on the memory.
- It offers better customer satisfaction, retention, and revenue growth.
- It provides various security mechanism that allows users to safely store their personal and professional information.
- It reduces costs by removing data replication.
- It provides a user-friendly interface to develop customized views.
- It provides various simple and fast deployment resources.
- It increases business user efficiency by providing data in real-time.
- It is used to perform tasks such as data integration, business integration, data services, and enterprise search.

7. Desktop Virtualization:

- The user's desktop is stored on a remote server, which allows the user to access his desktop from any device or location.
- It provides work convenience and security as one can access remotely.
- It also protects confidential data from being lost or stolen by keeping it safe on central servers.

2.8 Implementation Level of Virtualization:

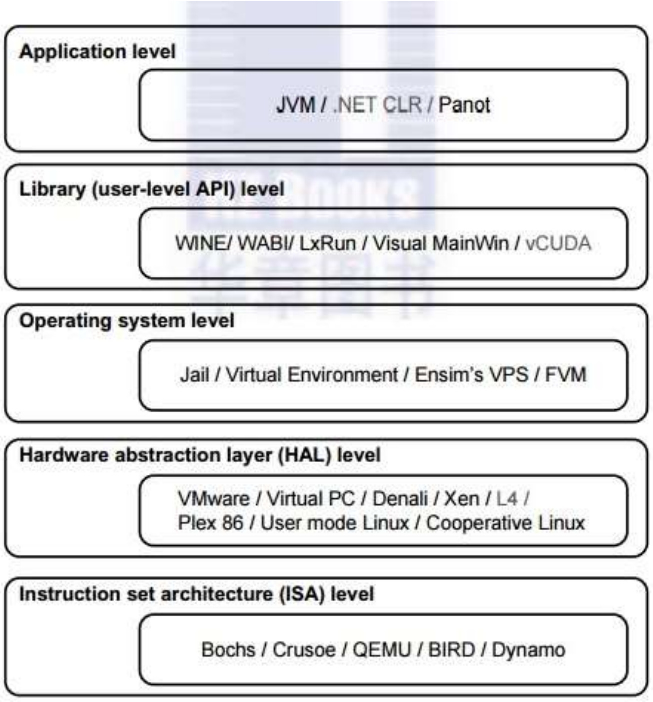

Application level
JVM / .NET CLR / Panot
Library (user-level API) level
WINE/ WABI/ LxRun / Visual MainWin / vCUDA
Operating system level
Jail / Virtual Environment / Ensim's VPS / FVM
Hardware abstraction layer (HAL) level
VMware / Virtual PC / Denali / Xen / L4 /
Plex 86 / User mode Linux / Cooperative Linux
Instruction set architecture (ISA) level
Bochs / Crusoe / QEMU / BIRD / Dynamo

Fig 2.12 Implementation Level of Virtualization

1. Virtualization at Instruction Set Architecture (ISA) level:

- **Instruction Set**: Every machine has an instruction set which is an interface between software and hardware.
- **Emulator:** An emulator is created which receives all the instructions from the Virtual machines, like for example if a virtual machine wants to access the printer then that instruction will be passed to this emulator.
- **Mapping of Instructions:** The emulator interprets what type of instruction it is and then map that instruction to the Host machine's instruction and then that instruction will be carried out on Host machine and the results will be passed to the emulator and emulator will return it to the virtual machine.

Problem: This technique is simple to implement but as every instruction has to be interpreted before mapping it, too much time is consumed, and performance becomes poor.

2. Virtualization at Hardware Abstraction Layer (HAL) level:

- **Mapping of virtual resources with physical resources:** In this type we map the virtual resources with the physical resources.
- **Distinguishing b/w privileged and non- privileged instructions:** We don't interpret every instruction, but we just check whether it is a privileged instruction or not.
- If the instruction is not privileged, we simply allow normal execution because already virtual and physical resources are mapped so accessing is simple.
- But if the instruction is privileged, we pass the control to VMM (Virtual Machine Monitor) and it deals with it accordingly

3. Virtualization at Operating System (O.S.) level:

- This refers to an abstraction layer between traditional OS and user applications.
- OS-level virtualization creates isolated containers on a single physical server and the OS instances to utilize the hard-ware and software in data centers.
- In virtualization at HAL level each virtual machine is built from scratch i.e., by installing O.S., application suites, networking systems, etc.
- So, to overcome this in Virtualization at Operating system level we share operating system between Virtual machines along with the hardware.
- So, we keep the base O.S. same and install only the differences in each single Virtual machine.
- For example, if we want to install different versions of windows on virtual machines (VM), you keep base O.S. of windows same and only install the differences among each VM.

4. Virtualization at Library Level or Programming language level:

- Virtualization with library interfaces is possible by controlling the communication link between applications and the rest of a system through APIs.
- We provide user with an emulator with which user can run applications of different O.S.s.
- Example of this is the WINE tool which was used mostly by mac users to play Counter Strike 1.6 game which was only available for windows in the start.

5. Virtualization at Application Layer level:

- In this kind of virtualization Virtual machines run as an application on the Host operating system.
- We create a virtualization layer which is present above the Host Operating system and it encapsulates all the applications from the underlying O.S.
- While all the Applications are loaded, Host O.S. provides them with a Runtime environment.
- But virtualization layer replaces a part of this Runtime environment and gives a Virtual Environment to the Virtualized applications.

2.9 Virtualization Structure

Virtualization is achieved through the software known as virtual machine monitor or the hypervisor. So, the virtualization structure could be understood by gaining complete knowledge of Hypervisor structure and their types.

Hypervisor:

- Hypervisor is a form of virtualization software used in Cloud hosting to divide and allocate the resources on various pieces of hardware.
- The program which provides partitioning, isolation or abstraction is called virtualization hypervisor.
- Hypervisor is a hardware virtualization technique that allows multiple guest operating systems (OS) to run on a single host system at the same time. A hypervisor is sometimes also called a virtual machine manager (VMM).

Types of Hypervisor:

Two types of hypervisor categories are available.

a) **Bare-metal virtualization hypervisors: (TYPE I HYPERVISOR)**
b) **Hosted virtualization hypervisors: (TYPE II HYPERVISOR)**

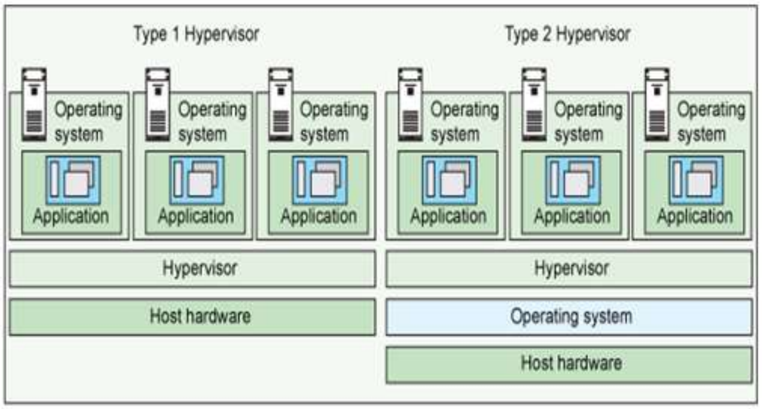


Fig. 2.13 Types of Hypervisor

a) Bare-metal virtualization hypervisors: (TYPE I HYPERVISOR)

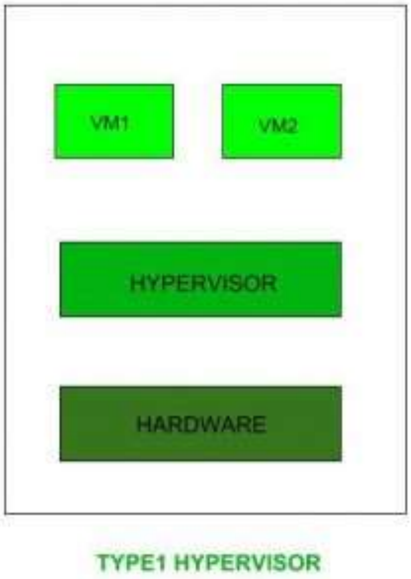


Fig 2.14 Bare-metal virtualization hypervisors

- It is deployed as a bare-metal installation (the first thing to be installed on a server as the operating system will be the hypervisor).
- The hypervisor will communicate directly with the underlying physical server hardware, manages all hardware resources and support execution of VMs.
- Hardware support is typically more limited, because the hypervisor usually has limited device drivers built into it.
- Well suited for enterprise data centers, because it usually comes with advanced features for resource management, high availability and security.
- Bare-metal virtualization hypervisors examples: VMware ESX and ESXi, Microsoft Hyper-V, Citrix Systems XenServer.
- The below figure shows stucture of TYPE I and TYPE II virtualization.

b) Hosted Virtualization Hypervisors: (TYPE II HYPERVISOR)

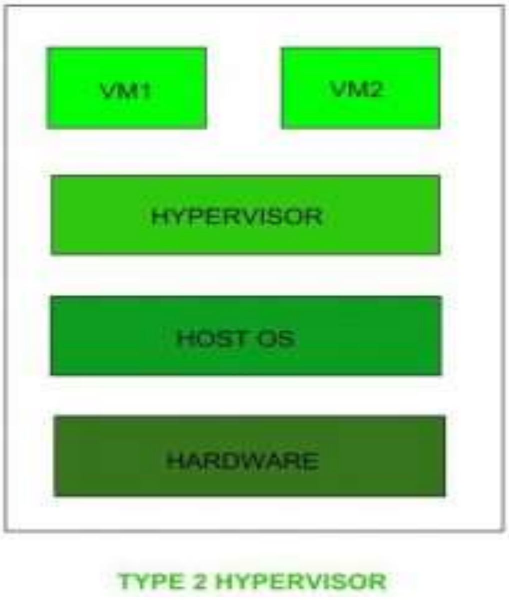


Fig 2.15 Hosted Virtualization Hypervisors

- The software is not installed onto the bare-metal, but instead is loaded on top of an already live operating system, so it requires you to first install an OS(Host OS).
- The Host OS integrates a hypervisor that is responsible for providing the virtual machines(VMs) with their virtual platform interface and for managing all context switching scheduling, etc.
- The hypervisor will invoke drivers or other component of the Host OS as needed.
- On the Host OS you may run Guest VMs, but you can also run native applications
- This approach provides better hardware compatibility than bare-metal virtualization, because the OS is responsible for the hardware drivers instead of the hypervisor.
- A hosted virtualization hypervisor does not have direct access to hardware and must go through the OS, which increases resource overhead and can degrade virtual machine (VM) performance.
- The latency is minimal and with today's modern software enhancements, the hypervisor can still perform optimally.
- Common for desktops, because they allow you to run multiple OSes. These virtualization hypervisor types are also popular for developers, to maintain application compatibility on modern OSes.
- Because there are typically many services and applications running on the host OS, the hypervisor often steals resources from the VMs running on it
- The most popular hosted virtualization hypervisors are: VMware Workstation, Server, Player and Fusion; Oracle VM VirtualBox; Microsoft Virtual PC; Parallels Desktop.

Advantages

- The latency is minimal and with today's modern software enhancements, the hypervisor can still perform optimally.
- This approach provides better hardware compatibility than bare-metal virtualization, because the OS is responsible for the hardware drivers instead of the hypervisor.

- Examples: VMware Workstation, Server, Player and Fusion; Oracle VM VirtualBox; Microsoft Virtual PC; Parallels Desktop.

2.9.1 Hypervisor Reference Model:

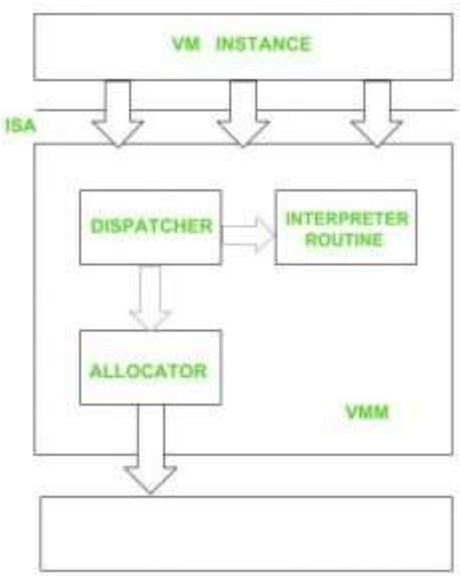


Fig 2.16 Hypervisor Reference Model

There are 3 main modules coordinate in order to emulate the underlying hardware:

1. Dispatcher
2. Allocator
3. Interpreter

1. Dispatcher:

The dispatcher behaves like the entry point of the monitor and reroutes the instructions of the virtual machine instance to one of the other two modules.

2.Allocator:

The allocator is responsible for deciding the system resources to be provided to the virtual machine instance. It means whenever virtual machine tries to execute an instruction that results in changing the machine resources associated with the virtual machine, the allocator is invoked by the dispatcher.

3.Interpreter:

The interpreter module consists of interpreter routines. These are executed, whenever virtual machine executes a privileged instruction.

2.9.2 Choosing the right hypervisor:

1. Understand your needs:

The company and its applications are the reason for the data center (and your job). Besides your company's needs, you (and your co-workers in IT) also have your own needs. Needs for a virtualization hypervisor are:

a. Flexibility
b. Scalability
c. Usability
d. Availability
e. Reliability
f. Efficiency
g. Reliable support

2. The cost of a hypervisor:

For many buyers, the toughest part of choosing a hypervisor is striking the right balance between cost and functionality. While a number of entry-level solutions are free, or practically free, the prices at the opposite end of the market can be staggering. Licensing frameworks also vary, so it's important to be aware of exactly what you're getting for your money.

3. Virtual machine performance:

Virtual systems should meet or exceed the performance of their physical counterparts, at least in relation to the applications within each server. Everything beyond meeting this benchmark is profit.

4. Ecosystem:

It's tempting to overlook the role of a hypervisor's ecosystem – that is, the availability of documentation, support, training, third-party developers and consultancies, and so on – in determining whether or not a solution is cost-effective in the long term.

5. Test for yourself:

You can gain basic experience from your existing desktop or laptop. You can run both VMware vSphere and Microsoft Hyper-V in either VMware Workstation or VMware Fusion to create a nice virtual learning and testing environment.

2.10 Tools and Mechanism:

The various tools can be categorized as follows

1. Xen Architecture
2. Binary Translation of Guest OS Requests Using a VMM.
3. KVM (Kernel-Based VM)

1. Xen Architecture:

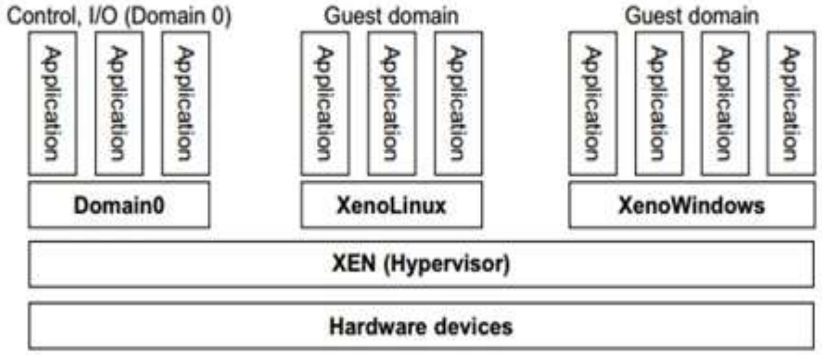


Fig 2.17 Xen Architecture

- Xen is an open-source hypervisor program developed by Cambridge University. Xen is a micro-kernel hypervisor, which separates the policy from the mechanism. The Xen hypervisor implements all the mechanisms, leaving the policy to be handled by Domain 0.
- It just provides a mechanism by which a guest OS can have direct access to the physical devices. As a result, the size of the Xen hypervisor is kept rather small. Xen provides a virtual environment located between the hardware and the OS.
- A number of vendors are in the process of developing commercial Xen hypervisors, among them are Citrix XenServer and Oracle VM .

The core components of a Xen system are the hypervisor, kernel, and applications. Like other virtualization systems, many guest OSes can run on top of the hypervisor. However, not all guest OSes are created equal, and one in particular control the others. The guest OS, which has control ability, is called Domain 0, and the others are called Domain U. Domain 0 is a privileged guest OS of Xen. It is first loaded when Xen boots without any file system drivers being available. Domain 0 is designed to access hardware directly and manage devices.

Therefore, one of the responsibilities of Domain 0 is to allocate and map hardware resources for the guest domains (the Domain U domains).

2. Binary Translation of Guest OS Requests Using a VMM:

This approach was implemented by VMware and many other software companies. As shown in Figure 3.6, VMware puts the VMM at Ring 0 and the guest OS at Ring 1. The VMM scans the instruction stream and identifies the privileged, control- and behaviour-sensitive instructions. When these instructions are identified, they are trapped into the VMM, which emulates the behaviour of these instructions. The method used in this emulation is called binary translation. Therefore, full virtualization combines binary translation and direct execution. The guest OS is completely decoupled from the underlying hardware. Consequently, the guest OS is unaware that it is being virtualized.

The performance of full virtualization may not be ideal, because it involves binary translation which is rather time-consuming. In particular, the full virtualization of I/O-intensive applications is a really a big challenge. Binary translation employs a code cache to store translated hot instructions to improve performance, but it increases the cost of memory usage. At the time of this writing, the performance of full virtualization on the x86 architecture is typically 80 percent to 97 percent that of the host machine.

3. KVM (Kernel-Based VM):

This is a Linux para-virtualization system—a part of the Linux version 2.6.20 kernel. Memory management and scheduling activities are carried out by the existing Linux kernel. The KVM does the rest, which makes it simpler than the hypervisor that controls the entire machine. KVM is a hardware-assisted para-virtualization tool, which improves performance and supports unmodified guest OSes such as Windows, Linux, Solaris, and other UNIX variants.

2.11 Virtualization of CPU:

- CPU virtualization involves a single CPU acting as if it were multiple separate CPUs. This allows an operating system to more effectively & efficiently utilize the CPU power in the computer so that it runs faster.
- CPU Virtualization emphasizes on running programs and instructions through virtual machine giving the feeling as it is working on a physical workstation. All the operations are handled by an emulator that controls software to run according to it.

Types of CPU Virtualization:

1. **Software-Based CPU Virtualization:**

- With software-based CPU virtualization, the guest application code runs directly on the processor, while the guest privileged code is translated, and the translated code runs on the processor.

- The translated code is slightly larger and usually runs more slowly than the native version. As a result, guest applications, which have a small, privileged code component, run with speeds very close to native. Applications with a significant privileged code component, such as system calls, traps, or page table updates can run slower in the virtualized environment.

2. **Hardware-Assisted CPU Virtualization:**

- Certain processors provide hardware assistance for CPU virtualization.

- When using this assistance, the guest can use a separate mode of execution called guest mode. The guest code, whether application code or privileged code, runs in the guest mode. On certain events, the processor exits out of guest mode and enters root mode. The hypervisor executes in the root mode, determines the reason for the exit, takes any required actions, and restarts the guest in guest mode.

- When you use hardware assistance for virtualization, there is no need to translate the code. As a result, system calls or trap-intensive workloads run very close to native speed. Some workloads, such as those involving updates to page tables, lead to a large number of exits from guest mode to root mode. Depending on the number of such exits and total time spent in exits, hardware-assisted CPU virtualization can speed up execution significantly.

2.11.1 Why CPU Virtualization is Important?

- Using CPU Virtualization, the overall performance and efficiency are improved to a great extent. This saves cost and money.
- As CPU Virtualization uses virtual machines to work on separate operating systems on a single sharing system, security is also maintained by it.
- It provides the best backup of computing resources since the data is stored and shared from a single system.
- It provides reliability to users that are dependent on a single system and also provides greater retrieval options of data for the user.
- It also offers great and fast deployment procedure options so that it reaches the client without any hassle.
- Virtualization ensures the desired data to reach the desired clients through the medium and checks any constraints are there and are also fast to remove it.

2.12 Virtualization of Memory:

- Physical memory across different servers is aggregated into a single virtualized memory pool and provided to different virtual machines as per users requirement.
- It enhances performance by providing greater memory capacity without any addition to the main memory.

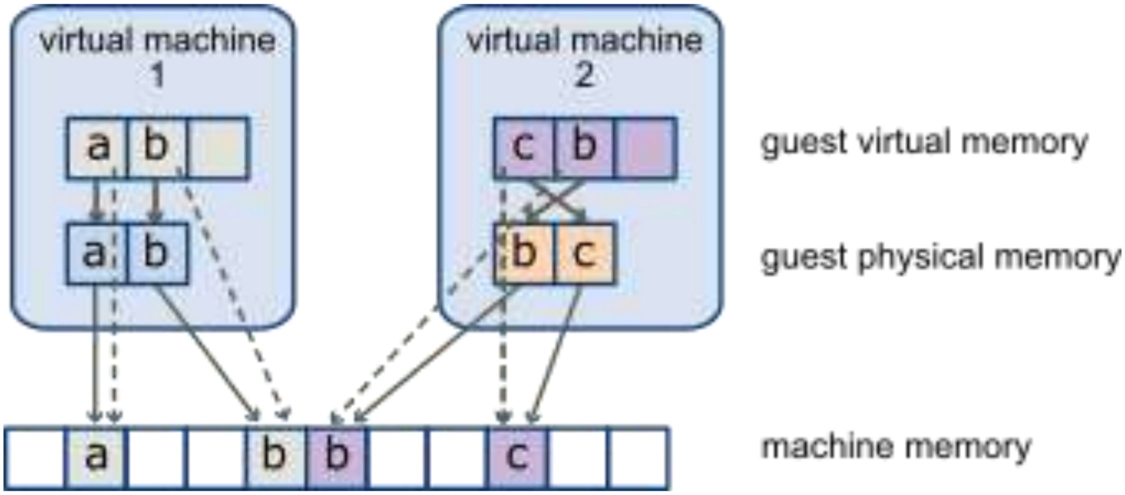


Fig 2.18 Virtualization of Memory

Basically, it consists of following two concepts.

1. **Mapping:**

A two-stage mapping process should be maintained by the guest OS and the VMM, respectively:

- virtual memory to physical memory.
- physical memory to machine memory.

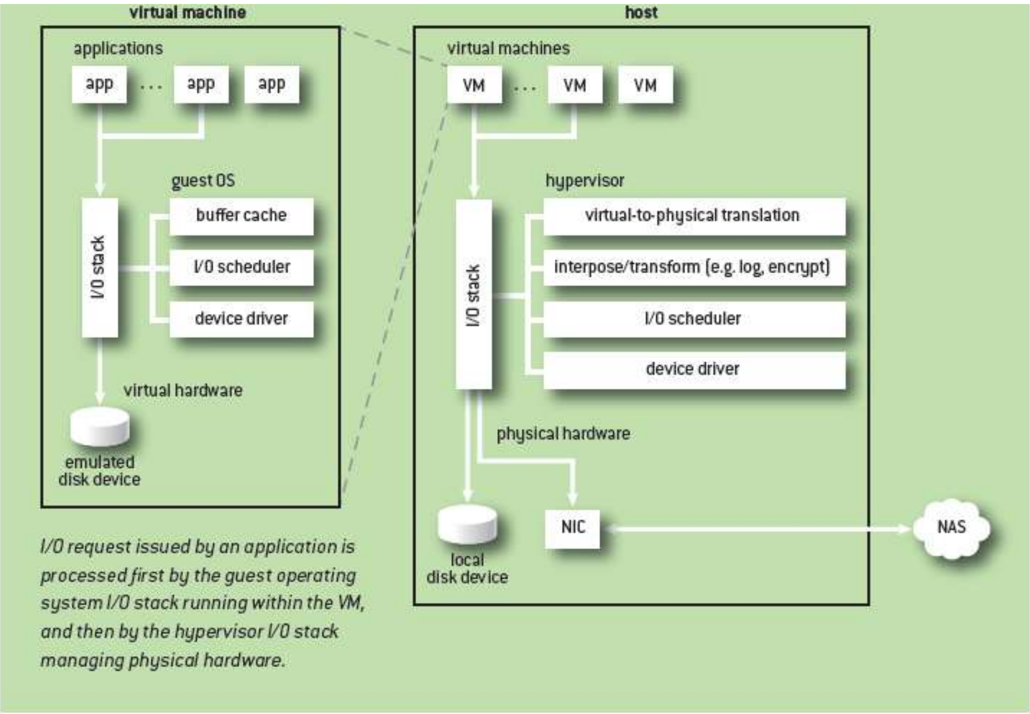
Furthermore, MMU (Memory Management Unit) virtualization should be supported, which is transparent to the guest OS. The guest OS continues to control the mapping of virtual addresses to the physical memory addresses of VMs. But the guest OS cannot directly access the actual machine memory. The VMM is responsible for mapping the guest physical memory to the actual machine memory.

2. Shadow Page Table:

- Shadow page tables are used by the hypervisor to keep track of the state in which the guest "thinks" its page tables should be.

- The guest can't be allowed access to the hardware page tables because then it would essentially have control of the machine. So, the hypervisor keeps the "real" mappings (guest virtual -> host physical) in the hardware when the relevant guest is executing and keeps a representation of the page tables that the guest thinks it's using "in the shadows."

2.13 Virtualization of I/O Devices:



I/O request issued by an application is processed first by the guest operating system I/O stack running within the VM, and then by the hypervisor I/O stack managing physical hardware.

Fig 2.19 Virtualization of I/O devices

- Figure depicts the flow of an I/O request in a virtualized system. When an application running within a VM issues an I/O request, typically by making a system call, it is initially processed by the I/O stack in the guest operating system, which is also running within the

VM. A device driver in the guest issues the request to a virtual I/O device, which the hypervisor then intercepts. The hypervisor schedules requests from multiple VMs onto an underlying physical I/O device, usually via another device driver managed by the hypervisor or a privileged VM with direct access to physical hardware.

- When a physical device finishes processing an I/O request, the two I/O stacks must be traversed again, but in the reverse order. The actual device posts a physical completion interrupt, which is handled by the hypervisor. The hypervisor determines which VM is associated with the completion and notifies it by posting a virtual interrupt for the virtual device managed by the guest operating system. To reduce overhead, some hypervisors perform virtual interrupt coalescing in software, similar to the hardware batching optimizations found in physical cards, which delay interrupt delivery with the goal of posting only a single interrupt for multiple incoming events.

2.14 Virtualization Support and Disaster Recovery:

- When it comes to backup and **disaster recovery**, **virtualization** changes everything by consolidating the entire server environment, along with all the workstations and other systems into a single virtual machine.

- With virtualization, your "hardware" is all virtual and completely separated from the actual, physical hardware in the host server. This separation means it's much easier to take a virtual machine (VM) and restore it to different hardware, as long as both servers are running the same virtual platform (hypervisor) and have comparable resources.

- Another factor with virtualization is that you simply have fewer machines to worry about. If you go from 50 physical machines to 10 virtual servers, you've reduced the number of restorations you have to do by 80 percent.

2.14.1 Elements of an effective disaster recovery plan:

1. **Disaster recovery team:** This assigned group of specialists will be responsible for creating, implementing and managing the disaster recovery plan. This plan should define each team member's role and responsibilities. In the event of a disaster, the recovery team should know how to communicate with each other, employees, vendors, and customers.

2. **Risk evaluation:** Assess potential hazards that put your organization at risk. Depending on the type of event, strategize what measures and resources will be needed to resume business. For example, in the event of a cyber attack, what data protection measures will the recovery team have in place to respond?

3. **Business-critical asset identification:** A good disaster recovery plan includes documentation of which systems, applications, data, and other resources are most critical for business continuity, as well as the necessary steps to recover data.

4. **Backups:** Determine what needs backup (or to be relocated), who should perform backups, and how backups will be implemented. Include a recovery point objective (RPO) that states the frequency of backups and a recovery time objective (RTO) that defines the maximum amount of downtime allowable after a disaster. These metrics create limits to guide the choice of IT strategy, processes and procedures that make up an organization's disaster recovery plan. The amount of downtime an organization can handle and how frequently the organization backs up its data will inform the disaster recovery strategy.

5. **Testing and optimization:** The recovery team should continually test and update its strategy to address ever-evolving threats and business needs. By continually ensuring that a company is ready to face the worst-case scenarios in disaster situations, it can successfully navigate such challenges. In planning how to respond to a cyber attack, for example, it's important that organizations continually test and optimize their security and

data protection strategies and have protective measures in place to detect potential security breaches.

2.14.2 How Virtualization helps with Disaster Recovery

- **Recover to any hardware**

  By using a virtualized environment, you don't have to worry about having completely redundant hardware. Instead, you can use almost any x86 platform as a backup solution, this allows you to save money by repurposing existing hardware and also gives your company more agility when it comes to hardware failure as almost any virtual server can be restarted on different hardware.

- **Backup and restore full images**

  By having your system completely virtualized each of your server's files are encapsulated in a single image file. An image is basically a single file that contains all of server's files, including system files, programs, and data; all in one location. By having these images, it makes managing your systems easy and backups become as simple as duplicating the image file and restores are simplified to simply mounting the image on a new server.

- **Run other workloads on standby hardware**

  A key benefit to virtualization is reducing the hardware needed by utilizing your existing hardware more efficiently. This frees up systems that can now be used to run other tasks or be used as a hardware redundancy. This mixed with features like VMware's High Availability, which restarts a virtual machine on a different server when the original hardware fails, or for a more robust disaster recovery plan you can use Fault Tolerance, which keeps both servers in sync with each other leading to zero downtime if a server should fail.

- **Easily copy system data to recovery site**

Having an offsite backup is a huge advantage if something were to happen to your specific location, whether it be a natural disaster, a power outage, or a water pipe bursting, it is nice to have all your information at an offsite location. Virtualization makes this easy by easily copying each virtual machine image to the offsite location and with the easy customizable automation process, it doesn’t add any more strain or man hours to the IT department.

Important Questions:

1. What do you mean by full virtualization?
2. What is a Hypervisor? Explain in detail with necessary illustrations.
3. What is the difference between process virtual machines, host VMMs and native VMMs?
4. Explain the characteristics and type of virtualization in Cloud Computing.
5. What is the importance of a virtual machine? What role do they play in cloud computing?
6. What do you understand by service-oriented architecture (SOA)? How does it support cloud computing?
7. Define Web Services with example. Describe how REST services are useful in creating web services.
8. Describe implementation level of virtualization in detail with examples.
9. How virtualization of I/O Devices and memory plays an important role in Cloud-Computing.
10. Why CPU Virtualization is Important?
11. Explain how Virtualization Support and Disaster Recovery.

UNIT-3

CLOUD ARCHITECTURE, SERVICES AND STORAGE

3.1 Layered Cloud Architecture Design:

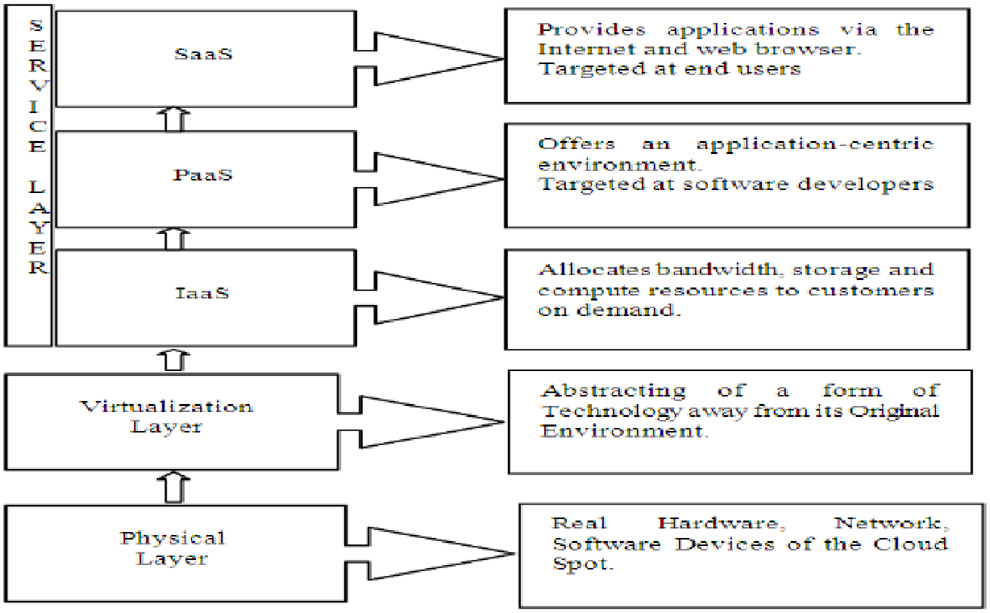

SERVICE LAYER
SaaS
Provides applications via the Internet and web browser. Targeted at end users
PaaS
Offers an application-centric environment. Targeted at software developers
IaaS
Allocates bandwidth, storage and compute resources to customers on demand.
Virtualization Layer
Abstracting of a form of Technology away from its Original Environment.
Physical Layer
Real Hardware, Network, Software Devices of the Cloud Spot.

Fig 3.1 Layered Cloud Architecture Design

1. **Physical Layer:**
   - The physical layer comprises physical compute, storage, and network resources
   - Compute systems execute software of providers and consumers
   - Storage systems store business and application data
   - Networks connect compute systems with each other and with storage systems – Networks also connect multiple data centers or multiple clouds to one another

2. Virtualization Layer:

- Virtualization is a good way to use and distribute system resources efficiently. The virtualization Layer is an additional abstraction layer between physical and service layer, hardware, computing, and the application running on it.

- A machine with a virtualization layer can create other (virtual) machines, where you can install additional operating systems. In this case, you are independent of hardware for the operating system and the applications.
- It can be used for creating virtual machines of all the physical resource that are available at physical layer and can be used for sharing the resources virtually to the users. The sharing of resources will lead to increased efficiencies and decreased (hardware-)costs.
- VMware and Microsoft Hyper-V are known virtualization tools, but there a plenty more of them like xen, which is the basis of Amazons Cloud Service Amazon EC2, and VM Virtual Box for host/guest-architecture.

3. Service Layer:

The service layer consists of *service models* of cloud-computing which is used for providing services to cloud users.

The service models are categorized into three basic models:

1. Software-as-a-Service (SaaS):

- SaaS is known as **'On-Demand Software'.**
- It is a software distribution model. In this model, the applications are hosted by a cloud service provider and publicized to the customers over internet.
- In SaaS, associated data and software are hosted centrally on the cloud server.
- User can access SaaS by using a thin client through a web browser.

- CRM, Office Suite, Email, games, etc. are the software applications which are provided as a service through Internet.
- The companies like Google, Microsoft provide their applications as a service to the end users.

2. Platform-as-a-Service (PaaS):

- PaaS is a programming platform for developers. This platform is generated for the programmers to create, test, run and manage the applications.
- A developer can easily write the application and deploy it directly into PaaS layer.
- PaaS gives the runtime environment for application development and deployment tools.
- Google Apps Engine (GAE), Windows Azure, SalesForce.com are the examples of PaaS.

3. Infrastructure-as-a-Service (IaaS):

- IaaS is a way to deliver a cloud computing infrastructure like server, storage, network, and operating system.
- The customers can access these resources over cloud computing platform i.e Internet as an on-demand service.
- In IaaS, you buy complete resources rather than purchasing server, software, data center space or network equipment.
- IaaS was earlier called as Hardware as a Service (HaaS). It is a Cloud computing platform-based model.
- HaaS differs from IaaS in the way that users have the bare hardware on which they can deploy their own infrastructure using most appropriate software.

3.2 NIST *(National Institute of Standards and Technology)* Cloud Computing Architecture:

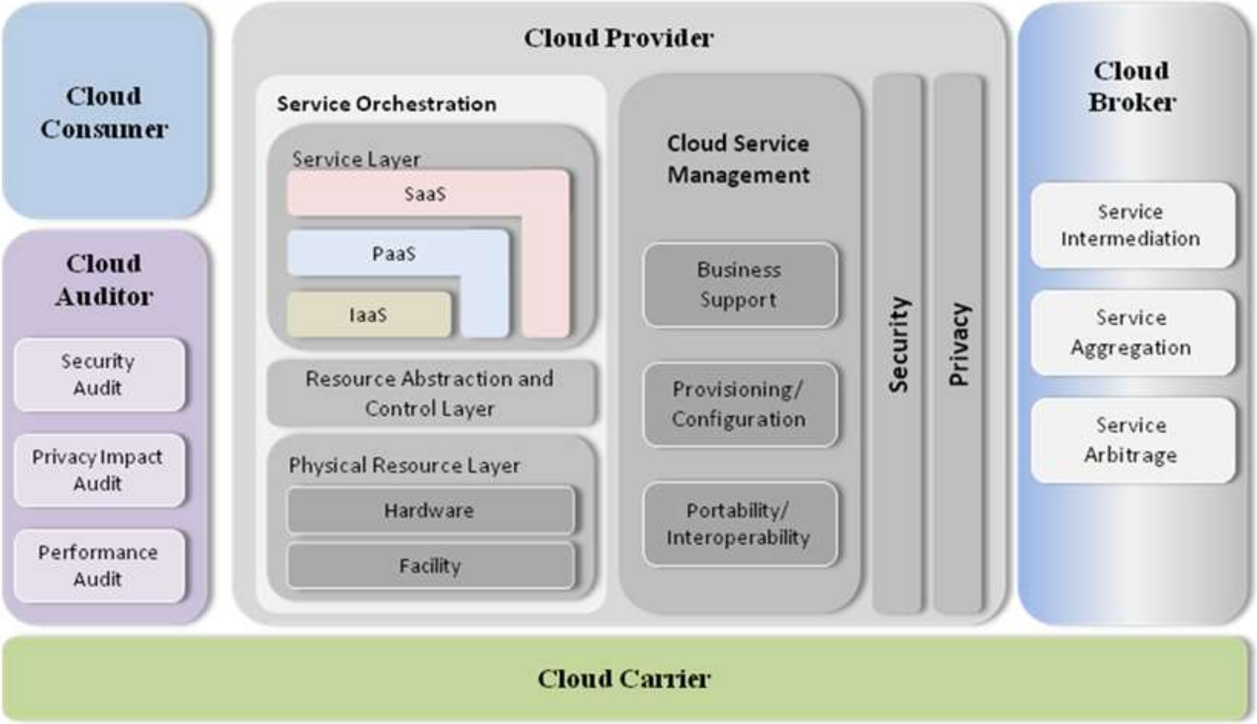


Fig 3.2 NIST Cloud Computing Architecture:

As shown in Figure 3.2, the NIST cloud computing reference architecture defines five major actors: cloud consumer, cloud provider, cloud carrier, cloud auditor and cloud broker. Each actor is an entity (a person or an organization) that participates in a transaction or process and/or performs tasks in cloud computing.

3.2.1 Actors in Cloud Computing

3.2.1.1 Cloud Consumer

3.2.1.2 Cloud Provider

3.2.1.3 Cloud Auditor

3.2.1.4 Cloud Broker

3.2.1.5 Cloud Carrier

3.2.1.1 Cloud Consumer

- A person or organization that maintains a business relationship with, and uses service from, Cloud Providers.
- The cloud consumer is the principal stakeholder for the cloud computing service. A cloud consumer represents a person or organization that maintains a business relationship with, and uses the service from a cloud provider. A cloud consumer browses the service catalog from a cloud provider, requests the appropriate service, sets up service contracts with the cloud provider, and uses the service.

3.2.1.2 Cloud consumers of SaaS:

SaaS applications in the cloud and made accessible via a network to the SaaS consumers.

3.2.1.3 Who are Cloud consumer of SaaS?

The consumers of SaaS can be an organization that provide their members with access to software applications, end users who directly use software applications, or software application administrators who configure applications for end users. SaaS consumers can be billed based on the number of end users, the time of use, the network bandwidth consumed, the amount of data stored or duration of stored data.

3.2.1.4 Cloud consumers of PaaS: can employ the tools and execution resources provided by cloud providers to develop, test, deploy and manage the applications hosted in a cloud environment.

3.2.1.5 Who are Cloud consumer of PaaS?

PaaS consumers can be application developers who design and implement application software, application testers who run and test applications in cloud-based environments, application deployers who publish applications into the cloud, and application administrators who configure and monitor application performance on a platform.

PaaS consumers can be billed according to, processing, database storage and network resources consumed by the PaaS application, and the duration of the platform usage.

3.2.1.6 Cloud consumers of IaaS:

Consumers of IaaS have access to virtual computers, network-accessible storage, network infrastructure components, and other fundamental computing resources on which they can deploy and run arbitrary software.

3.2.1.7 Who are Cloud consumer of IaaS?

The consumers of IaaS can be system developers, system administrators and IT managers who are interested in creating, installing, managing and monitoring services for IT infrastructure operations.

IaaS consumers are provisioned with the capabilities to access these computing resources, and are billed according to the amount or duration of the resources consumed, such as CPU hours used by virtual computers, volume and duration of data stored, network bandwidth consumed, number of IP addresses used for certain intervals.

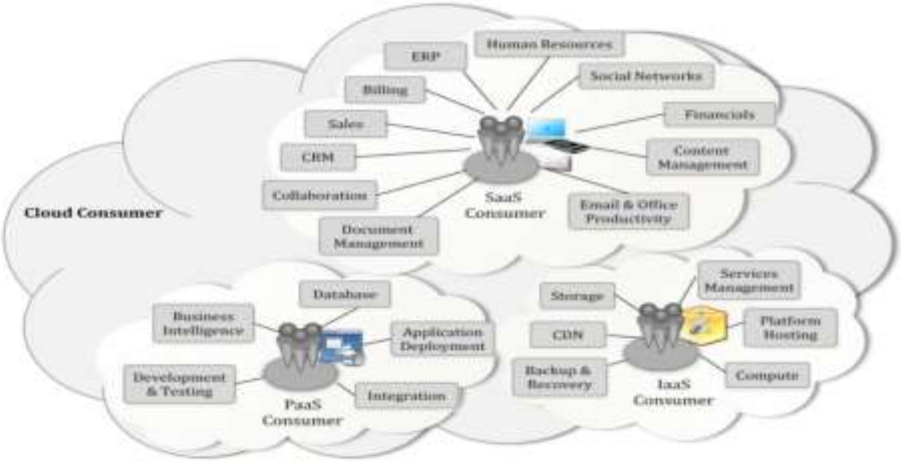


Fig 3.3 Cloud consumers of SaaS, PaaS, IaaS

3.2.2 Cloud Providers

- A person, organization, or entity responsible for making a service available to interested parties.
- A Cloud Provider acquires and manages the computing infrastructure required for providing the services, runs the cloud software that provides the services, and makes arrangement to deliver the cloud services to the Cloud Consumers through network access.

3.2.2.1 What Cloud Providers does for SaaS?

For Software as a Service, the cloud provider deploys, configures, maintains and updates the operation of the software applications on a cloud infrastructure so that the services are provisioned at the expected service levels to cloud consumers. The provider of SaaS assumes most of the responsibilities in managing and controlling the applications and the infrastructure, while the cloud consumers have limited administrative control of the applications.

3.2.2.2 What Cloud Providers does for PaaS?

- For PaaS, the Cloud Provider manages the computing infrastructure for the platform and runs the cloud software that provides the components of the platform, such as runtime software execution stack, databases, and other middleware components.
- The PaaS Cloud Provider typically also supports the development
- For deployment and management process of the PaaS Cloud Consumer by providing tools such as integrated development environments (IDEs), development version of cloud software, software development kits (SDKs), deployment and management tools.
- The PaaS Cloud Consumer has control over the applications and possibly some of the hosting environment settings but has no or limited access to the infrastructure

underlying the platform such as network, servers, operating systems (OS), or storage.

3.2.2.3 What Cloud Providers does for IaaS?

For IaaS –

- The Cloud Provider acquires the physical computing resources underlying the service, including the servers, networks, storage and hosting infrastructure.
- The Cloud Provider runs the cloud software necessary to makes computing resources available to the IaaS Cloud Consumer through a set of service interfaces and computing resource abstractions, such as virtual machines and virtual network interfaces.
- The IaaS Cloud Consumer in turn uses these computing resources, such as a virtual computer, for their fundamental computing needs Compared to SaaS and PaaS Cloud Consumers, an IaaS Cloud Consumer has access to more fundamental forms of computing resources and thus has more control over the more software components in an application stack, including the OS and network.
- The IaaS Cloud Provider, on the other hand, has control over the physical hardware and cloud software that makes the provisioning of these infrastructure services possible, for example, the physical servers, network equipments, storage devices, host OS and hypervisors for virtualization.

3.2.2.4 Activities performed by Cloud Provider:

A Cloud Provider's activities can be described in five major areas, as shown in below Figure, a cloud provider conducts its activities in the areas of service deployment, service orchestration, cloud service management, security, and privacy. We will see detail in respective sections.

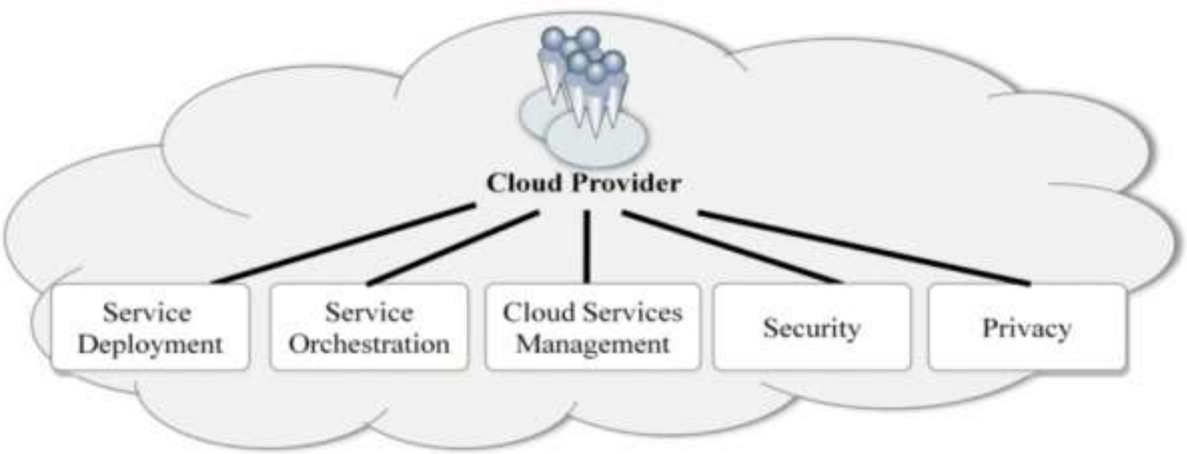


Figure 3.4 – Cloud Provider – Major Activities

3.2.2.4 Service Deployment:

A **service deployment** is a configuration of a collection of SAS Foundation Services that specifies the data necessary to instantiate the services, as well as dependencies upon other services. You create **service deployments** for applications that will **deploy** or access the services.

There are four ***cloud deployment*** models: public, private, community, and hybrid.

3.2.2.5 Service Orchestration:

A cloud orchestrator automates the management, coordination and organization of complicated computer systems, services, and middleware. In addition to reduced personnel involvement, orchestration eliminates the potential for errors introduced into provisioning, scaling or other cloud processes.

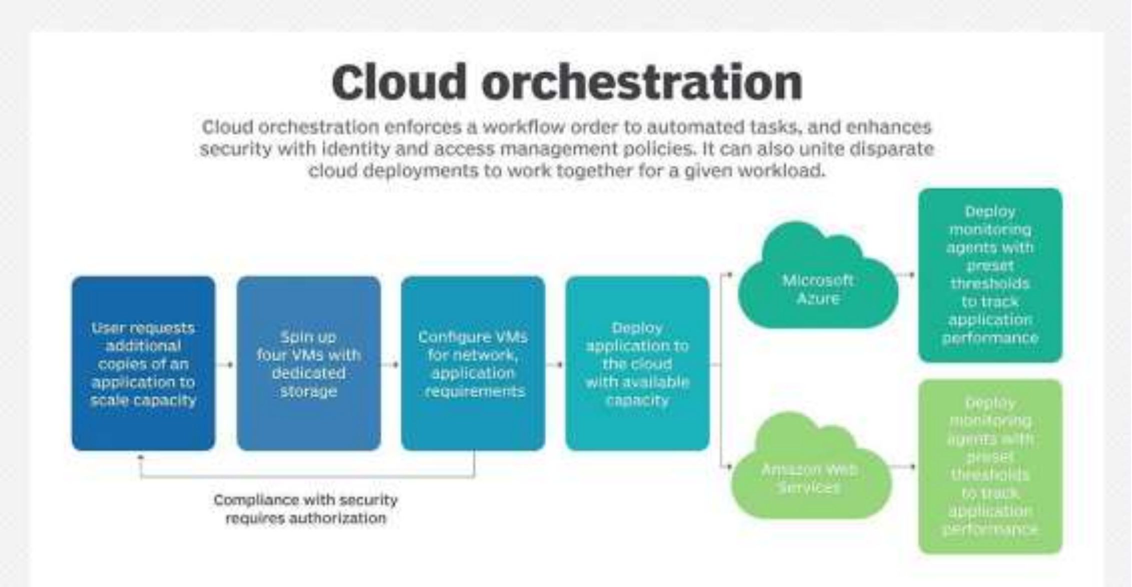

Cloud orchestration
Cloud orchestration enforces a workflow order to automated tasks, and enhances security with identity and access management policies. It can also unite disparate cloud deployments to work together for a given workload.
User requests additional copies of an application to scale capacity
Spin up four VMs with dedicated storage
Configure VMs for network, application requirements
Deploy application to the cloud with available capacity
Microsoft Azure
Deploy monitoring agents with preset thresholds to track application performance
Amazon Web Services
Deploy monitoring agents with preset thresholds to track application performance
Compliance with security requires authorization

Fig 3.5 Cloud Orchestration

3.2.2.6 Cloud Service Management:

Cloud Service Management and Operations entails all the activities that an organization does to plan, design, deliver, operate, and control the IT and **cloud services** that it offers to customers. **Service management** includes the operational aspects of your applications and **services**. For this it consists of following activities.

- **Business Support:**

  Cloud computing provides businesses with simple IT management and maintenance support. Besides, once you move to cloud, the IT infrastructure will also be updated automatically without any extra expense. Hence, not much maintenance is required as everything is managed by the service provider.

- **Provisioning and Configuration:**

  Provisioning is the process of setting up IT infrastructure. It can also refer to the steps required to manage access to data and resources and make them available to users and systems. **Provisioning** is not the same thing as **configuration**, but they are both steps in the deployment process.

  Cloud configuration is the process of setting hardware and software details for elements of a **cloud** environment to ensure that they can interoperate and communicate.

- **Portability and Interoperability:**

  Cloud portability is the ability to transfer applications between **cloud** environments without messing with the app itself. Because a **cloud** provider's environment is inherently different from others, moving apps between different providers can cause compatibility issues.

  Cloud interoperability is the ability of a customer's system to interact with a **cloud** service or the ability for one **cloud** service to interact with other **cloud** services by exchanging information according to a prescribed method to obtain predictable results.

3.2.2.7 Security:

Security management is the identification of an organization's assets (including people, buildings, machines, systems and information assets), followed by the development, documentation, and implementation of policies and procedures for protecting assets.

3.2.2.8 Privacy:

If we discuss in terms of data, Data Privacy Management enables organizations to assess and continuously measure data privacy compliance with multifactor risk scoring and monitoring of data access and movement. Orchestrate risk remediation by automating data protection and cost-effectively report on data subject requests with transparency.

3.2.3 Cloud Auditor:

- A party that can conduct independent assessment of cloud services, information system operations, performance and security of the cloud implementation.
- Audits are performed to verify conformance to standards through review of objective evidence. A cloud auditor can evaluate the services provided by a cloud provider in terms of security controls, privacy impact, performance, etc.

- Auditing is especially important for federal agencies as “agencies should include a contractual clause enabling third parties to assess security controls of cloud providers”.

Auditing in Cloud Computing can be categorized as follows:

3.2.3.1 Security Audit:

Security audit is being performed for assessing the security issues within the Cloud Computing process. The process to audit cloud vendors should be straightforward and performed by taking an inventory of data. It must determine the most important data to secure using a simple three or four level classification: public, internal, confidential, and restricted.

3.2.3.2 Privacy Impact Audit:

A privacy impact audit can help Federal agencies comply with applicable privacy laws and regulations governing an individual's privacy, and to ensure confidentiality, integrity, and availability of an individual's personal information at every stage of development and operation.

3.2.3.3 Performance Audit:

Performance audit refers to an independent examination of a program, function, operation or the management systems and procedures of a governmental or non-profit entity to assess whether the entity is achieving economy, efficiency, and effectiveness in the employment of available resources.

3.2.4 Cloud Broker:

An entity that manages the use, performance and delivery of cloud services, and negotiates relationships between Cloud Providers and Cloud Consumers.

In general, a cloud broker can provide services in three categories:

Service Intermediation: A cloud broker enhances a given service by improving some specific capability and providing value-added services to cloud consumers. The improvement can be managing access to cloud services, identity management, performance reporting, enhanced security, etc.

Service Aggregation: A cloud broker combines and integrates multiple services into one or more new services. The broker provides data integration and ensures the secure data movement between the cloud consumer and multiple cloud providers.

Service Arbitrage: Service arbitrage is like service aggregation except that the services being aggregated are not fixed. Service arbitrage means a broker has the flexibility to choose services from multiple agencies. The cloud broker, for example, can use a credit-scoring service to measure and select an agency with the best score.

3.2.4.1 Why we need Cloud Broker?

As cloud computing evolves, the integration of cloud services can be too complex for cloud consumers to manage. A cloud consumer may request cloud services from a cloud broker, instead of contacting a cloud provider directly.

3.2.5 Cloud Carrier:

An intermediary that provides connectivity and transport of cloud services from Cloud Providers to Cloud Consumers.

3.2.5.1 Who is Cloud Carrier?

- A cloud carrier acts as an intermediary that provides connectivity and transport of cloud services between cloud consumers and cloud providers.
- Cloud carriers provide access to consumers through network, telecommunication, and other access devices. For example, cloud consumers can obtain cloud services through network access devices, such as computers, laptops, mobile phones, mobile Internet devices (MIDs), etc.

- The distribution of cloud services is normally provided by network and telecommunication carriers or a transport agent, where a transport agent refers to a business organization that provides physical transport of storage media such as high-capacity hard drives.

- Note that a cloud provider will set up SLAs with a cloud carrier to provide services consistent with the level of SLAs offered to cloud consumers and may require the cloud carrier to provide dedicated and secure connections between cloud consumers and cloud providers.

3.3 Public, Private and Hybrid Clouds:

- The terms Public, Private and Hybrid Clouds refer to Deployment Models of the cloud.
- Cloud deployment refers to the enablement of SaaS (software as a service), PaaS (platform as a service) or IaaS (infrastructure as a service) solutions that may be accessed on demand by end users or consumers.
- Cloud deployment model represents the exact category of cloud environment based on proprietorship, size, and access and also describes the nature and purpose of the cloud.

3.3.1 Public Cloud Model:

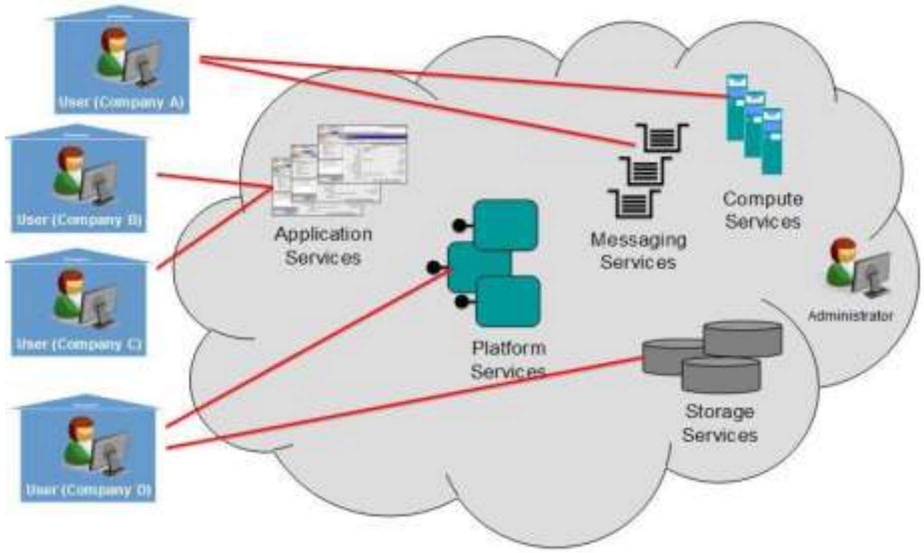


Fig 3.2 Public Cloud Model

- A public cloud is a cloud service offered to multiple customers by a cloud provider. It is a type of computing in which a service provider makes resources available to the common public via the internet. Resources vary by provider but may include storage capabilities, applications, or virtual machines.
- Public cloud operates on the **pay-as-per-use model** and administrated by the **third party**, i.e., Cloud service provider. In the Public cloud, the same storage is being used by multiple users at the same time.

Examples: Amazon Elastic Compute **Cloud** (EC2), Microsoft Azure, IBM's Blue **Cloud**, Sun **Cloud**, and Google **Cloud** are **examples** of the **public cloud**.

3.3.2 Advantages of Public Cloud Model:

High scalability:

Public cloud solutions allow you to evolve at almost infinite speed. This would not be possible in a local data center. Because the division of resources between customers is done dynamically, your business can double or even triple the amount of computing and storage to meet peak demand. All without increasing your workload or exponentially increasing the cost of the system.

Cost reduction

The costs associated with the hardware, applications and bandwidth are responsibility of the supplier. The payment of the service is usually monthly or annual and according to the usage as it follows the model, pay-as-you-go.

Disaster Recovery

A large majority of IT managers see the implementation of a disaster recovery plan as expensive, complex, and difficult to deploy. But now service providers such as Microsoft Azure provide disaster recovery for all major IT systems without the expenditure of any secondary infrastructure. Public cloud has minimal risk of losing data as most of the cloud service providers will have multiple back-up infrastructures.

Reliability and flexibility

Public hosting makes it easy to adapt to peak loads. Depending on its needs, the client can add or delete resources. This type of service reduces the complexity and implementation time required for testing and deploying new applications.

Cons of Public Cloud

Cloud computing products undergo constant updates to both servers and security features. Still, the risks of intrusions and possible service instabilities weigh against the cloud model. However, if properly implemented, they can be as secure as private cloud implementation. Be sure to use the appropriate security methods, such as Intrusion Detection and Prevention (IDPs) systems.

3.3.3 Disadvantages of Public Cloud Model:

Security

The protection and privacy of data hosted by cloud providers remain the top two concerns of any major business. Public cloud services offered by any leading vendors are secure, but the real difficulty is to use them in a secure way. Companies must indeed adopt good practices in terms of cybersecurity. There is no risk of intrusion between neighbours on a public network, but misuse of the infrastructure contracted by another company can put yours at risk as a successful attack on the main server opens a breach for each client’s system. But this kind of situation occurs rarely when the provider is reliable and of quality.

Lack of customization

The multitenancy atmosphere of the public cloud can limit or restrain any customization. This can be catastrophic for organizations that have a complex application process or complicated network architecture.

Minimal Understanding of the Back-end Process

Third-party access to confidential information creates a risk of compromising the sensitive data of the company. The cloud user is responsible for application-level security.In the public cloud scenario, we generally don’t have any idea of how our data is being handled in the back end i.e. how it is processed to give us the desired results.

3.3.4 Private Cloud Model:

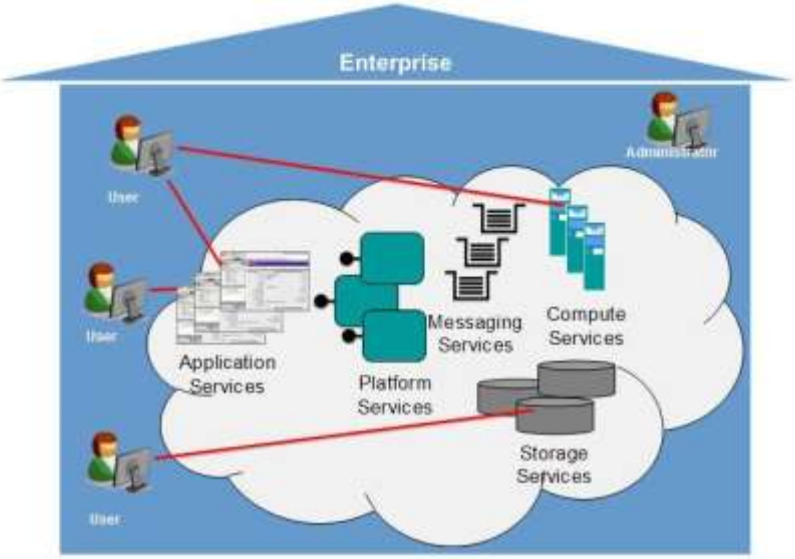


Fig 3.3 Private Cloud Model

- **Private Cloud is** also termed as '**Internal Cloud**', which allows the accessibility of systems and services within a specific boundary or organization. The cloud platform is implemented in a cloud-based secure environment guarded by advanced firewalls under the surveillance of the IT department that belongs to a particular organization.
- Private clouds permit only authorized users, providing the organizations greater control over data and its security. Business organizations that have dynamic, critical, secured, management demand-based requirements should adopt Private Cloud.

Examples: HP Data Centers, Microsoft, Elastra-**private cloud**, and Ubuntu are the **example** of a **private cloud**.

3.3.5 Advantages of Private Cloud Model:

Enhanced security and privacy: In addition to the perfectly robust security that is possible on individual virtual machines, a private cloud can be isolated from all but the company who owns it. This restricted access, which can integrate with a firm's firewall and other remote access policies, offers an additional layer of security.

Improved reliability: When compared to either dedicated hardware or public cloud alternatives, private cloud offers a greater degree of reliability thanks to a fault resilient and redundant architecture that isn't shared in any way.

Improved performance: The resources within your private cloud infrastructure are at the disposal of your company and your company alone. There is no contention with other companies for capacity (only with your own workloads) and far less chance that a malicious attack against another firm will affect your ability to function.

Increased flexibility: Unlike a physical machine, a virtual machine can be scaled up and down seamlessly. And when you own all the virtual machines, you can reallocate resources dynamically, wherever they are needed most.

Total control: Although there is a fair amount of universal best practice that you should no doubt follow, you are free to build and configure your private cloud in any way you like. For example, you have the freedom to use any operating systems and applications you please and to allocate resources in any way you see fit.

3.3.6 Disadvantages of Private Cloud Model:

Cost: With exclusivity comes increased cost. If you plan to build your own private cloud, you face a large capital outlay. Fortunately, you can rent your private cloud from a hosting service provider, for a monthly fee, and still benefit from all the advantages.

Under-utilisation: With a private cloud, the cost of capacity under-utilisation is a cost to you, not to your provider. Therefore, managing and maximising utilisation becomes your concern.

Platform scaling: Since you are unlikely to want to retain significant, un-utilised capacity, based on the previous point, large upward changes in your requirements are likely to require scaling of the physical infrastructure. This is fine but may take longer than simply scaling a virtual machine within existing capacity.

3.3.7 Hybrid Cloud Model:

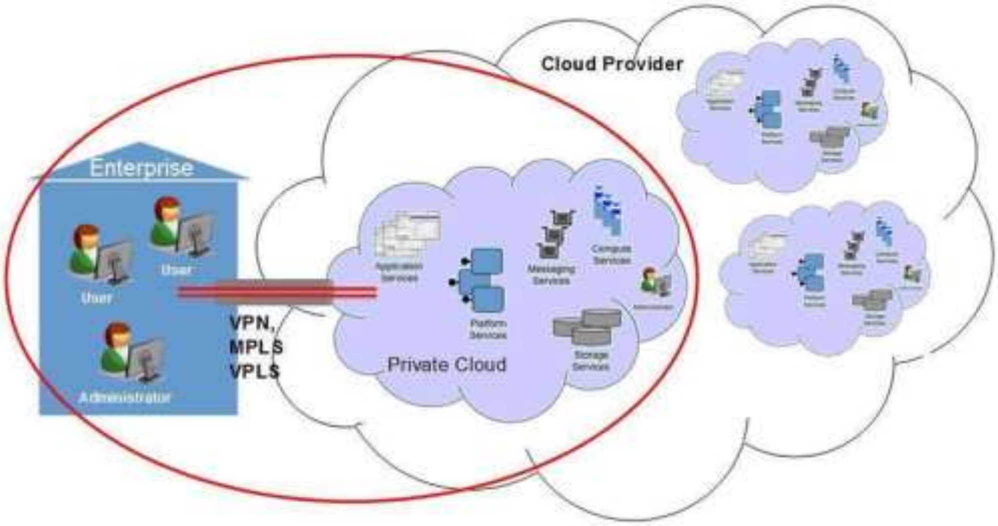


Fig 3.4 Hybrid Cloud Model

- A hybrid cloud model consists of public cloud and private cloud components. The public cloud model relies on a third-party provider for on-demand IT resources, such as virtual machines, applications, or storage, as well as services such as data analytics over the internet or a dedicated network.
- It refers to a mixed computing, storage, and services environment made up of on-premises infrastructure, private cloud services, and a public cloud.
- Example: Amazon Web Services (AWS) or Microsoft Azure—with orchestration among the various platforms.

3.3.8 Advantages of the Hybrid Cloud:

Saving– The hybrid cloud helps organizations save costs, both in infrastructure and in application support. It presents a more moderate initial investment.

Scalability– The hybrid cloud is a system capable of adapting to the demands that each company needs, for space, memory, and speed. By moving as many non-critical functions as possible to the public cloud, the organization can benefit from the scalability of the public cloud and, at the same time, reduce the demand from the private one.

Security– Having the most critical data stored in the private cloud not only ensures that they are well protected but also provides that company information is stored according to the parameters established by current data protection regulations.

Flexibility– Having the advantages of the public and private cloud within reach allows organizations a full range of options when they have to choose which service is best for each distinct need.

3.3.9 Disadvantages of Hybrid Cloud

Reliability– The reliability of the services depends on the technological and financial capacity of the cloud service providers.

Information– The separated information of the company must travel through different nodes to reach their destination, each of them is a source of insecurity.

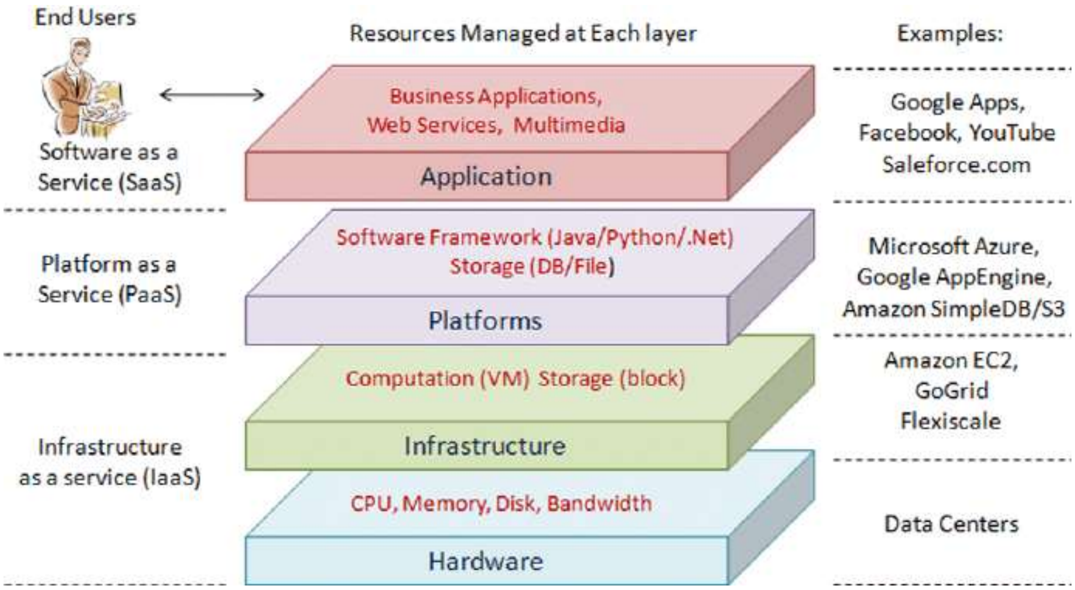
Centralization– The centralization of the applications and the storage of the data creates an interdependence of the service providers.

Security, privacy, and compliance– Security can also be stress in the cloud, mainly if you handle grouped data and customer information. Consistency in the cloud can also become a problem, which may require the creation of a private cloud, if necessary, to protect private data.

Proximity– Ensure that all PC viewing, and programming devices are impeccable with web-based organization, stage, or establishment. While the IT department may have some greater degree of control in the regulation of the mix, proximity is often “what you see is what you get” in terms of incidental expenses.

3.4 IaaS, PaaS, SaaS-Architectural Design Challenges:

Cloud computing is offered in three different ***Service Models*** which each satisfy a unique set of business requirements. These three models are known as Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS).


End Users
Resources Managed at Each layer
Examples:
Software as a Service (SaaS)
Business Applications, Web Services, Multimedia
Application
Google Apps, Facebook, YouTube Saleforce.com
Platform as a Service (PaaS)
Software Framework (Java/Python/.Net) Storage (DB/File)
Platforms
Microsoft Azure, Google AppEngine, Amazon SimpleDB/S3
Computation (VM) Storage (block)
Infrastructure
Amazon EC2, GoGrid Flexiscale
Infrastructure as a service (IaaS)
CPU, Memory, Disk, Bandwidth
Hardware
Data Centers

3.4.1 SaaS: Software as a Service

- Software as a Service, also known as cloud application, services, represents the most utilized option for businesses in the cloud market.

- SaaS utilizes the internet to deliver applications, which are managed by a third-party vendor, to its users.
- A majority of SaaS applications run directly through your web browser, which means they do not require any downloads or installations on the client side.

3.4.2 SaaS Characteristics

- Managed from a central location
- Hosted on a remote server
- Accessible over the internet
- Users not responsible for hardware or software updates

3.4.3 When to Use SaaS

SaaS may be the most beneficial option in several situations, including:

- Startups or small companies that need to launch ecommerce quickly and don't have time for server issues or software
- Short-term projects that require quick, easy, and affordable collaboration
- Applications that aren't needed too often, such as tax software
- Applications that need both web and mobile access

3.4.4 SaaS Limitations & Concerns

- **Interoperability.** Integration with existing apps and services can be a major concern if the SaaS app is not designed to follow open standards for integration. In this case, organizations may need to design their own integration systems or reduce dependencies with SaaS services, which may not always be possible.

- **Vendor lock-in.** Vendors may make it easy to join a service and difficult to get out of it. For instance, the data may not be portable–technically or cost-effectively–across SaaS apps from other vendors without incurring significant cost or inhouse engineering rework. Not every vendor follows standard APIs, protocols, and tools, yet the features could be necessary for certain business tasks.
- **Lack of integration support.** Many organizations require deep integrations with on-premise apps, data, and services. The SaaS vendor may offer limited support in this regard, forcing organizations to invest internal resources in designing and managing integrations. The complexity of integrations can further limit how the SaaS app or other dependent services can be used.

- **Data security.** Large volumes of data may have to be exchanged to the backend data centers of SaaS apps in order to perform the necessary software functionality. Transferring sensitive business information to public-cloud based SaaS service may result in compromised security and compliance in addition to significant cost for migrating large data workloads.
- **Customization.** SaaS apps offer minimal customization capabilities. Since a one-size-fits-all solution does not exist, users may be limited to specific functionality, performance, and integrations as offered by the vendor. In contrast, on-premise solutions that come with several software development kits (SDKs) offer a high degree of customization options.

- **Lack of control.** SaaS solutions involves handing control over to the third-party service provider. These controls are not limited to the software–in terms of the version, updates, or appearance–but also the data and governance. Customers may therefore need to redefine their data security and governance models to fit the features and functionality of the SaaS service.
- **Feature limitations.** Since SaaS apps often come in a standardized form, the choice of features may be a compromising tradeoff against security, cost, performance, or other organizational policies. Furthermore, vendor lock-in, cost, or security concerns

may mean it’s not viable to switch vendors or services to serve new feature requirements in the future.

- **Performance and downtime.** Because the vendor controls and manages the SaaS service, your customers now depend on vendors to maintain the service’s security and performance. Planned and unplanned maintenance, cyber-attacks, or network issues may impact the performance of the SaaS app despite adequate service level agreement (SLA) protections in place.

3.4.5 Examples of SaaS

Popular examples of SaaS include:

- Google Workspace (formerly GSuite)
- Dropbox
- Salesforce
- Cisco WebEx
- SAP Concur
- GoToMeeting

3.4.6 PaaS: Platform as a Service

Cloud platform services, also known as Platform as a Service (PaaS), provide cloud components to certain software while being used mainly for applications. PaaS delivers a framework for developers that they can build upon and use to create customized applications. All servers, storage, and networking can be managed by the enterprise or a third-party provider while the developers can maintain management of the applications.

3.4.7 PaaS Delivery

The delivery model of PaaS is similar to SaaS, except instead of delivering the software over the internet, PaaS provides a platform for software creation. This platform is delivered via the web, giving developers the freedom to concentrate on building the software without having to worry about operating systems, software updates, storage, or infrastructure.

PaaS allows businesses to design and create applications that are built into the PaaS with special software components. These applications, sometimes called middleware, are scalable and highly available as they take on certain cloud characteristics.

3.4.8 PaaS Advantages

No matter the size of your company, using PaaS offers numerous advantages, including:

- Simple, cost-effective development and deployment of apps
- Scalable
- Highly available
- Developers can customize apps without the headache of maintaining the software
- Significant reduction in the amount of coding needed
- Automation of business policy
- Easy migration to the hybrid model

3.4.9 PaaS Characteristics

PaaS has many characteristics that define it as a cloud service, including:

- Builds on virtualization technology, so resources can easily be scaled up or down as your business changes
- Provides a variety of services to assist with the development, testing, and deployment of apps
- Accessible to numerous users via the same development application
- Integrates web services and databases

3.4.10 When to Use PaaS

Utilizing PaaS is beneficial, sometimes even necessary, in several situations. For example, PaaS can streamline workflows when multiple developers are working on the same development project. If other vendors must be included, PaaS can provide great speed and flexibility to the entire process. PaaS is particularly beneficial if you need to create customized applications.

This cloud service also can greatly reduce costs and it can simplify some challenges that come up if you are rapidly developing or deploying an app.

3.4.11 PaaS Limitations & Concerns

- **Data security.** Organizations can run their own apps and services using PaaS solutions, but the data residing in third-party, vendor-controlled cloud servers poses security risks and concerns. Your security options may be limited as customers may not be able to deploy services with specific hosting policies.

- **Integrations.** The complexity of connecting the data stored within an onsite data center or off-premise cloud is increased, which may affect which apps and services can be adopted with the PaaS offering. Particularly when not every component of a legacy IT system is built for the cloud, integration with existing services and infrastructure may be a challenge.
- **Vendor lock-in.** Business and technical requirements that drive decisions for a specific PaaS solution may not apply in the future. If the vendor has not provisioned convenient migration policies, switching to alternative PaaS options may not be possible without affecting the business.

- **Customization of legacy systems.** PaaS may not be a plug-and-play solution for existing legacy apps and services. Instead, several customizations and configuration changes may be necessary for legacy systems to work with the PaaS service. The resulting customization can result in a complex IT system that may limit the value of the PaaS investment altogether.

- **Runtime issues.** In addition to limitations associated with specific apps and services, PaaS solutions may not be optimized for the language and frameworks of your choice. Specific framework versions may not be available or perform optimally with the PaaS service. Customers may not be able to develop custom dependencies with the platform.
- **Operational limitation.** Customized cloud operations with management automation workflows may not apply to PaaS solutions, as the platform tends to limit operational capabilities for end users. Although this is intended to reduce the operational burden on end users, the loss of operational control may affect how PaaS solutions are managed, provisioned, and operated.

3.4.12 Examples of PaaS

Popular examples of PaaS include:

- AWS Elastic Beanstalk
- Windows Azure
- Heroku
- Force.com
- Google App Engine
- OpenShift

3.4.13 IaaS: Infrastructure as a Service

Cloud infrastructure services, known as Infrastructure as a Service (IaaS), are made of highly scalable and automated compute resources. IaaS is fully self-service for accessing and monitoring computers, networking, storage, and other services. IaaS allows businesses to purchase resources on-demand and as-needed instead of having to buy hardware outright.

3.4.14 IaaS Delivery

IaaS delivers cloud computing infrastructure, including servers, network, operating systems, and storage, through virtualization technology. These cloud servers are typically provided to the organization through a dashboard or an API, giving IaaS clients complete control over the entire infrastructure. IaaS provides the same technologies and capabilities as a traditional data center without having to physically maintain or manage all of it. IaaS clients can still access their servers and storage directly, but it is all outsourced through a “virtual data center” in the cloud.

As opposed to SaaS or PaaS, IaaS clients are responsible for managing aspects such as applications, runtime, OSes, middleware, and data. However, providers of the IaaS manage the servers, hard drives, networking, virtualization, and storage. Some providers even offer more services beyond the virtualization layer, such as databases or message queuing.

3.4.15 IaaS Advantages

IaaS offers many advantages, including:

- The most flexible cloud computing model
- Easy to automate deployment of storage, networking, servers, and processing power
- Hardware purchases can be based on consumption
- Clients retain complete control of their infrastructure
- Resources can be purchased as-needed
- Highly scalable

3.4.16 IaaS Characteristics

Characteristics that define IaaS include:

- Resources are available as a service
- Cost varies depending on consumption
- Services are highly scalable
- Multiple users on a single piece of hardware
- Organization retain complete control of the infrastructure
- Dynamic and flexible

3.4.17 When to Use IaaS

Just as with SaaS and PaaS, there are specific situations when IaaS is most advantageous.

- **Startups and small companies** may prefer IaaS to avoid spending time and money on purchasing and creating hardware and software.
- **Larger companies** may prefer to retain complete control over their applications and infrastructure, but they want to purchase only what they actually consume or need.
- **Companies experiencing rapid growth** like the scalability of IaaS, and they can change out specific hardware and software easily as their needs evolve.

Anytime you are unsure of a new application's demands, IaaS offers plenty of flexibility and scalability.

3.4.18 IaaS Limitations & Concerns:

- **Security.** While the customer is in control of the apps, data, middleware, and the OS platform, security threats can still be sourced from the host or other virtual machines (VMs). Insider threat or system vulnerabilities may expose data communication between the host infrastructure and VMs to unauthorized entities.
- **Legacy systems operating in the cloud.** While customers can run legacy apps in the cloud, the infrastructure may not be designed to deliver specific controls to secure the legacy apps. Minor enhancement to legacy apps may be required before migrating

them to the cloud, possibly leading to new security issues unless adequately tested for security and performance in the IaaS systems.

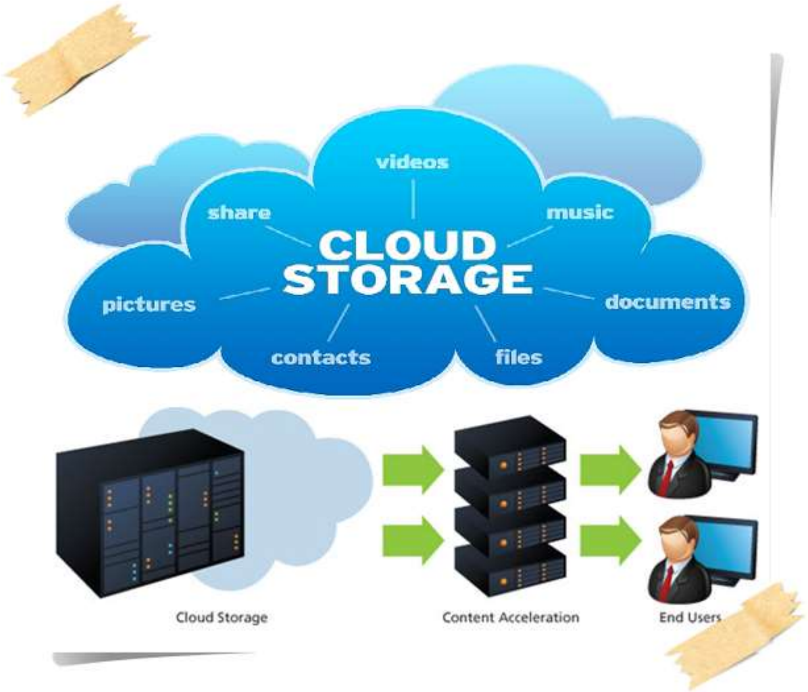
- **Internal resources and training.** Additional resources and training may be required for the workforce to learn how to effectively manage the infrastructure. Customers will be responsible for data security, backup, and business continuity. Due to inadequate control into the infrastructure however, monitoring and management of the resources may be difficult without adequate training and resources available inhouse.
- **Multi-tenant security.** Since the hardware resources are dynamically allocated across users as made available, the vendor is required to ensure that other customers cannot access data deposited to storage assets by previous customers. Similarly, customers must rely on the vendor to ensure that VMs are adequately isolated within the multitenant cloud architecture.

3.4.19 Examples of IaaS

Popular examples of IaaS include:

- DigitalOcean
- Linode
- Rackspace
- Amazon Web Services (AWS)
- Cisco Metacloud
- Microsoft Azure
- Google Compute Engine (GCE)

3.5 Cloud Storage:



- Cloud storage is a cloud computing model that stores data on the Internet through a cloud computing provider who manages and operates data storage as a service.
- It's delivered on demand with just-in-time capacity and costs and eliminates buying and managing your own data storage infrastructure.

3.6 Storage as a Service (STaaS)

- Storage as a Service (STaaS) is a cloud computing model in which subscribers can rent storage from cloud providers.
- The service is mainly used to solve storage issues and offsite backup challenges.
- Storage is rented out on subscription (monthly, yearly or space) basis.

3.7 Advantages of Cloud Storage:

Cost- factually speaking, backing up data isn’t always cheap, especially when take the cost of equipment into account. Additionally, there is the cost of the time it takes to manually complete routine backups. Storage as a service reduces much of the cost associated with traditional backup methods, providing ample storage space in the cloud for a low monthly fee.

Invisibility- Storage as a service is invisible, as no physical presence of it is seen in its deployment and so it doesn’t take up valuable office space.

Security- In this service type, data is encrypted both during transmission and while at rest, ensuring no unauthorized user access to files.

Automation- Storage as a service makes the tedious process of backing up easy to accomplish through automation. Users can simply select what and when they want to backup, and the service does all the rest.

Accessibility- By going for storage as a service, users can access data from smart phones, netbooks to desktops and so on.

Syncing- Syncing ensures your files are automatically updated across all of your devices. This way, the latest version of a file a user saved on their desktop is available on your smart phone.

Sharing- Online storage services allow the users to easily share data with just a few clicks

Collaboration- Cloud storage services are also ideal for collaboration purposes. They allow multiple people to edit and collaborate on a single file or document. Thus, with this feature users need not worry about tracking the latest version or who has made what changes.

Data Protection- By storing data on cloud storage services, data is well protected by all kind of catastrophes such as floods, earthquakes and human errors.

Disaster Recovery- as said earlier, data stored in cloud is not only protected from catastrophes by having the same copy at several places but can also favour disaster recovery to ensure business continuity.

3.8 Cloud Storage Providers:

- **Amazon S3(Simple Storage Service)** –object storage service that offers industry-leading scalability, data availability, security, and performance.
- **Icedrive –** Best mobile cloud storage service
- **pCloud** – Best cloud storage service with lifetime access
- **Zoolz –** Best for high-volume, long-term cloud storage
- **IDrive** – Best cloud storage service for backups
- **Sync** – Best for heavily regulated industries
- **LiveDrive** – Best for remote team collaboration
- **Google Drive** – Best free cloud storage service
- **Microsoft OneDrive** – Best cloud storage service for PC power users
- **Apple's iCloud** – Best cloud storage service for Apple power users
- **Dropbox** – Most popular cloud storage service

3.9 Amazon S3 (Simple Storage Service):

Amazon Simple Storage Service is storage for the Internet. It is designed to make web-scale computing easier for developers. Amazon S3 has a simple web services interface that you can use to store and retrieve any amount of data, at any time, from anywhere on the web.

3.9.1 Amazon S3 Features:

- **Low cost and Easy to Use −** Using Amazon S3, the user can store a large amount of data at very low charges.
- **Secure −** Amazon S3 supports data transfer over SSL and the data gets encrypted automatically once it is uploaded. The user has complete control over their data by configuring bucket policies using AWS IAM.
- **Scalable −** Using Amazon S3, there need not be any worry about storage concerns. We can store as much data as we have and access it anytime.

- **Higher performance** − Amazon S3 is integrated with Amazon CloudFront, that distributes content to the end users with low latency and provides high data transfer speeds without any minimum usage commitments.
- **Integrated with AWS services** − Amazon S3 integrated with AWS services include Amazon CloudFront, Amazon CLoudWatch, Amazon Kinesis, Amazon RDS, Amazon Route 53, Amazon VPC, AWS Lambda, Amazon EBS, Amazon Dynamo DB, etc.

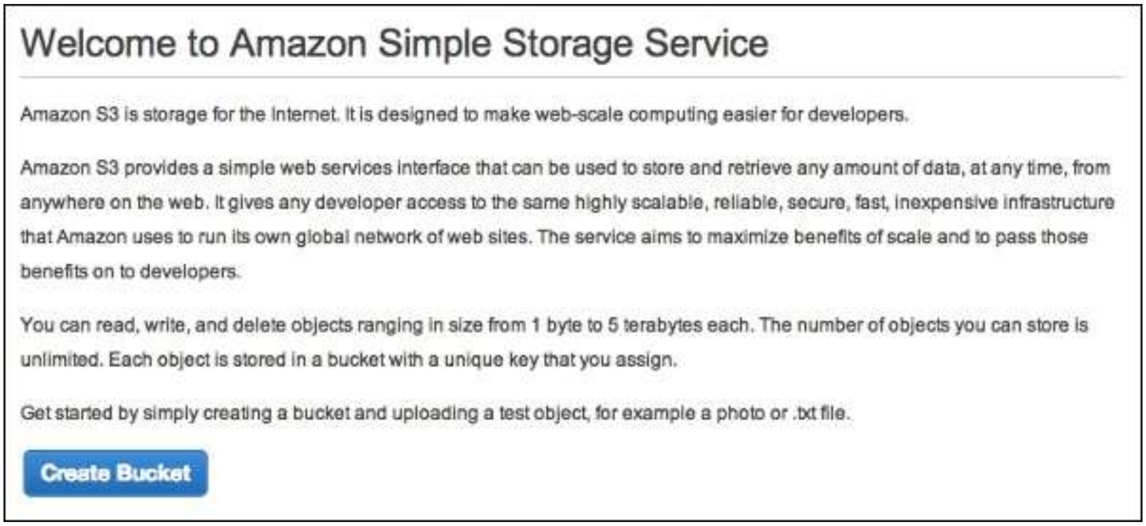
3.9.2 How to Configure S3?

Following are the steps to configure a S3 account.

Step 1 − Open the Amazon S3 console using this link − https://console.aws.amazon.com/s3/home

Step 2 − Create a Bucket using the following steps.

A prompt window will open. Click the Create Bucket button at the bottom of the page.



Welcome to Amazon Simple Storage Service

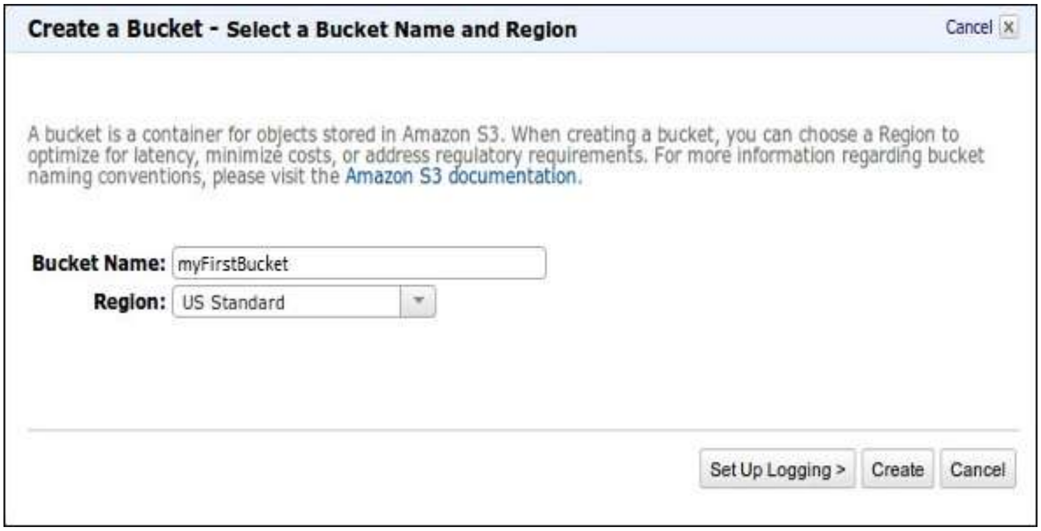
Amazon S3 is storage for the Internet. It is designed to make web-scale computing easier for developers.

Amazon S3 provides a simple web services interface that can be used to store and retrieve any amount of data, at any time, from anywhere on the web. It gives any developer access to the same highly scalable, reliable, secure, fast, inexpensive infrastructure that Amazon uses to run its own global network of web sites. The service aims to maximize benefits of scale and to pass those benefits on to developers.

You can read, write, and delete objects ranging in size from 1 byte to 5 terabytes each. The number of objects you can store is unlimited. Each object is stored in a bucket with a unique key that you assign.

Get started by simply creating a bucket and uploading a test object, for example a photo or .txt file.

Create a Bucket dialog box will open. Fill the required details and click the Create button.



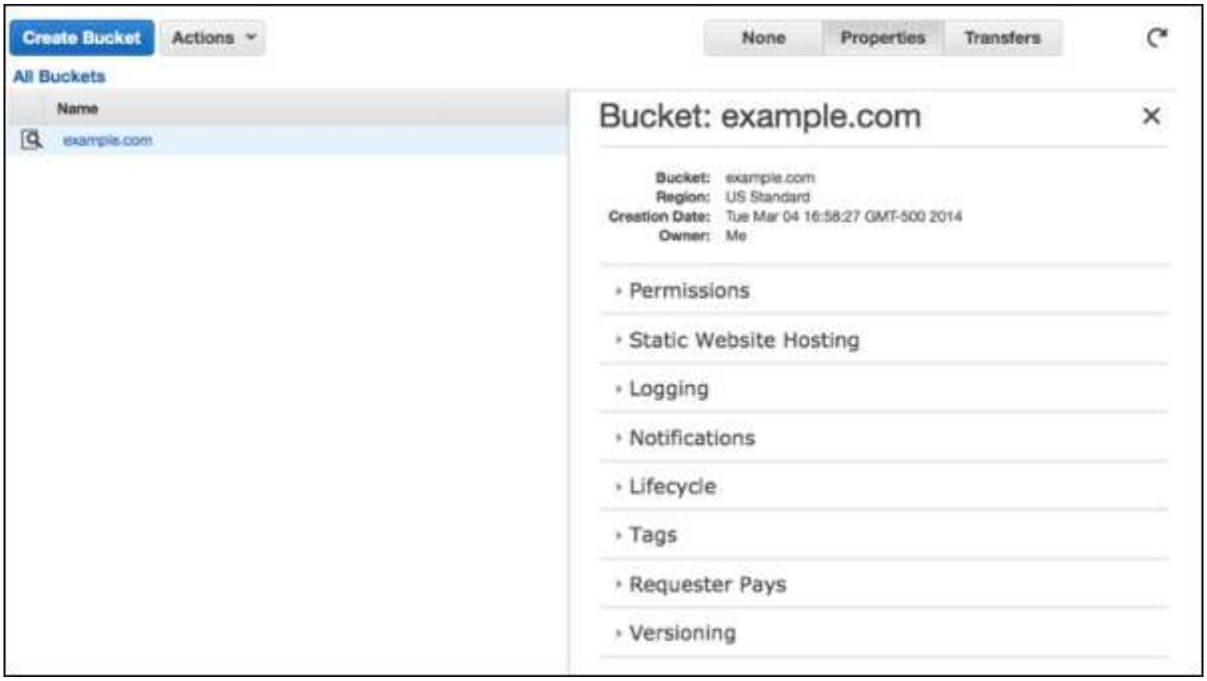
Create a Bucket - Select a Bucket Name and Region

A bucket is a container for objects stored in Amazon S3. When creating a bucket, you can choose a Region to optimize for latency, minimize costs, or address regulatory requirements. For more information regarding bucket naming conventions, please visit the Amazon S3 documentation.

Bucket Name: myFirstBucket

Region: US Standard

The bucket is created successfully in Amazon S3. The console displays the list of buckets and its properties.



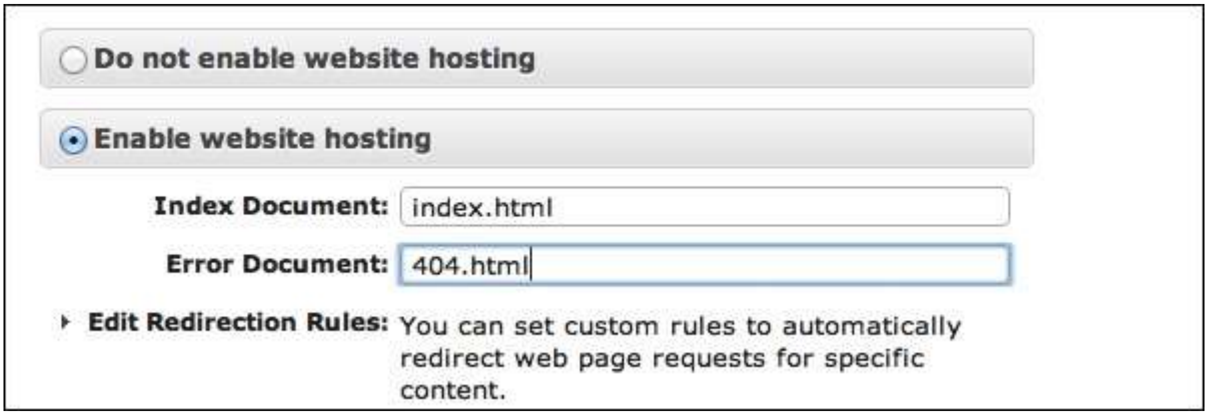
All Buckets

Name: example.com

Bucket: example.com

Bucket: example.com
Region: US Standard
Creation Date: Tue Mar 04 16:58:27 GMT-500 2014
Owner: Me

Select the Static Website Hosting option. Click the radio button Enable website hosting and fill the required details.



○ Do not enable website hosting

● Enable website hosting

Index Document: index.html

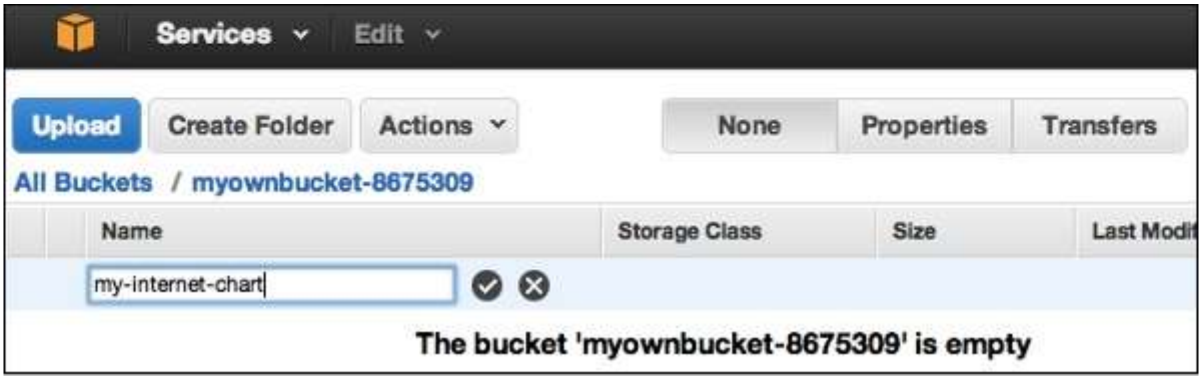
Error Document: 404.html

Edit Redirection Rules: You can set custom rules to automatically redirect web page requests for specific content.

Step 3 − Add an Object to a bucket using the following steps.

Open the Amazon S3 console using the following link − https://console.aws.amazon.com/s3/home

Click the Upload button.

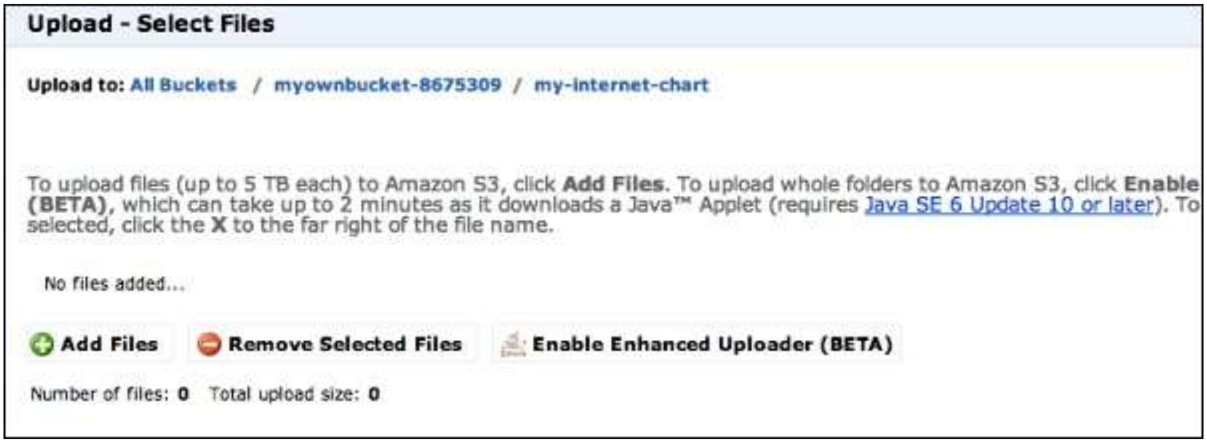


All Buckets / myownbucket-8675309

Name	Storage Class	Size	Last Modi
my-internet-chart			

The bucket 'myownbucket-8675309' is empty

Click the Add files option. Select those files which are to be uploaded from the system and then click the Open button.



Upload - Select Files

Upload to: All Buckets / myownbucket-8675309 / my-internet-chart

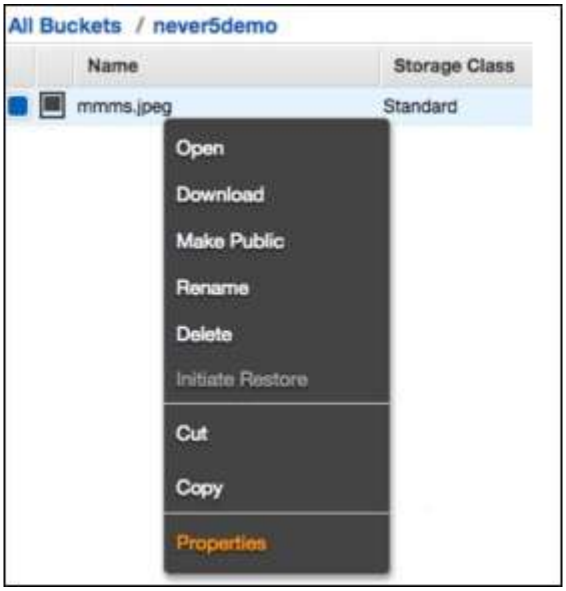
To upload files (up to 5 TB each) to Amazon S3, click **Add Files**. To upload whole folders to Amazon S3, click **Enable** **(BETA)**, which can take up to 2 minutes as it downloads a Java™ Applet (requires Java SE 6 Update 10 or later). To selected, click the **X** to the far right of the file name.

No files added...

Number of files: **0** Total upload size: **0**

Click the start upload button. The files will get uploaded into the bucket.

To open/download an object − In the Amazon S3 console, in the Objects & Folders list, right-click on the object to be opened/downloaded. Then, select the required object.



All Buckets / never5demo

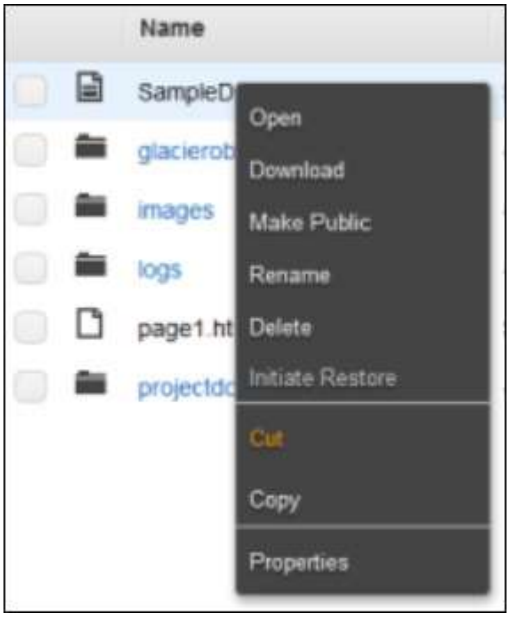
Name	Storage Class
mmms.jpeg	Standard

How to Move S3 Objects?

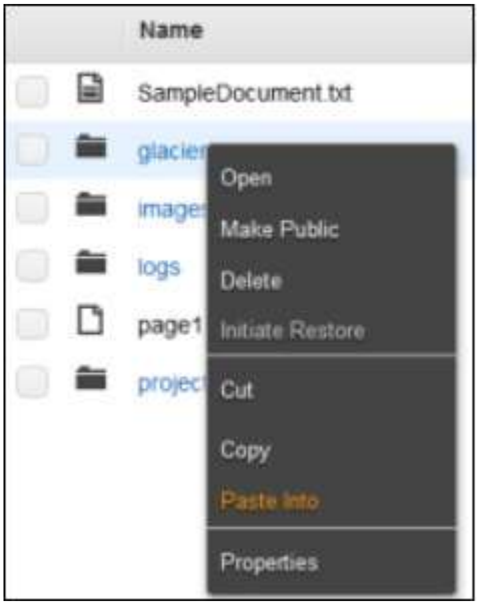
Following are the steps to move S3 objects.

step 1 − Open Amazon S3 console.

step 2 − Select the files & folders option in the panel. Right-click on the object that is to be moved and click the Cut option.



step 3 − Open the location where we want this object. Right-click on the folder/bucket where the object is to be moved and click the Paste into option.

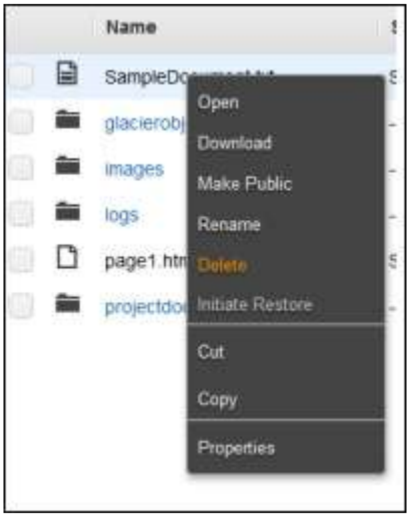


How to Delete an Object?

Step 1 − Open Amazon S3.

Step 2 − Select the files & folders option in the panel. Right-click on the object that is to be deleted. Select the delete option.

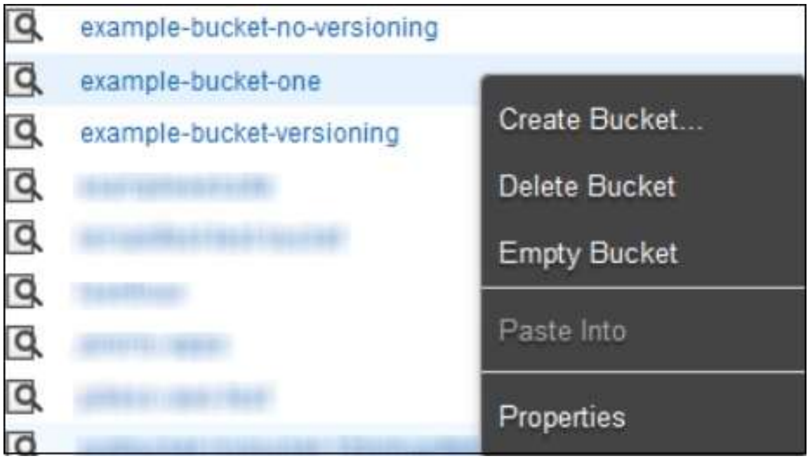
Step 3 − A pop-up window will open for confirmation. Click Ok.



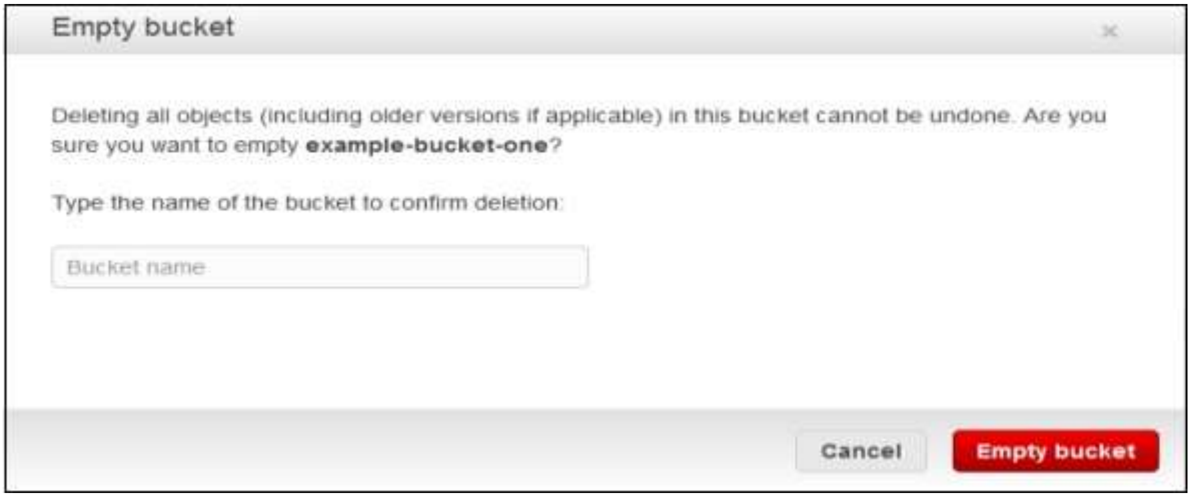
How to Empty a Bucket?

Step 1 − Open Amazon S3 console.

Step 2 − Right-click on the bucket that is to be emptied and click the empty bucket option.



Step 3 − A confirmation message will appear on the pop-up window. Read it carefully and click the **Empty bucket** button to confirm.



Deleting all objects (including older versions if applicable) in this bucket cannot be undone. Are you sure you want to empty **example-bucket-one**?

Type the name of the bucket to confirm deletion:

Bucket name

Important Questions:

1. Identify NIST cloud computing reference architecture with a neat schematic diagram.
2. Give the various Service-Models of Cloud-Computing.
3. How the various deployment models of Cloud-Computing are helpful to the users.
4. What are the benefits of having layered architecture in cloud-computing?
5. What are the various design challenges in IaaS, PaaS, and SaaS models?
6. Compare IaaS, PaaS, and SaaS with examples.
7. What do you understand by Cloud orchestration? How it is helpful in cloud computing process.
8. Define Storage-as-a Service w.r.t. to cloud-computing also give the various benefits.
9. Define the role of Amazon S3(Simple Storage Service) in cloud computing. How to configure S3 give steps.
10. List the name of the venders who provide cloud services to users.

UNIT-4

RESOURCE MANAGEMENT AND SECURITY IN CLOUD

4.1 Inter Cloud Resource Management:

- The inter cloud resource management services are built to perform resource discovery, match, select, composition, negotiate, schedule and monitor operations.
- The software agents are built with decision making and agent interaction features.
- The agent communications are carried out with coordination, cooperation and negotiation models.
- The inter cloud resource management framework is composed to solve the resource management and communication needs.

4.1.1 Taxonomy of Resource Management:

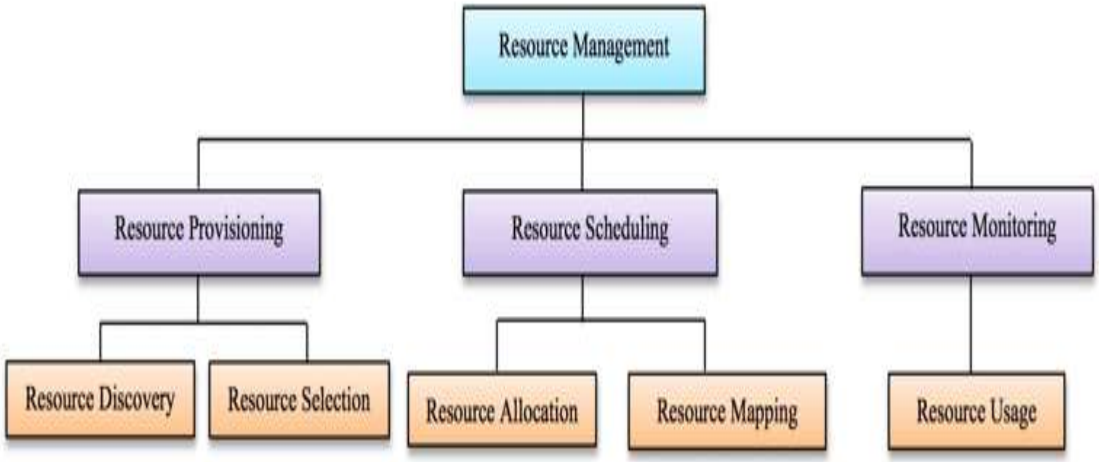


Fig 4.1 Taxonomy of Resource Management

In general, Resource Management can be classified as follows:

1. Resource Provisioning

2. Resource Scheduling
3. Resource monitoring

1. Resource Provisioning:

Resource provisioning means the selection, deployment, and run-time management of software (e.g., database management servers, load balancers) and hardware resources (e.g., CPU, storage, and network) for ensuring guaranteed performance for applications.

Management of resources by resource provisioning further includes ***resource discovery*** *and* ***resource selection.***

In ***resource discovery*** searching for various resources are performed by different users and then in ***resource selection*** any resource of user’s choice is being selected.

2. Resource Scheduling:

Resource scheduling indicates the process of organizing the resources among the different cloud users according to certain rules and regulation of resources usage under a specified cloud environment. Resource scheduling in resource management is the basic technology of cloud computing.

Resource scheduling further includes the ***Resource Allocation*** and ***Resource Mapping.***

Resource allocation is one of the main influential factors to provide efficient and economical processing of resources in the cloud environment. Resource Allocation describes the allotment of resources to different users of their choice

Resource mapping is a strategy for identifying and analysing the resources, and other services that currently exist in cloud. It describes that how many instances of any resource is currently being allotted to any user and for how much time, when this resource will get free and allotted to next user etc.

3. Resource Monitoring:

Resource monitoring is a method of reviewing, observing, and managing the operational workflow in a cloud-based IT infrastructure. Manual or automated management techniques confirm the availability and performance of websites, servers, applications, and other cloud infrastructure.

It includes ***Resource Usage***.

The ***Resource Usage*** view lists for each resource the assigned tasks and the total amount of work that the resource is scheduled to perform on each task, whether per day, week, month, or other time.

4.2 Resource Provisioning

- Provisioning is the process of setting up IT infrastructure. It can also refer to the steps required to manage access to data and resources and make them available to users and systems.

- Provisioning is not the same thing as configuration, but they are both steps in the deployment process. Once something has been provisioned, the next step is configuration.

- When the term "provisioning" is used, it can mean many different types of provisioning, such as server provisioning, network provisioning, user provisioning, service provisioning, and more.

- This resource provisioning takes Service Level Agreement (SLA) into consideration for providing service to the cloud users.

- This is an initial agreement between the cloud users and cloud service providers which ensures Quality of Service (QoS) parameters like performance, availability, reliability, response time etc.

4.3 Resource Provisioning Methods:

1. Static Provisioning
2. Dynamic Provisioning
3. User Self-Provisioning

1. **Static Provisioning:**

   - For applications that have predictable and generally unchanging demands/workloads, it is possible to use “static provisioning" effectively.

   - With advance provisioning, the customer contracts with the provider for services and the provider prepares the appropriate resources in advance of start of service.

   - The customer is charged a flat fee or is billed on a monthly basis.

2. **Dynamic Provisioning:**

   - In cases where demand by applications may change or vary, “dynamic provisioning" techniques have been suggested
   - In this method VMs may be migrated on-the-fly to new compute nodes within the cloud.
   - With dynamic provisioning, the provider allocates more resources as they are needed and removes them when they are not.
   - The customer is billed on a pay-per-use basis. When dynamic provisioning is used to create a hybrid cloud, it is sometimes referred to as cloud bursting.

3. **User Self Provisioning:**

   - With user self- provisioning (also known as cloud self- service), the customer purchases resources from the cloud provider through a web form.
   - User creates a customer account and pay for resources with a credit card.
   - The provider's resources are available for customer use.

4.3.1 Parameters for Resource Provisioning

- Response time: The resource provisioning algorithm designed must take minimal time to respond when executing the task.

- Minimize Cost: From the Cloud user point of view cost should be minimized.
- Revenue Maximization: This is to be achieved from the Cloud Service Provider's view.
- Fault tolerant: The algorithm should continue to provide service in spite of failure of nodes.
- Reduced SLA Violation: The algorithm designed must be able to reduce SLA violation.
- Reduced Power Consumption: VM placement & migration techniques must lower power consumption.

4.4 Global Exchange of Cloud Resources:

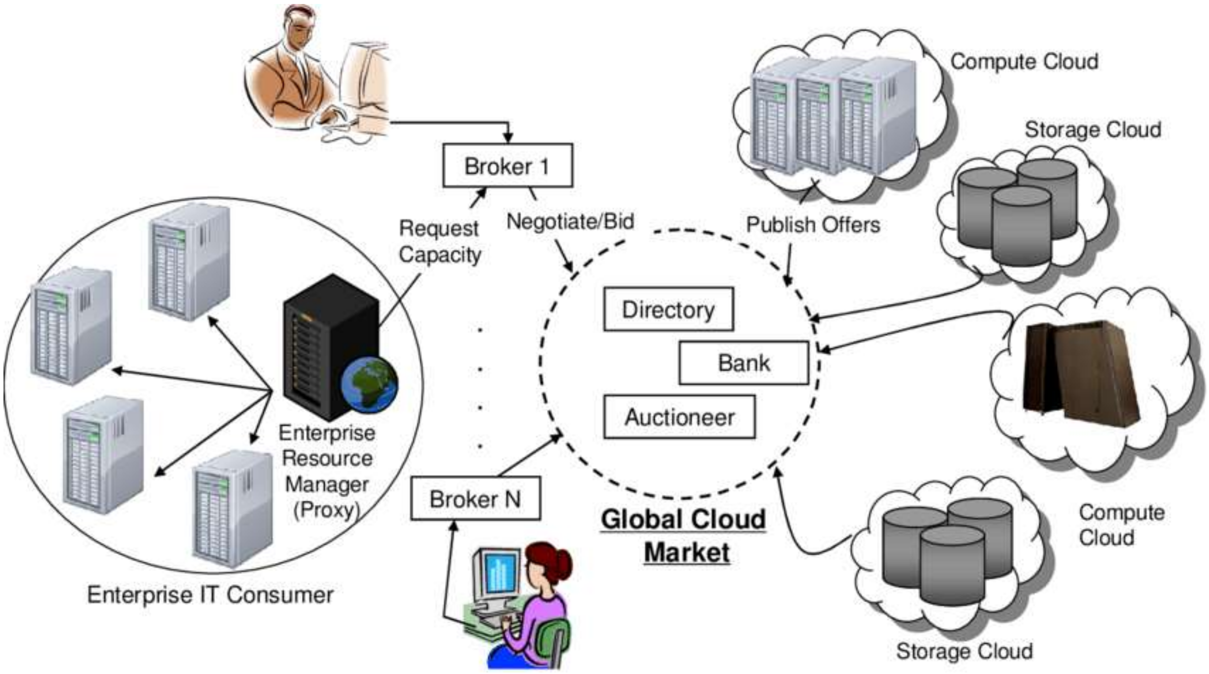


Fig 4.2 Global Exchange of Cloud Resources

The various steps of exchanging resources can be given as follows:

- The market directory allows participants to locate providers or consumers with the right offers.
- Auctioneers periodically clear bids and asks received from market participants.

- The banking system ensures that financial transactions pertaining to agreements between participants are carried out.
- Brokers perform the same function in such a market as they do in real-world markets: they mediate between consumers and providers by buying capacity from the provider and sub-leasing these to the consumers. A broker can accept requests from many users who have a choice of submitting their requirements to different brokers.
- Consumers, brokers and providers are bound to their requirements and related compensations through SLAs. An SLA specifies the details of the service to be provided in terms of metrics agreed upon by all parties, and penalties for meeting and violating the expectations, respectively.
- A provider is equipped with a price-setting mechanism which sets the current price for the resource based on market conditions, user demand, and current level of utilization of the resource. Pricing can be either fixed or variable depending on the market conditions.
- An admission-control mechanism at a provider’s end selects the auctions to participate in or the brokers to negotiate with, based on an initial estimate of the utility. The negotiation process proceeds until an SLA is formed or the participants decide to break off.
- These mechanisms interface with the resource management systems of the provider in order to guarantee the allocation being offered or negotiated can be reclaimed, so that SLA violations do not occur.
- The resource management system also provides functionalities such as advance reservations that enable guaranteed provisioning of resource capacity.
- Brokers gain their utility through the difference between the price paid by the consumers for gaining resource shares and that paid to the providers for leasing their resources. Therefore, a broker has to choose those users whose applications can provide it maximum utility.
- A broker interacts with resource providers and other brokers to gain or to trade resource shares. A broker is equipped with a negotiation module that is informed by the current conditions of the resources and the current demand to make its decisions.

- Consumers have their own utility functions that cover factors such as deadlines, fidelity of results, and turnaround time of applications. They are also constrained by the amount of resources that they can request at any time, usually by a limited budget. Consumers also have their own limited IT infrastructure that is generally not completely exposed to the Internet. Therefore, a consumer participates in the utility market through a resource management proxy that selects a set of brokers based on their offerings. He then forms SLAs with the brokers that bind the latter to provide the guaranteed resources. The enterprise consumer then deploys his own environment on the leased resources or uses the provider's interfaces in order to scale his applications.

4.5 Security Overview:

4.5.1 What Is Cloud Security?

- Cloud security is the **protection** of data stored online via cloud computing platforms from theft, leakage, and deletion.
- Cloud computing security consists of a set of policies, controls, procedures, and technologies that work together to protect cloud-based systems, data, and infrastructure. These security measures are configured to protect cloud data, support regulatory compliance, and protect customers' privacy as well as setting authentication rules for individual users and devices.

4.6 Cloud Security Challenges:

Some of the advanced cloud-native security challenges and the multiple layers of risk faced by today's cloud-oriented organizations include:

Increased Attack Surface

The public cloud environment has become a large and highly attractive attack surface for hackers who exploit poorly secured cloud ingress ports in order to access and disrupt workloads and data in the cloud. Malware, Zero-Day, Account Takeover, and many other malicious threats have become a day-to-day reality.

Lack of Visibility and Tracking

In the IaaS model, the cloud providers have full control over the infrastructure layer and do not expose it to their customers. The lack of visibility and control is further extended in the PaaS and SaaS cloud models. Cloud customers often cannot effectively identify and quantify their cloud assets or visualize their cloud environments.

Ever-Changing Workloads

Cloud assets are provisioned and decommissioned dynamically—at scale and at velocity. Traditional security tools are simply incapable of enforcing protection policies in such a flexible and dynamic environment with its ever-changing and ephemeral workloads.

DevOps, DevSecOps and Automation

Organizations that have embraced the highly automated DevOps CI/CD culture must ensure that appropriate security controls are identified and embedded in code and templates early in the development cycle. Security-related changes implemented *after* a workload has been deployed in production can undermine the organization’s security posture as well as lengthen time to market.

Granular Privilege and Key Management

Often cloud user roles are configured very loosely, granting extensive privileges beyond what is intended or required. One common example is giving database delete or write permissions to untrained users or users who have no business need to delete or add database assets. At the application level, improperly configured keys and privileges expose sessions to security risks.

Complex Environments

Managing security in a consistent way in the hybrid and multicloud environments favoured by enterprises these days requires methods and tools that work seamlessly across public cloud providers, private cloud providers, and on-premises deployments—including branch office edge protection for geographically distributed organizations.

Cloud Compliance and Governance

All the leading cloud providers have aligned themselves with most of the well-known accreditation programs such as PCI 3.2, NIST 800-53, HIPAA and GDPR. However, customers are responsible for ensuring that their workload and data processes are compliant. Given the poor visibility as well as the dynamics of the cloud environment, the compliance audit process becomes close to mission impossible unless tools are used to achieve continuous compliance checks and issue real-time alerts about misconfigurations.

4.6.1 Why is cloud security important?

Cloud security offers many benefits, including:

Centralized security: Just as cloud computing centralizes applications and data, cloud security centralizes protection. Cloud-based business networks consist of numerous devices and endpoints that can be difficult to manage when dealing with shadow IT or BYOD. Managing these entities centrally enhances traffic analysis and web filtering, streamlines the monitoring of network events and results in fewer software and policy updates. Disaster recovery plans can also be implemented and actioned easily when they are managed in one place.

Reduced costs: One of the benefits of utilizing cloud storage and security is that it eliminates the need to invest in dedicated hardware. Not only does this reduce capital expenditure, but it also reduces administrative overheads. Where once IT teams were firefighting security issues reactively, cloud security delivers proactive security features that offer protection 24/7 with little or no human intervention.

Reduced Administration: When you choose a reputable cloud services provider or cloud security platform, you can kiss goodbye to manual security configurations and almost constant security updates. These tasks can have a massive drain on resources, but when you move them to the cloud, all security administration happens in one place and is fully managed on your behalf.

Reliability: Cloud computing services offer the ultimate in dependability. With the right cloud security measures in place, users can safely access data and applications within the cloud no matter where they are or what device they are using.

4.6.2 Methods of Cloud Computing Security:

1. **Cloud Security Controls**
2. **Security**
3. **Privacy**

1. Cloud Security Controls:

Cloud security architecture is effective only if the correct defensive implementations are in place. An efficient cloud security architecture should recognize the issues that will arise with security management. The security management addresses these issues with security controls. These controls are put in place to safeguard any weaknesses in the system and reduce the effect of an attack. While there are many types of controls behind a cloud security architecture, they can usually be found in one of the following categories:

- **Deterrent controls:**
  These controls are intended to reduce attacks on a cloud system. Much like a warning sign on a fence or a property, deterrent controls typically reduce the threat level by informing potential attackers that there will be adverse consequences for them if they proceed. (Some consider them a subset of preventive controls.)

- **Preventive controls:**

  Preventive controls strengthen the system against incidents, generally by reducing if not actually eliminating vulnerabilities. Strong authentication of cloud users, for instance, makes it less likely that unauthorized users can access cloud systems, and more likely that cloud users are positively identified.

- **Detective controls:**

  Detective controls are intended to detect and react appropriately to any incidents that occur. In the event of an attack, a detective control will signal the preventative or corrective controls to address the issue. System and network security monitoring, including intrusion detection and prevention arrangements, are typically employed to detect attacks on cloud systems and the supporting communications infrastructure.

- **Corrective controls:**

  Corrective controls reduce the consequences of an incident, normally by limiting the damage. They come into effect during or after an incident. Restoring system backups in order to rebuild a compromised system is an example of a corrective control.

2. **Security:** It deals with following activities for the purpose of cloud security:

- **Identity management:**

  Every enterprise will have its own identity management system to control access to information and computing resources. Cloud providers either integrate the customer's identity management system into their own infrastructure, using federation or SSO technology or a biometric-based identification system or provide an identity management system of their own.

- **Physical security:**

Cloud service providers physically secure the IT hardware (servers, routers, cables etc.) against unauthorized access, interference, theft, fires, floods etc. and ensure that essential supplies (such as electricity) are sufficiently robust to minimize the possibility of disruption. This is normally achieved by serving cloud applications from ‘world-class’ (i.e. professionally specified, designed, constructed, managed, monitored and maintained) data centers.

- **Personnel security:**

  Various information security concerns relating to the IT and other professionals associated with cloud services are typically handled through pre-, para- and post-employment activities such as security screening potential recruits, security awareness, and training programs, proactive.

3. Privacy:

Providers ensure that all critical data (credit card numbers, for example) are masked or encrypted and that only authorized users have access to data in its entirety. Moreover, digital identities and credentials must be protected as should any data that the provider collects or produces about customer activity in the cloud.

- **Data security:**

  A number of security threats are associated with cloud data services: not only traditional security threats, such as network eavesdropping, illegal invasion, and denial of service attacks, but also specific cloud computing threats, such as side-channel attacks, virtualization vulnerabilities, and abuse of cloud services. The following security requirements limit the threats.

- **Confidentiality:**

  Data confidentiality is the property that data contents are not made available or disclosed to illegal users. Outsourced data is stored in a cloud and out of the owners’ direct control. Only authorized users can access the sensitive data while others, including CSPs, should not gain any information about the data. Meanwhile, data owners expect to fully utilize cloud data services, e.g., data search, data computation, and data sharing, without the leakage of the data contents to CSPs or other adversaries.

- **Access controllability:**

  Access controllability means that a data owner can perform the selective restriction of access to her or his data outsourced to the cloud. Legal users can be authorized by the owner to access the data, while others can not access it without permissions.

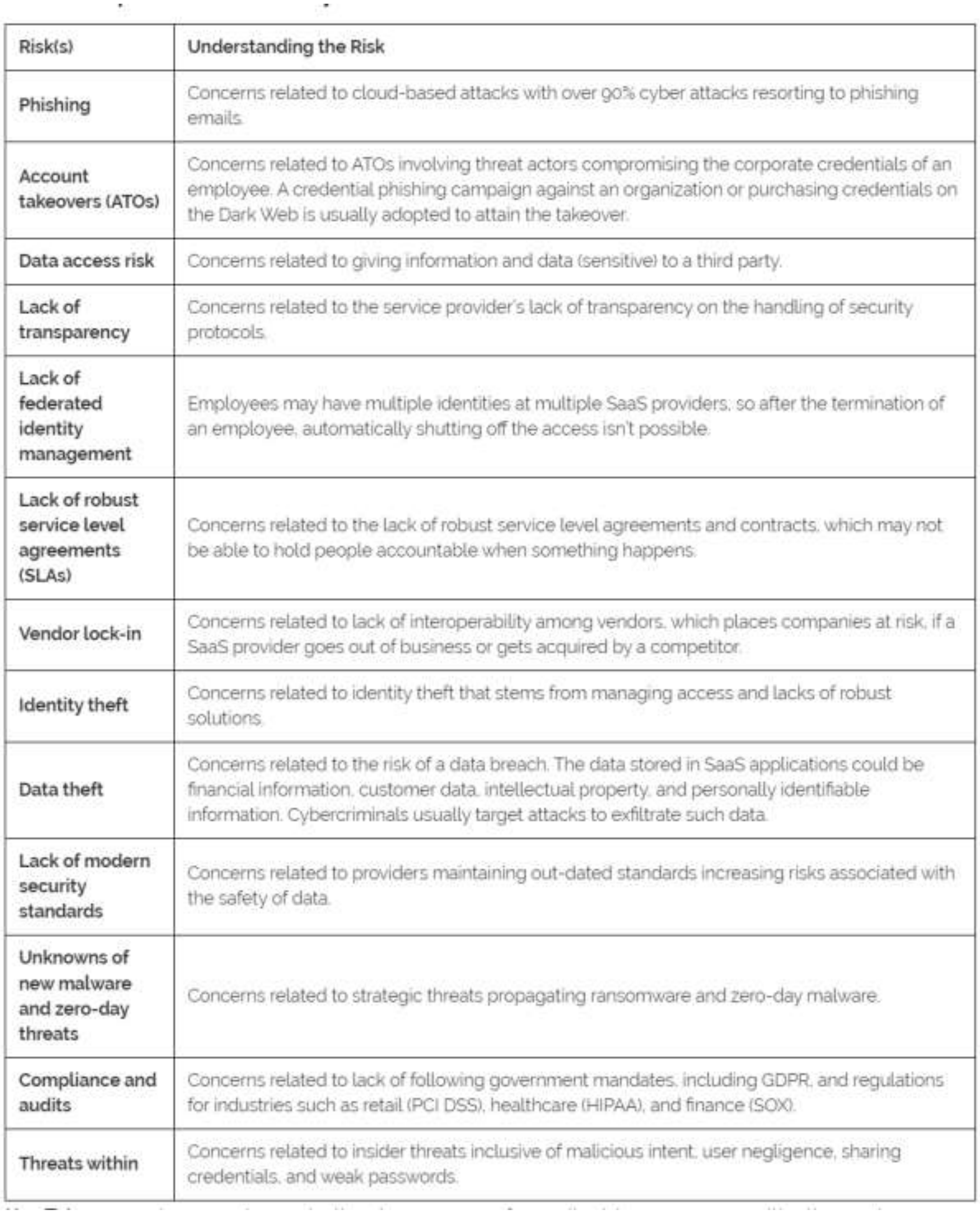
- **Integrity:**

  Data integrity demands to maintain and assure the accuracy and completeness of data. A data owner always expects that data in a cloud can be stored correctly and trustworthily. It means that the data should not be illegally tampered, improperly modified, deliberately deleted, or maliciously fabricated. If any undesirable operations corrupt or delete the data, the owner should be able to detect the corruption or loss.

4.7 Software-as-a-Service (SaaS) Security:

SaaS security refers to securing user privacy and corporate data in subscription-based cloud applications. SaaS applications carry a large amount of sensitive data and can be accessed from almost any device by a mass of users, thus poking a risk to privacy and sensitive information. The various risk to SaaS security is given in following table

Table 4.1 Biggest SaaS Security Risks



Risk(s)	Understanding the Risk
Phishing	Concerns related to cloud-based attacks with over 90% cyber attacks resorting to phishing emails.
Account takeovers (ATOs)	Concerns related to ATOs involving threat actors compromising the corporate credentials of an employee. A credential phishing campaign against an organization or purchasing credentials on the Dark Web is usually adopted to attain the takeover.
Data access risk	Concerns related to giving information and data (sensitive) to a third party.
Lack of transparency	Concerns related to the service provider's lack of transparency on the handling of security protocols.
Lack of federated identity management	Employees may have multiple identities at multiple SaaS providers, so after the termination of an employee, automatically shutting off the access isn't possible.
Lack of robust service level agreements (SLAs)	Concerns related to the lack of robust service level agreements and contracts, which may not be able to hold people accountable when something happens.
Vendor lock-in	Concerns related to lack of interoperability among vendors, which places companies at risk, if a SaaS provider goes out of business or gets acquired by a competitor.
Identity theft	Concerns related to identity theft that stems from managing access and lacks of robust solutions.
Data theft	Concerns related to the risk of a data breach. The data stored in SaaS applications could be financial information, customer data, intellectual property, and personally identifiable information. Cybercriminals usually target attacks to exfiltrate such data.
Lack of modern security standards	Concerns related to providers maintaining out-dated standards increasing risks associated with the safety of data.
Unknowns of new malware and zero-day threats	Concerns related to strategic threats propagating ransomware and zero-day malware.
Compliance and audits	Concerns related to lack of following government mandates, including GDPR, and regulations for industries such as retail (PCI DSS), healthcare (HIPAA), and finance (SOX).
Threats within	Concerns related to insider threats inclusive of malicious intent, user negligence, sharing credentials, and weak passwords.

4.7.1 SaaS security practices:

Below are SaaS security practices that organizations can adopt to protect data in their SaaS applications.

Detect rogue services and compromised accounts:

The average organization uses 1,935 unique cloud services. Unfortunately, the IT departments believe they use only 30 cloud services, according to the 2019 McAfee Cloud Adoption and Risk Report. Moreover, nearly 9% of those cloud services were rated as high-risk services. Organizations can use tools, such as cloud access security brokers (CASB) to audit their networks for unauthorized cloud services and compromised accounts.

Apply identity and access management (IAM):

A role-based identity and access management solution can ensure that end users do not gain access to more resources than they require for their jobs. IAM solutions use processes and user access policies to determine what files and applications a particular user can access. An organization can apply role-based permissions to data so that end users will see only the data they're authorized to view.

Encrypt cloud data:

Data encryption protects both data at rest (in storage) and data in transit between the end user and the cloud or between cloud applications. Government regulations usually require encryption of sensitive data. Sensitive data includes financial information, healthcare data, and personally identifiable information (PII). While a SaaS vendor may provide some type of encryption, an organization can enhance data security by applying its own encryption, such as by implementing a cloud access security broker (CASB).

Enforce data loss prevention (DLP):

DLP software monitors for sensitive data within SaaS applications or outgoing transmissions of sensitive data and blocks the transmission. DLP software detects and prevents sensitive data from being downloaded to personal devices and blocks malware or hackers from attempting to access and download data.

Monitor collaborative sharing of data:

Collaboration controls can detect granular permissions on files that are shared with other users, including users outside the organization who access the file through a web link. Employees may inadvertently or intentionally share confidential documents through email, team spaces, and cloud storage sites such as Dropbox.

Incorporating Security in the SDLC Process:

Integrating security in all the phases of the SDLC process helps with a security review at every phase. The approach creates a stronger application, and you can implement secure coding best practices, especially during code reviews. Enforcing security guidelines can prevent security bugs from creeping in and eliminate significant setbacks. You can also use an excellent static application security testing (SAST) tool to analyse your application source code and highlight the security vulnerabilities, if any.

Securing Deployment:

Deployment can be either done on a public cloud or via a SaaS vendor. When opting for self-deployment, you need to research thoroughly and adopt adequate safeguards. However, if you opt for services of dedicated cloud providers such as Google and Amazon, as a rule, they take care of facets such as network security, data security, data segregation, and more. It is strongly recommended to adopt the security settings as recommended by public cloud vendors while deploying your SaaS application on public clouds.

Integrating Real-time Protection:

Incorporating real-time monitoring through protection logic into the code at the development stage can help to differentiate between legitimate queries and attacks. The output is quite critical and can help to protect the product from breaches and attacks such as SQL injections, account takeovers, and XSS attacks.

4.8 Security Governance:

Cloud security governance refers to the management model that facilitates effective and efficient **security** management and operations in the **cloud** environment so that an enterprise's business targets are achieved.

4.8.1 Cloud Security Governance Challenges

Some of the challenges that Cloud Governance features help us in tackling are:

Performance Management

When any business uses cloud services, it is the service provider's responsibility to supply proper services and enhance performance. If the service provider goes down, then the client's performance using the Cloud Governance services will also go down. To avoid such conditions, a proper Governance Cloud Model, a set of policies, and choosing one of the best Cloud Governance solutions can be helpful.

Governance/Control

A proper set of policies and procedures helps to support your organizational strategies and business goals. It will enhance the business and will reduce the operational cost of the company.

Cost Management

Organizations using cloud services will be in profit as it is a very cost-effective approach. The policies are designed in such a way that, if followed, will result in high profit. A properly considered Cloud Governance Model optimizes cost by conducting better financial analytics and automating policies or keeping management reporting practice will help provide cost management.

Security Issues

The security of the data is also one of the major concerns. It is so because the security lope holes can be avoided by using some strict rules, Cloud Governance best practices, and policies. A Governance Model should build proper authentication policies to protect the confidentiality, integrity, and availability of the information.

4.8.2 Cloud Security Governance Deployment Framework:

The cloud Deployment Framework consist of five phases. Each step considers the critical security issues that must be considered and analyzed by the system developer or the system administrator. The security guideline and potential vulnerabilities in this work are referred to the cloud security alliance (CSA) and NISA risks and vulnerabilities models. Auditing and information security management system standards have also been considered. The various phases have been described below:

1. Initiation Phase
2. Development Phase
3. Implementation Phase
4. Operation phase
5. Destroy

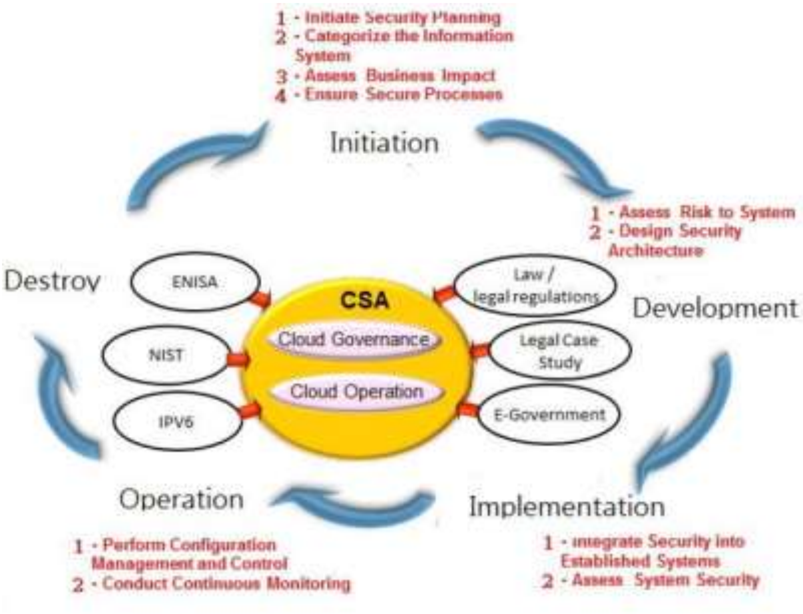

1 - Initiate Security Planning
2 - Categorize the Information System
3 - Assess Business Impact
4 - Ensure Secure Processes
Initiation
1 - Assess Risk to System
2 - Design Security Architecture
Development
Law / legal regulations
Legal Case Study
E-Government
CSA
Cloud Governance
Cloud Operation
ENISA
NIST
IPV6
Destroy
Implementation
1 - integrate Security into Established Systems
2 - Assess System Security
Operation
1 - Perform Configuration Management and Control
2 - Conduct Continuous Monitoring

Fig 4.3 Cloud Security Governance Deployment Framework

1. **Initiation Phase**:

   There are four important procedures that should be emphasized in the initiation phase: initiating security planning, categorizing the information system, accessing business impact, and ensuring process security at the beginning of system development. Three components must be considered in order to achieve these goals, i.e., cloud deployment assessment, enterprise governance and risk management, and law and electronic evidence planning.

2. **Development Phase**:

   In the develop phase, the developer has analyzed the whole technical and non-technical issues. Therefore, they need to consider the secure architecture design and assess the risk of system. Three component are included in develop phase, i.e., compliance and auditing, information lifecycle management and portability and interoperability.

3. **Implementation Phase**:

After completing the initiation and development phases, implementation issues that are closer to the technique related issues must be dealt with. In this phase, security integration into established systems is the main object for seamless evolving management and technique issues. Furthermore, an assessed system security methodology must be applied in this phase. In the implementation phase, four critical security components should be considered in cloud service, including application security, information life cycle, encryption and keys, and virtualization,

4. **Operation Phase**:

In the operation phase, attention is placed on performance management and control. The system administrator also needs to continuously conduct a monitoring mechanism. In this phase, four critical components should be considered, i.e., traditional security business continuity and disaster recovery, data center operations, incident response, and security as a service. These components are highly related to traditional information security related issues,

5. **Destroy:**

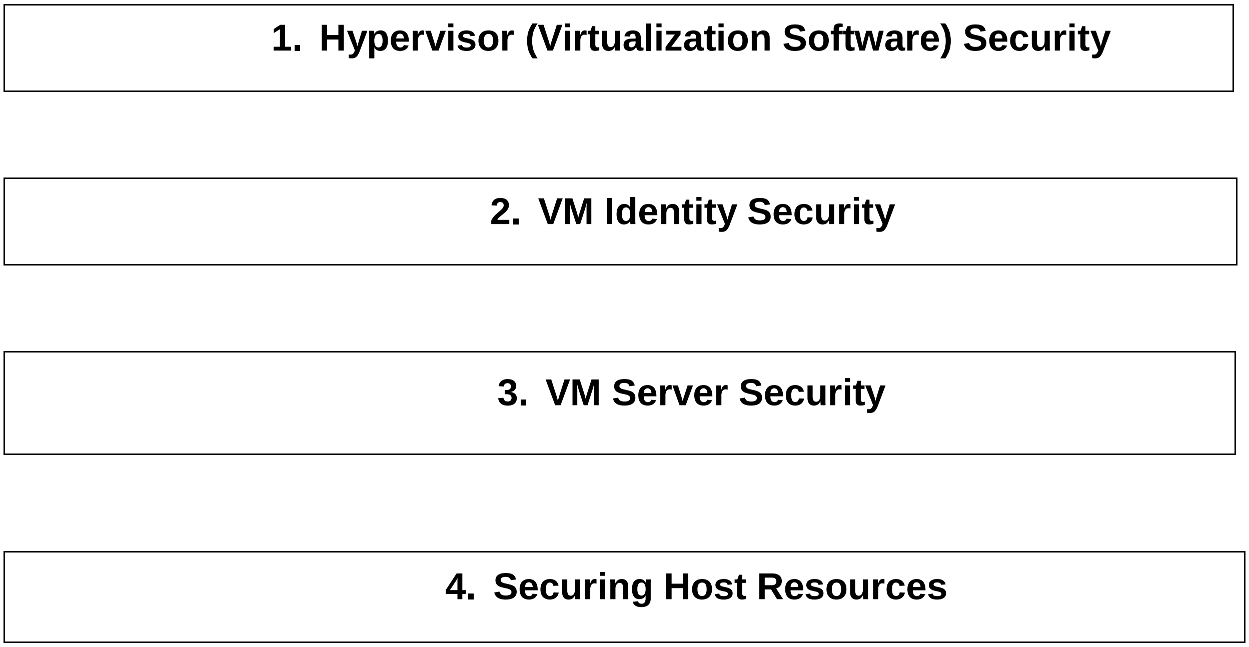
This is the last phase of this life cycle which ensures the successful completion of above 4 processes. After following all these steps provider atleast will be able to ensure the security concerns.

4.9 Virtual Machine Security:

Virtual Machine security, or security virtualization, refers to security solutions that are software-based and designed to work within a virtualized IT environment. This differs from traditional, hardware-based network security, which is static and runs on devices such as traditional firewalls, routers, and switches.

4.9.1 Virtual Machine Security architecture:

To enable the security features to virtual Machine first we need to understand the exact location of virtual machine and then the components connected to it. The architecture of virtual machine security is given below:



1. Hypervisor (Virtualization Software) Security

2. VM Identity Security

3. VM Server Security

4. Securing Host Resources

Fig 4.4 Virtual Machine Security

1. **Hypervisor Security:**

Hypervisor security is the process of ensuring the hypervisor, the software that enables virtualization, is secure throughout its life cycle. This includes during development and in implementation. Common security practices for hypervisors include limiting the users in a local system, limiting attack surfaces, and keeping all systems updated.

2. **VM Identity Security:**

Machine identities govern the confidentiality and integrity of information between machines. To assure their unique identities, machines use keys and certificates, much like people employ usernames and passwords. Without the proper management of machine identities, organizations cannot guarantee the confidentiality of information that flows to authorized machines and prevent the flow of information to unauthorized machines.

3. VM Server Security:

Virtualization offers many advantages, but ineffective virtual server security and compliance policies can offset any benefits. Virtual machine (VM) sprawl and software license management, in particular, might be vulnerable to security breaches or audit failure, which can be costly to fix.

To resolve security and software licensing issues, it’s important to ensure that all physical machines are correctly inventoried, with all virtual environments and related software included in a regularly maintained list.

4. Securing Host Resources:

Host Resource security describes how your server is set up for the following tasks: Preventing attacks. Minimizing the impact of a successful attack on the overall system. Responding to attacks when they occur.

4.9.2 Virtualization security attacks/ issues:

Larger software stacks and greater numbers of APIs, along with a lower degree of security assurance in the code, increase the risk. We highlight the following attacks in virtualized environments

VM escape:

Virtual machines are designed to support strong isolation between the host and the VMs. But the vulnerabilities in the operating system running inside the VM can aid attackers to insert a malicious program into it. When that program is run, VM breaks the isolated boundaries and starts communicating with the operating system directly bypassing the VMM layer. Such an exploit opens the door to attackers to gain access to the host machine and launch further attacks.

Hyperjacking:

Hyperjacking is an attack in which a hacker takes malicious control over the hypervisor that creates the virtual environment within a virtual machine (VM) host. The point of the attack is to target the operating system that is below that of the virtual machines so that the attacker's program can run and the applications on the VMs above it will be completely oblivious to its presence.

VM sprawl:

VM sprawling occurs when a large number of virtual machines exist in the environment without proper management or control. Since they retain the system resources (i.e., memory, disks, network channels etc.) during this period, these resources cannot be assigned to other VMs, and they are effectively lost.

Hypervisor vulnerabilities:

A hypervisor or VMM is formed to run numerous guest VMs and applications simultaneously on a single host machine and to provide separation among the guest VMs. Despite the fact that hypervisors are anticipated to be vigorous and secure, they are accessible to attacks. If attackers gain command of the hypervisor, all the VMs and the data accessed by them will be under their full control to utilize.

Single point of failure:

Existing virtualized environments are based on the hypervisor technology that controls the access of the VMs to physical resources and is important for the overall functioning of the system. Therefore, failure of the hypervisor due to overused infrastructure or software faults leads to the collapse of the overall system.

Inside-VM attack:

VM can get infected with malware or OS rootkits at runtime. Such attack can take complete control of the VM and significantly compromise its security state.

Outside-VM attack:

Attacks from the host OS and co-located VMs are known as outside-VM attacks. Outside-VM attacks are hard for customers to defeat. A malicious VM can potentially access other VMs through shared memory, network connections, and other shared resources. For example, if a malicious VM determines where another VM’s allocated memory lies, then it could read or write to that location and interfere with the other’s operation.

Cross VM side channel attack:

To maximize resource utilization, multiple VMs are usually placed on the same physical server in the Cloud environment and this co-resident placement is a potential threat to cross VM side channel attack. The basic idea is: a malicious VM penetrates the isolation between VMs, and then access the shared hardware and cache locations to extract confidential information from the target VM.

Outdated SW packages in VMs:

Outdated software packages in virtual machines can pose serious security threats in the virtualized environment. Because of the low cost and the ease of creation, users tend to create new virtual machines for different tasks, branch new virtual machines based on the old ones, snapshot machines or even rollback machines to an earlier state. These operations may have serious security implications, for example, a machine rollback operation may expose a software bug that has already been fixed.

4.9.3 How is physical security different from virtualized security?

- Traditional physical security is hardware-based, and as a result, it’s inflexible and static.
- The traditional approach depends on devices deployed at strategic points across a network and is often focused on protecting the network perimeter (as with a traditional firewall). However, the perimeter of a virtualized, cloud-based network is necessarily porous and workloads and applications are dynamically created, increasing the potential attack surface.
- Traditional security also relies heavily upon port and protocol filtering, an approach that’s ineffective in a virtualized environment where addresses and ports are assigned dynamically. In such an environment, traditional hardware-based security is not

enough; a cloud-based network requires virtualized security that can move around the network along with workloads and applications.

4.10 IAM (Identity and access management) Security Standards:

- Identity and access management (IAM) in enterprise IT is about defining and managing the roles and access privileges of individual network users and the circumstances in which users are granted (or denied) those privileges.
- Those users might be customers (customer identity management) or employees (employee identity management.
- The core objective of IAM systems is one digital identity per individual. Once that digital identity has been established, it must be maintained, modified and monitored throughout each user's "access lifecycle."

4.10.1 IAM tools:

API security enables IAM for use with B2B commerce, integration with the cloud, and microservices-based IAM architectures. Forrester sees API security solutions being used for single sign-on (SSO) between mobile applications or user-managed access. This would allow security teams to manage IoT device authorization and personally identifiable data.

- **Customer identity and access management (CIAM)** allow "comprehensive management and authentication of users; self-service and profile management; and integration with CRM, ERP, and other customer management systems and databases," according to the report.

- **Identity analytics (IA)**will allow security teams to detect and stop risky identity behaviours using rules, machine learning, and other statistical algorithms.

- **Identity as a service (IDaaS)** includes "software-as-a-service (SaaS) solutions that offer SSO from a portal to web applications and native mobile applications as well as some level of user account provisioning and access request management," according to the report

- **Identity management and governance (IMG)** provides automated and repeatable ways to govern the identity life cycle. This is important when it comes to compliance with identity and privacy regulations.

- **Risk-based authentication (RBA)** solutions "take in the context of a user session and authentication and form a risk score. The firm can then prompt high-risk users for 2FA and allow low-risk users to authenticate with single factor (e.g., username plus password) credentials," according to the report.

4.10.2 Benefits of IAM Security Standards:

1. Improving User Experiences:

While this may seem the most obvious benefit, it deserves to be said: SSO (Single sign-on) eliminates the need for users to remember and input multiple passwords to access different areas of your system. Gone are the days of trying to keep dozens of password variations straight; with SSO (Single Sign on), users can enjoy automatic logins every time they move to a different connected system. All three vendors offer a variety of user authentication schemes ranging from more strict multi-factor authentication to federated solutions that leverage existing user security profiles.

2. Enhancing Security Profiles:

Just because SSO *can* grant users automatic access to all applications does not mean it *has* to. More advanced IAM systems, most commonly using Security Assertion Markup Language (SAML) 2.0 can use SSO with additional levels of security. IAM systems can authenticate and authorize users based on the access level indicated in their directory profiles. IAM system can also automatically control user access using other factors to specific functions of your system.

3. Simplifies Auditing and Reporting:

Consolidating user identities and passwords with SSO makes it easier for IT departments to audit where and how these user credentials are used. In the event that user credentials are compromised, IAM systems make it easier for IT departments to identify which user was compromised and which data was accessed during the breach. PingFederate allows you to monitor sign on performance metrics, traffic, and compliance centrally. Detailed audit trails allow systems to record user provisioning and de-provisioning as employees are on-boarded or terminated. OneLogin allows you to run detailed analytical reports on users, apps, logins other events.

4. Allows Easy Access No Matter Where You Are:

IAM/SSO allows users to access to all interconnected systems, regardless of where the user is physically located. This can be especially useful for large companies doing business globally, providing ease of access to employees, partners, and clients alike. OneLogin offers apps that allow users to access any enterprise web-based application anywhere on any device. OneLogin Mobile identity management provides users one-click access to all enterprise apps on smartphones and tablets.

5. Increases Productivity and Reduces IT Costs:

The original benefit of SSO for IT departments was to eliminate the cost of internal help desks helping users locked out of their application accounts. IAM is purporting to do much more. By leveraging already existing identity stores such as Active Director or LDAP, IAM allows you to extend what you have into the future. Cloud-based and mobile-based IAM tools not only allow users to authenticate from anywhere anytime, they also provide the extensive audit trails, analytics, access rules and policies to truly automate identity access and management across the enterprise.

Important Questions:

1. Draw the architecture and explain the importance of workflow management systems in cloud.
2. How inter-cloud resource management is helpful to the users. What services it provides.
3. how resource provisioning is helpful to the users. What are the various methods of resource provisioning?
4. Explain the Cloud Computing security architecture using suitable block diagram.
5. What are cloud security challenges? How is security provided to data at various stages in context of cloud?
6. Explain the global exchange of cloud resources using suitable block diagram.
7. Give the major security aspects of Virtual Machine Security.
8. Give the various issues in virtual machine security.
9. Explain IAM security standards. Give the various IAM security tools.
10. What do you understand by Security Governance? Give the Cloud Security Governance Challenges

UNIT 5

CLOUD TECHNOLOGIES AND ADVANCEMENTS

5.1 Hadoop:

Hadoop is an open-source software framework for storing data and running applications on clusters of commodity hardware. It provides massive storage for any kind of data, enormous processing power and the ability to handle virtually limitless concurrent tasks or jobs.

5.1.1 History of Hadoop:

- **In 2002,** Doug Cutting and Mike Cafarella started to work on a project, **Apache Nutch.** It is an open-source web crawler software project.

- ***While working on Apache Nutch, they were dealing with big data. To store that data, they have to spend a lot of costs which becomes the consequence of that project. This problem becomes one of the important reasons for the emergence of Hadoop.***

- **In 2003,** Google introduced a file system known as GFS (Google file system). It is a proprietary distributed file system developed to provide efficient access to data.
- **In 2004,** Google released a white paper on Map Reduce. This technique simplifies the data processing on large clusters.
- **In 2005,** Doug Cutting and Mike Cafarella introduced a new file system known as NDFS (Nutch Distributed File System). This file system also includes Map reduce.
- **In 2006,** Doug Cutting quit Google and joined Yahoo. On the basis of the Nutch project, Dough Cutting introduces a new project Hadoop with a file system known as HDFS (Hadoop Distributed File System). Hadoop first version 0.1.0 released in this year.Doug Cutting gave named his project Hadoop after his son's toy elephant.

- **In 2007,** Yahoo runs two clusters of 1000 machines.
- **In 2008,** Hadoop became the fastest system to sort 1 terabyte of data on a 900-node cluster within 209 seconds.
- **In 2013,** Hadoop 2.2 was released.
- **In 2017,** Hadoop 3.0 was released.
- **In 2018,** Apache Hadoop 3.1 version released.

5.1.2 Hadoop Architecture:

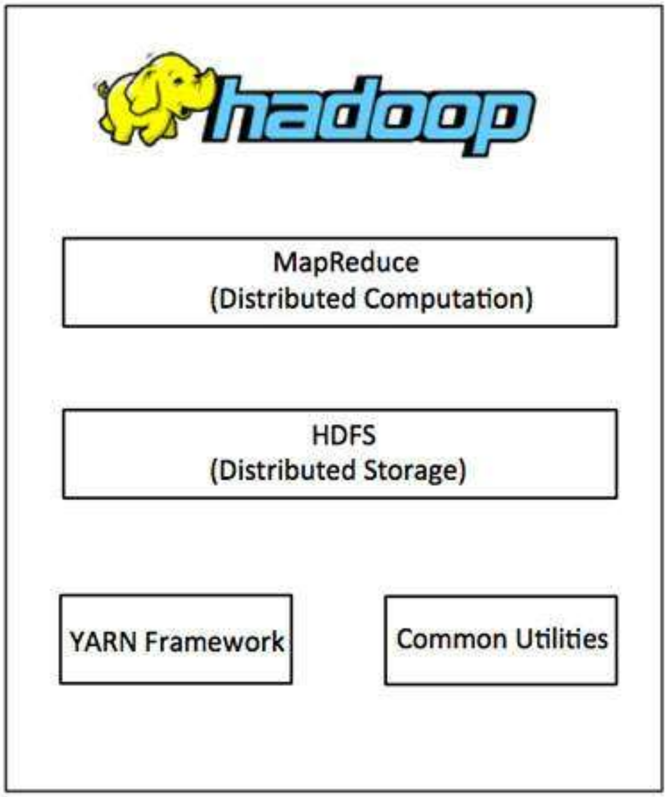


Fig 5.1 Hadoop Architecture

Hadoop architecture consists of following modules:

MapReduce

MapReduce is a parallel programming model for writing distributed applications devised at Google for efficient processing of large amounts of data (multi-terabyte datasets), on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner. The MapReduce program runs on Hadoop which is an Apache open-source framework.

Hadoop Distributed File System

The Hadoop Distributed File System (HDFS) is based on the Google File System (GFS) and provides a distributed file system that is designed to run on commodity hardware. It has many similarities with existing distributed file systems. However, the differences from other distributed file systems are significant. It is highly fault-tolerant and is designed to be deployed on low-cost hardware. It provides high throughput access to application data and is suitable for applications having large datasets.

Apart from the above-mentioned two core components, Hadoop framework also includes the following two modules −

- **Hadoop Common** − These are Java libraries and utilities required by other Hadoop modules.
- **Hadoop YARN** − This is a framework for job scheduling and cluster resource management.

5.1.3 Working of Hadoop:

- Data is initially divided into directories and files. Files are divided into uniform sized blocks of 128M and 64M (preferably 128M).
- These files are then distributed across various cluster nodes for further processing.
- HDFS, being on top of the local file system, supervises the processing.
- Blocks are replicated for handling hardware failure.
- Checking that the code was executed successfully.
- Performing the sort that takes place between the map and reduce stages.

- Sending the sorted data to a certain computer.
- Writing the debugging logs for each job.

5.1.4 Advantages of Hadoop

- Hadoop framework allows the user to quickly write and test distributed systems. It is efficient, and it automatic distributes the data and work across the machines and in turn, utilizes the underlying parallelism of the CPU cores.
- Hadoop does not rely on hardware to provide fault-tolerance and high availability (FTHA), rather Hadoop library itself has been designed to detect and handle failures at the application layer.
- Servers can be added or removed from the cluster dynamically and Hadoop continues to operate without interruption.
- Another big advantage of Hadoop is that apart from being open source, it is compatible on all the platforms since it is Java based.

5.2 MapReduce:

A MapReduce is a data processing tool which is used to process the data parallelly in a distributed form. It was developed in 2004, on the basis of paper titled as "MapReduce: Simplified Data Processing on Large Clusters," published by Google.

5.2.1 Why MapReduce?

Traditional Enterprise Systems normally have a centralized server to store and process data. The following illustration depicts a schematic view of a traditional enterprise system. Traditional model is certainly not suitable to process huge volumes of scalable data and cannot be accommodated by standard database servers. Moreover, the centralized system creates too much of a bottleneck while processing multiple files simultaneously.

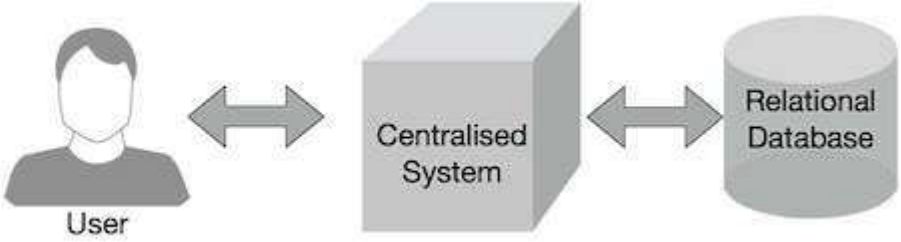


Fig 5.2 Traditional System

Google solved this bottleneck issue using an algorithm called MapReduce. MapReduce divides a task into small parts and assigns them to many computers. Later, the results are collected at one place and integrated to form the result dataset.

5.2.2 How MapReduce Works?

The MapReduce algorithm contains two important tasks, namely Map and Reduce.

- The Map task takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key-value pairs).
- The Reduce task takes the output from the Map as an input and combines those data tuples (key-value pairs) into a smaller set of tuples.

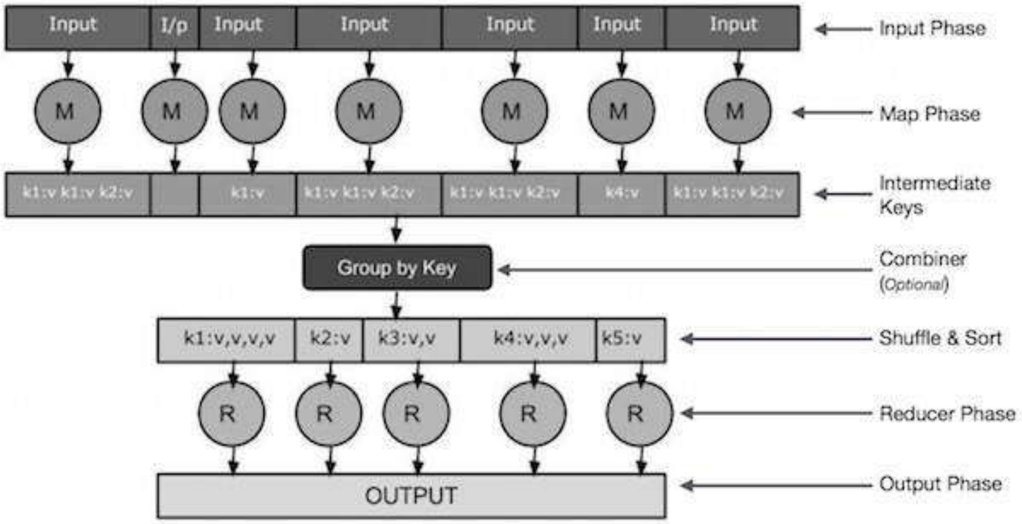


Fig 5.3 Working of MapReduce

- **Input Phase −** Here we have a Record Reader that translates each record in an input file and sends the parsed data to the mapper in the form of key-value pairs.
- **Map −** Map is a user-defined function, which takes a series of key-value pairs and processes each one of them to generate zero or more key-value pairs.
- **Intermediate Keys −** They key-value pairs generated by the mapper are known as intermediate keys.
- **Combiner −** A combiner is a type of local Reducer that groups similar data from the map phase into identifiable sets. It takes the intermediate keys from the mapper as input and applies a user-defined code to aggregate the values in a small scope of one mapper. It is not a part of the main MapReduce algorithm; it is optional.
- **Shuffle and Sort −** The Reducer task starts with the Shuffle and Sort step. It downloads the grouped key-value pairs onto the local machine, where the Reducer is running. The individual key-value pairs are sorted by key into a larger data list. The data list groups the equivalent keys together so that their values can be iterated easily in the Reducer task.
- **Reducer −** The Reducer takes the grouped key-value paired data as input and runs a Reducer function on each one of them. Here, the data can be aggregated, filtered, and combined in a number of ways, and it requires a wide range of processing. Once the execution is over, it gives zero or more key-value pairs to the final step.
- **Output Phase −** In the output phase, we have an output formatter that translates the final key-value pairs from the Reducer function and writes them onto a file using a record writer.

5.2.3 Example: wordcount example is given below:

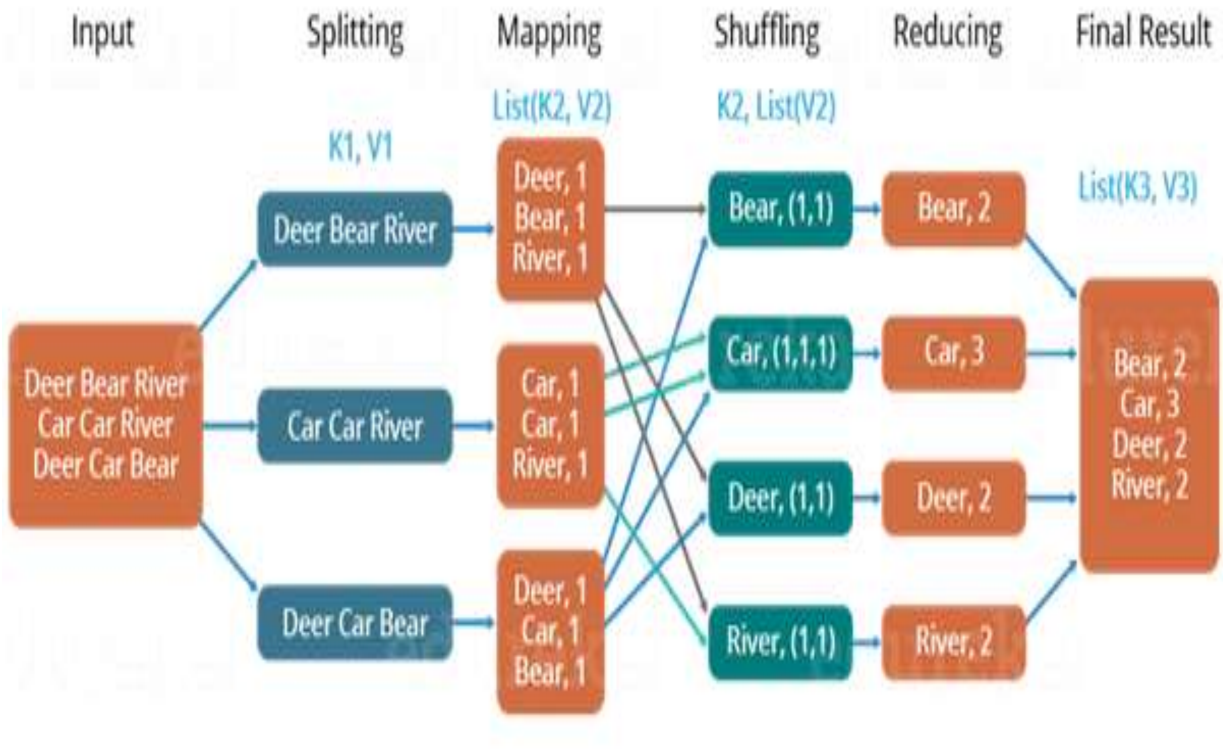


Fig 5.4 Wordcount example

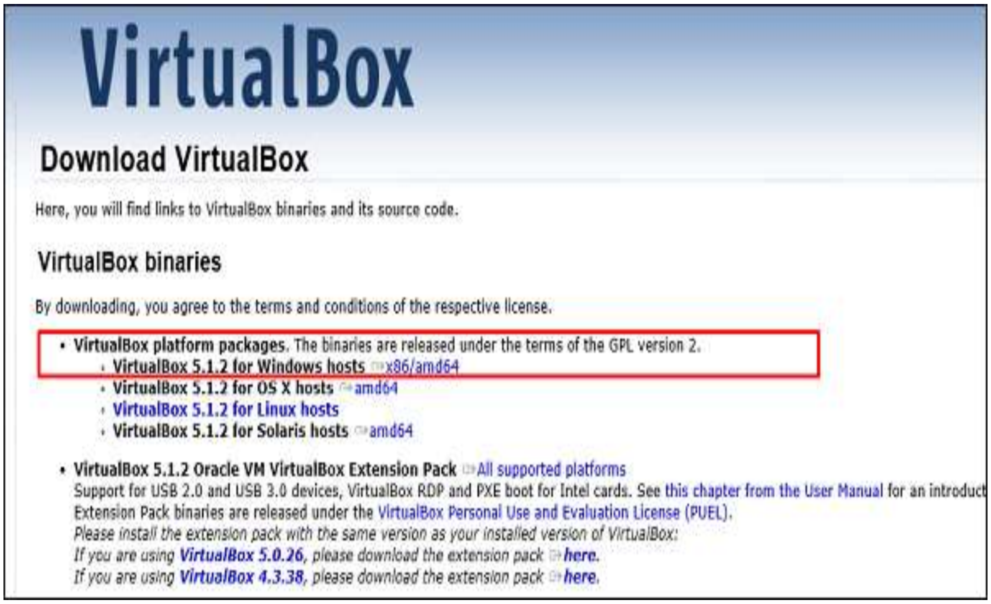
5.3 Virtual Box:

VirtualBox is a general-purpose virtualization tool for x86 and x86-64 hardware, targeted at server, desktop, and embedded **use**, that allows users and administrators to easily run multiple guest operating systems on a single host.

5.3.1 Installing VirtualBox

To start with, we will download VirtualBox and install it. We should follow the steps given below for the installation.

Step 1 − To download VirtualBox, click on the following link https://www.virtualbox.org/wiki/Downloads Now, depending on your OS, select which version to install. In our case, it will be the first one (Windows host).



VirtualBox

Download VirtualBox

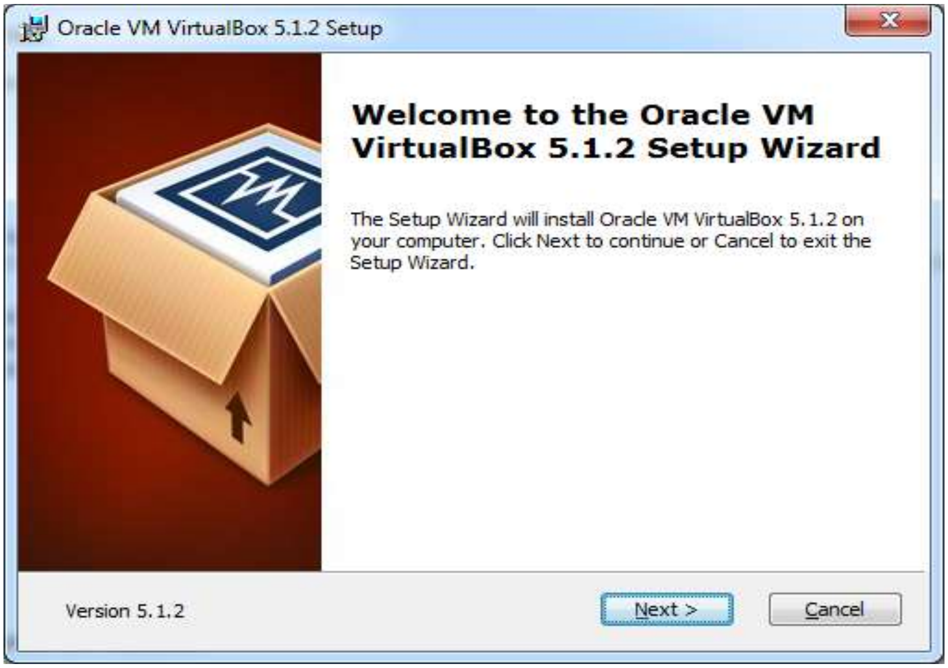
Here, you will find links to VirtualBox binaries and its source code.

VirtualBox binaries

By downloading, you agree to the terms and conditions of the respective license.

- VirtualBox platform packages. The binaries are released under the terms of the GPL version 2.
  - VirtualBox 5.1.2 for Windows hosts x86/amd64
  - VirtualBox 5.1.2 for OS X hosts amd64
  - VirtualBox 5.1.2 for Linux hosts
  - VirtualBox 5.1.2 for Solaris hosts amd64
- VirtualBox 5.1.2 Oracle VM VirtualBox Extension Pack All supported platforms
  Support for USB 2.0 and USB 3.0 devices, VirtualBox RDP and PXE boot for Intel cards. See this chapter from the User Manual for an introduct
  Extension Pack binaries are released under the VirtualBox Personal Use and Evaluation License (PUEL).
  Please install the extension pack with the same version as your installed version of VirtualBox:
  If you are using ***VirtualBox 5.0.26****, please download the extension pack* ***here.***
  If you are using ***VirtualBox 4.3.38****, please download the extension pack* ***here.***

Step 2 − Once the option is selected, click on “Next”.



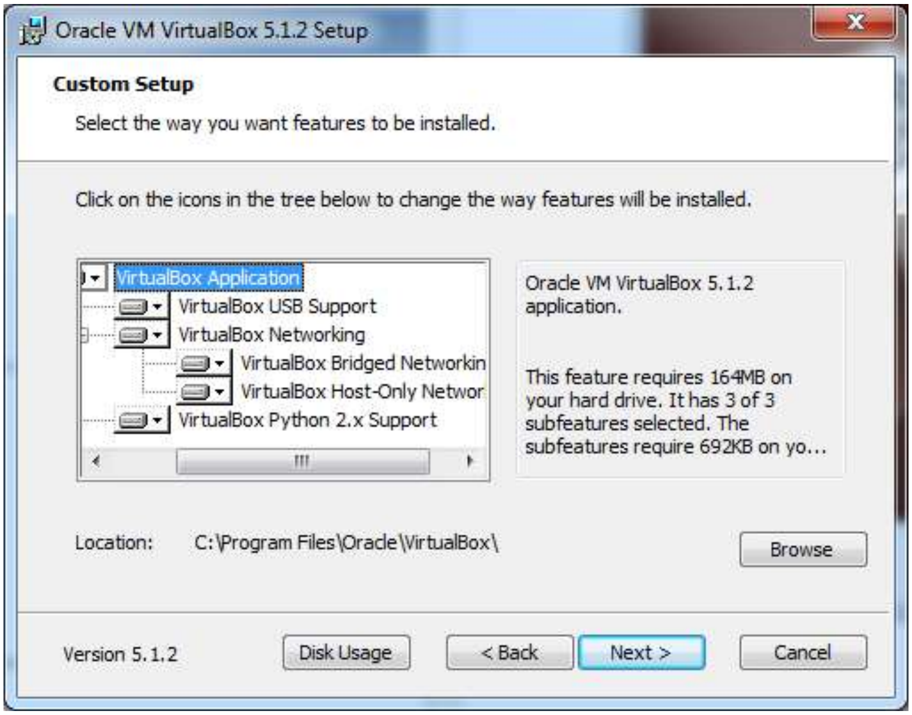
Oracle VM VirtualBox 5.1.2 Setup

Welcome to the Oracle VM VirtualBox 5.1.2 Setup Wizard

The Setup Wizard will install Oracle VM VirtualBox 5.1.2 on your computer. Click Next to continue or Cancel to exit the Setup Wizard.

Version 5.1.2

Step 3 − You have the option asking where to install the application. We can leave it as default and click on “Next”.



Oracle VM VirtualBox 5.1.2 Setup

Custom Setup

Select the way you want features to be installed.

Click on the icons in the tree below to change the way features will be installed.

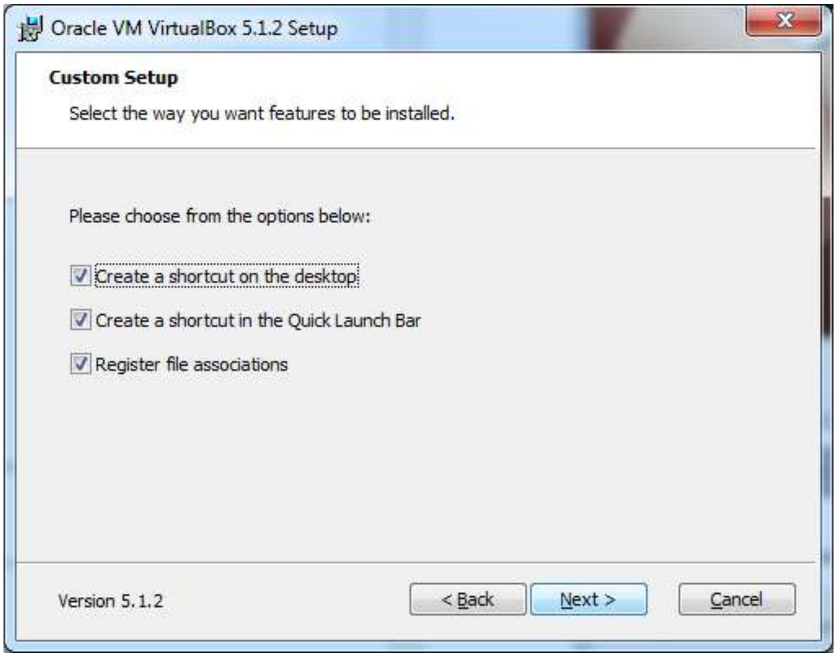
- VirtualBox Application
  - VirtualBox USB Support
  - VirtualBox Networking
    - VirtualBox Bridged Networkin
    - VirtualBox Host-Only Networ
  - VirtualBox Python 2.x Support

Oracle VM VirtualBox 5.1.2 application.

This feature requires 164MB on your hard drive. It has 3 of 3 subfeatures selected. The subfeatures require 692KB on yo...

Location: C:\Program Files\Oracle\VirtualBox\

Step 4 − Once the options are selected as shown in the following screenshot, click on Next.



Oracle VM VirtualBox 5.1.2 Setup

Custom Setup

Select the way you want features to be installed.

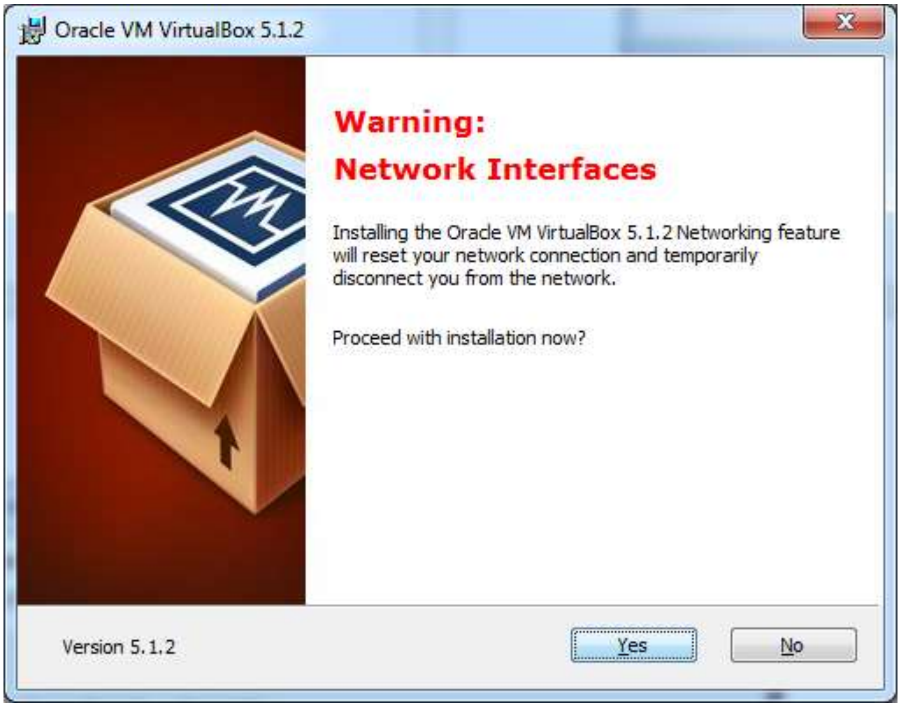
Please choose from the options below:

☑ Create a shortcut on the desktop

☑ Create a shortcut in the Quick Launch Bar

☑ Register file associations

Step 5 − A dialog box will come up asking whether to proceed with the installation. Click "Yes".



Oracle VM VirtualBox 5.1.2

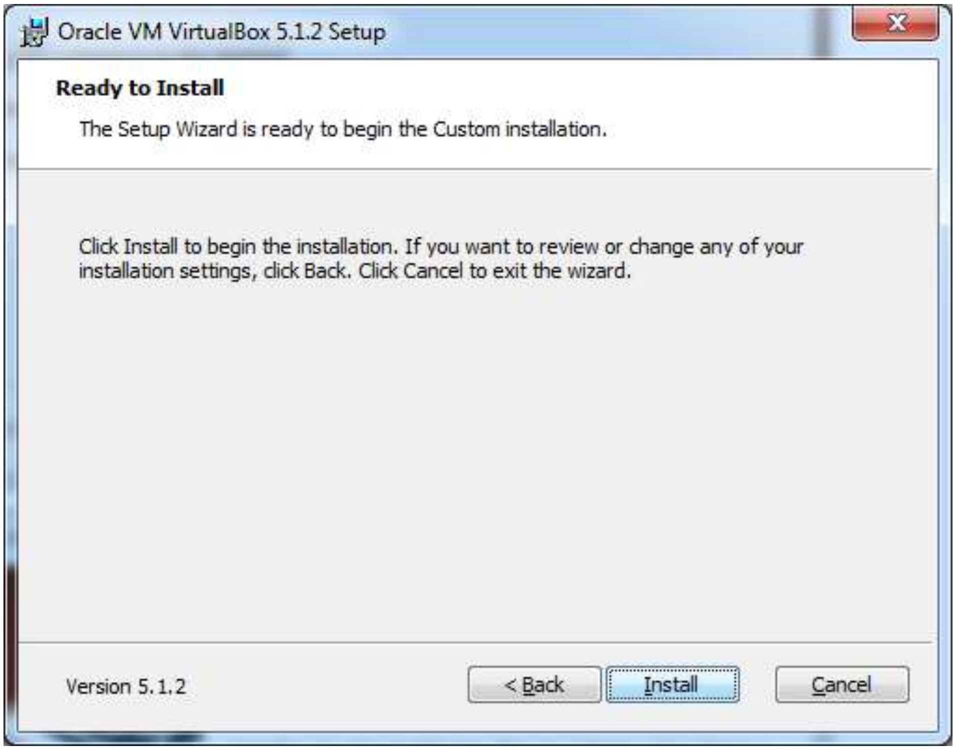
Warning:
Network Interfaces

Installing the Oracle VM VirtualBox 5.1.2 Networking feature will reset your network connection and temporarily disconnect you from the network.

Proceed with installation now?

Version 5.1.2

Step 6 − In the next step, click on "Install".



Oracle VM VirtualBox 5.1.2 Setup

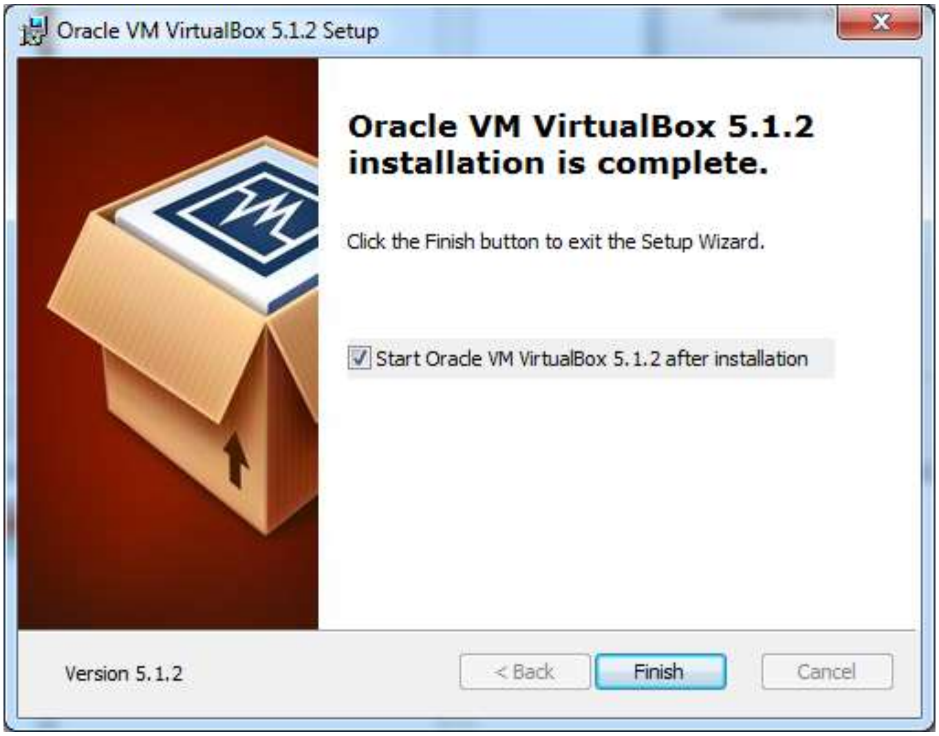
Ready to Install

The Setup Wizard is ready to begin the Custom installation.

Click Install to begin the installation. If you want to review or change any of your installation settings, click Back. Click Cancel to exit the wizard.

Version 5.1.2

Step 7 − Tick the start VirtualBox check box and click on “Finish”.



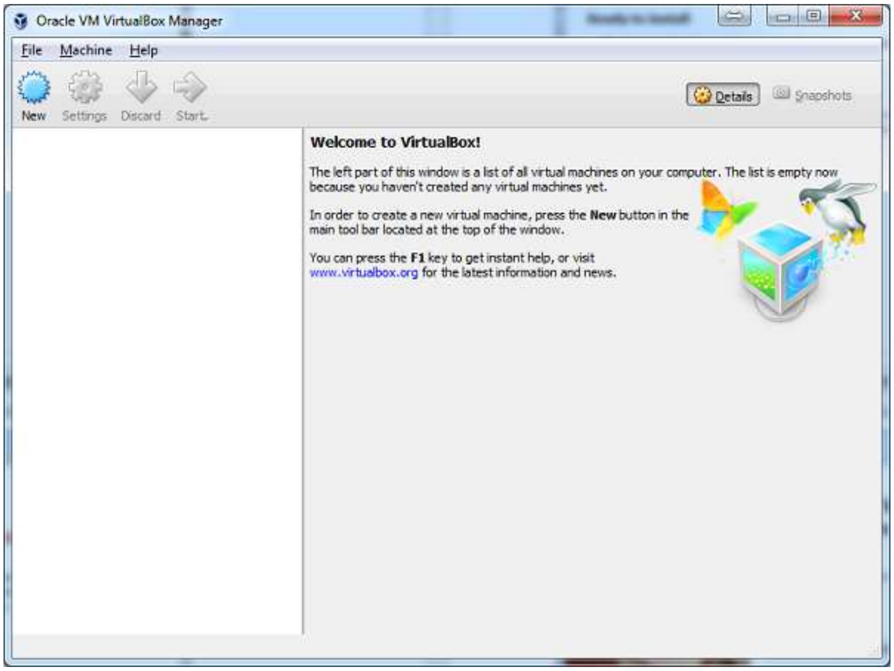
Oracle VM VirtualBox 5.1.2 installation is complete.

Click the Finish button to exit the Setup Wizard.

☑ Start Oracle VM VirtualBox 5.1.2 after installation

Version 5.1.2

Step 8 − VirtualBox application will now open as shown in the following screenshot. Now, we are ready to install the virtual machines.



Welcome to VirtualBox!

The left part of this window is a list of all virtual machines on your computer. The list is empty now because you haven't created any virtual machines yet.

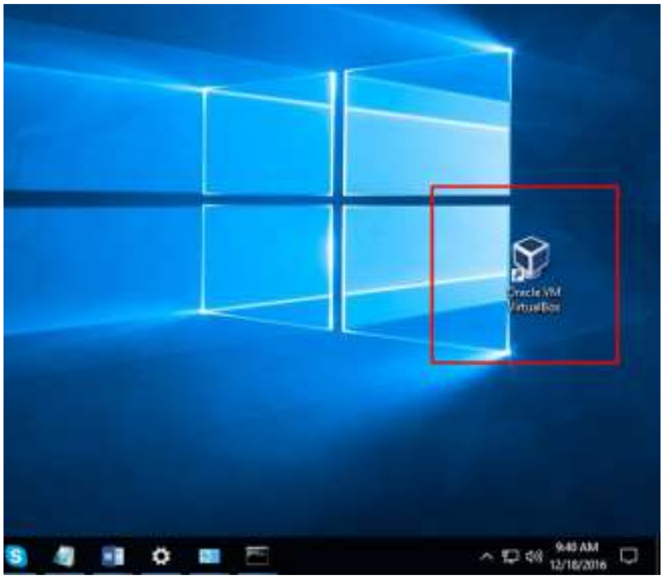
In order to create a new virtual machine, press the **New** button in the main tool bar located at the top of the window.

You can press the **F1** key to get instant help, or visit www.virtualbox.org for the latest information and news.

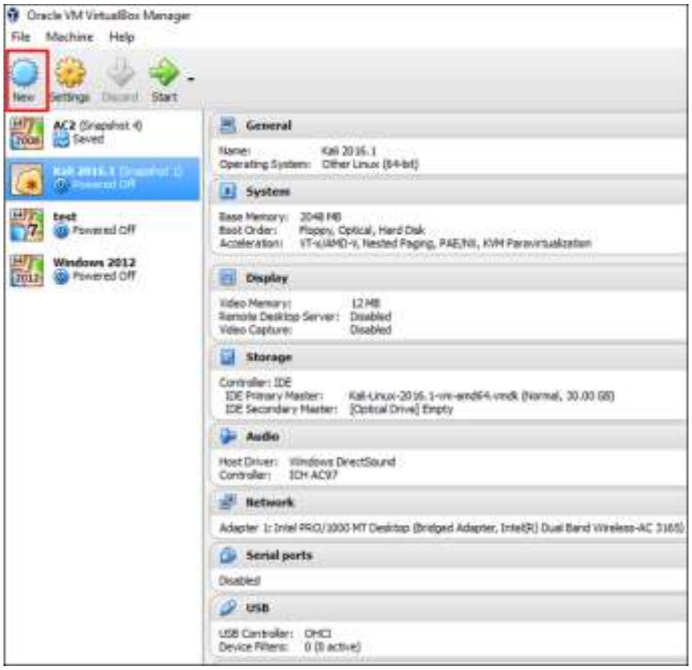
5.3.2 Creating a VM with VirtualBox

To create a virtual machine with Oracle VirtualBox, we should follow the steps given below.

Step 1 − To begin with, click on the “Oracle VM VirtualBox” icon on the desktop as shown in the screenshot below.



Step 2 − The next step is to click on “New” button, which is in the top left hand side of the screen.



General

Name: Kali 2016.1
Operating System: Other Linux (64-bit)

System

Base Memory: 2048 MB
Boot Order: Floppy, Optical, Hard Disk
Acceleration: VT-x/AMD-V, Nested Paging, PAE/NX, KVM Paravirtualization

Display

Video Memory: 12 MB
Remote Desktop Server: Disabled
Video Capture: Disabled

Storage

Controller: IDE
IDE Primary Master: Kali-Linux-2016.1-vm-amd64.vmdk (Normal, 30.00 GB)
IDE Secondary Master: [Optical Drive] Empty

Audio

Host Driver: Windows DirectSound
Controller: ICH AC97

Network

Adapter 1: Intel PRO/1000 MT Desktop (Bridged Adapter, Intel(R) Dual Band Wireless-AC 3165)

Serial ports

Disabled

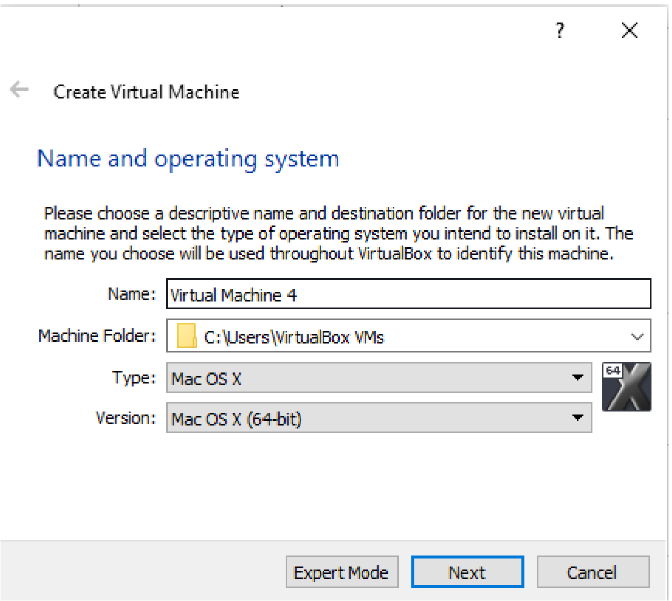
USB

USB Controller: OHCI
Device Filters: 0 (0 active)

Step 3 − A table will pop-up requesting you the parameters for the virtual machine. These will be −

- **Name** − We have to put a friendly name for this Virtual Machine.
- **Type** − Enter the OS that is going to be installed on it.
- **Version** − Enter the specific version for that OS, which we have selected earlier.

Once all the above parameters are filled, click on "Next".



Create Virtual Machine

Name and operating system

Please choose a descriptive name and destination folder for the new virtual machine and select the type of operating system you intend to install on it. The name you choose will be used throughout VirtualBox to identify this machine.

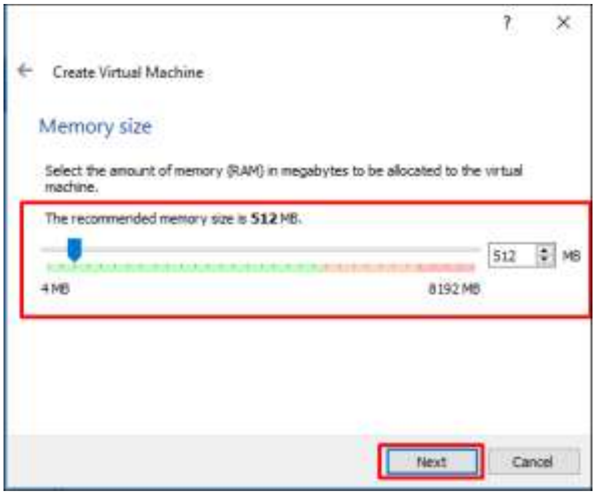
Name: Virtual Machine 4

Machine Folder: C:\Users\VirtualBox VMs

Type: Mac OS X

Version: Mac OS X (64-bit)

Step 4 − Select the amount of memory that you need to allocate in this VM → Click on "Next".



Create Virtual Machine

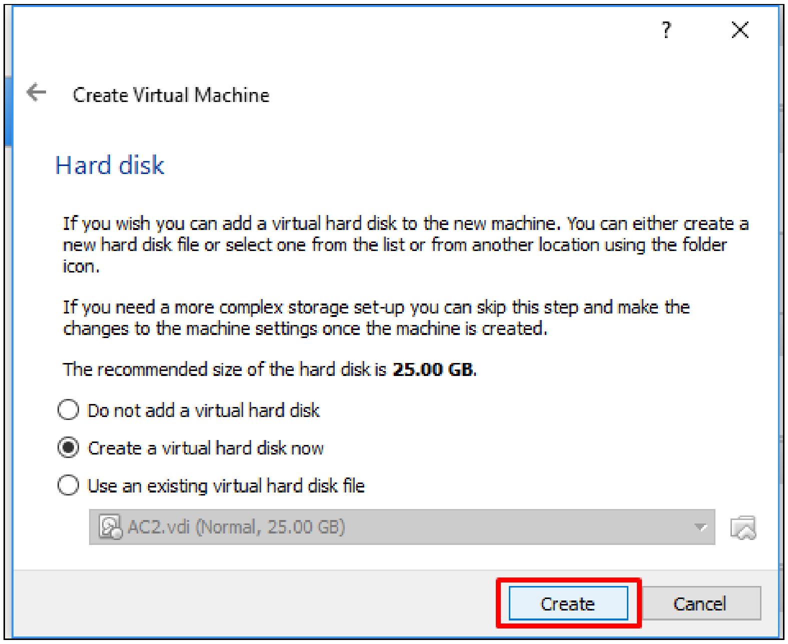
Memory size

Select the amount of memory (RAM) in megabytes to be allocated to the virtual machine.

The recommended memory size is **512** MB.

512 MB

Step 5 − Check one of the three options for the HDD and click on “Create”.



Create Virtual Machine

Hard disk

If you wish you can add a virtual hard disk to the new machine. You can either create a new hard disk file or select one from the list or from another location using the folder icon.

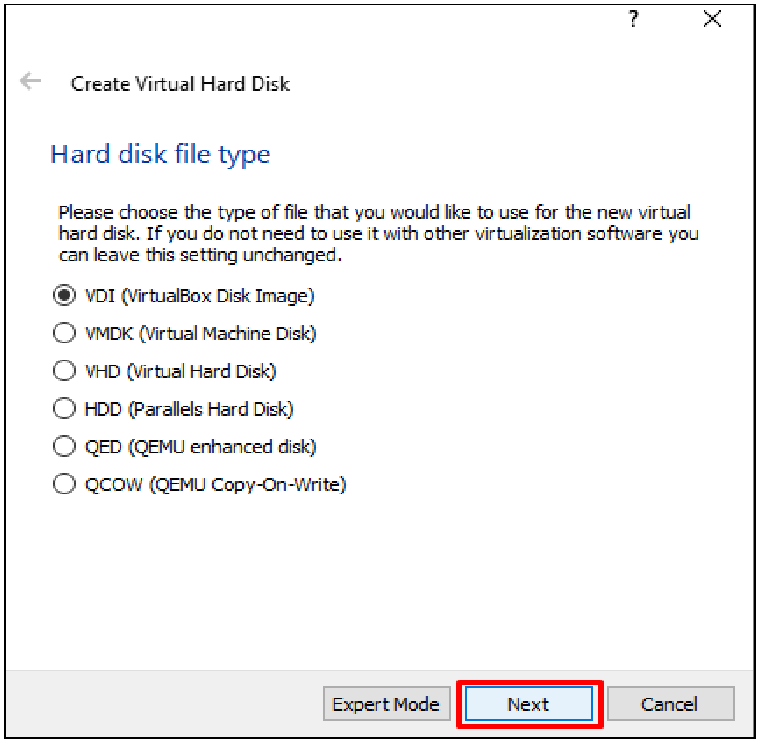
If you need a more complex storage set-up you can skip this step and make the changes to the machine settings once the machine is created.

The recommended size of the hard disk is **25.00 GB**.

○ Do not add a virtual hard disk
● Create a virtual hard disk now
○ Use an existing virtual hard disk file

AC2.vdi (Normal, 25.00 GB)

Step 6 − Select a file extension for your virtual HDD (It is recommended to use a common file extension that most of the hypervisors use like VHD) → click on “Next”.



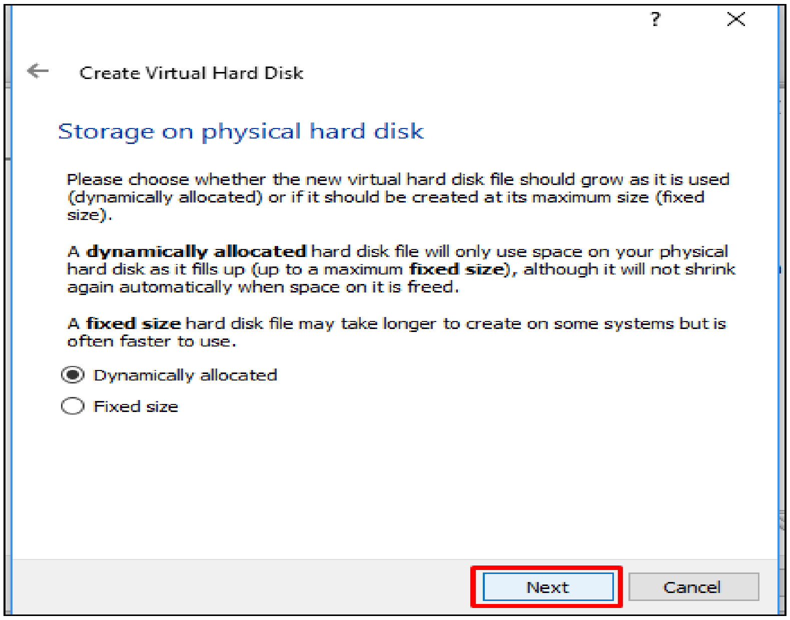
Create Virtual Hard Disk

Hard disk file type

Please choose the type of file that you would like to use for the new virtual hard disk. If you do not need to use it with other virtualization software you can leave this setting unchanged.

● VDI (VirtualBox Disk Image)
○ VMDK (Virtual Machine Disk)
○ VHD (Virtual Hard Disk)
○ HDD (Parallels Hard Disk)
○ QED (QEMU enhanced disk)
○ QCOW (QEMU Copy-On-Write)

Step 7 − Choose whether you want the Virtual HDD as dynamic or fixed. This is based on your needs → Click on “Next”.



Storage on physical hard disk

Please choose whether the new virtual hard disk file should grow as it is used (dynamically allocated) or if it should be created at its maximum size (fixed size).

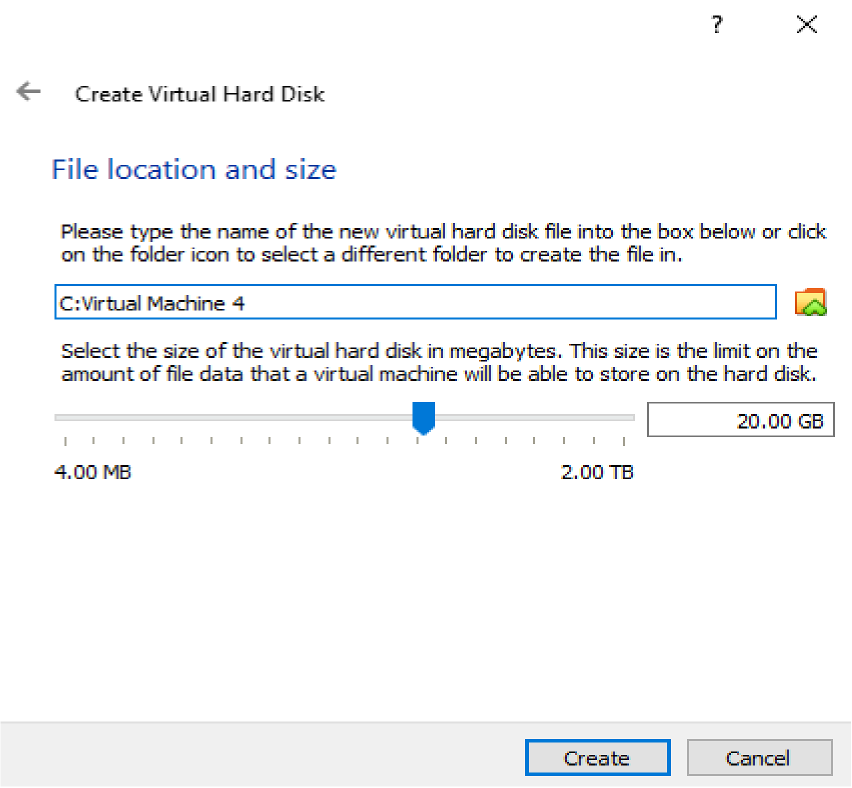
A **dynamically allocated** hard disk file will only use space on your physical hard disk as it fills up (up to a maximum **fixed size**), although it will not shrink again automatically when space on it is freed.

A **fixed size** hard disk file may take longer to create on some systems but is often faster to use.

● Dynamically allocated

○ Fixed size

Step 8 − Put a name for your virtual HDD file and select the disk size for your VM → Click on “Create”.



File location and size

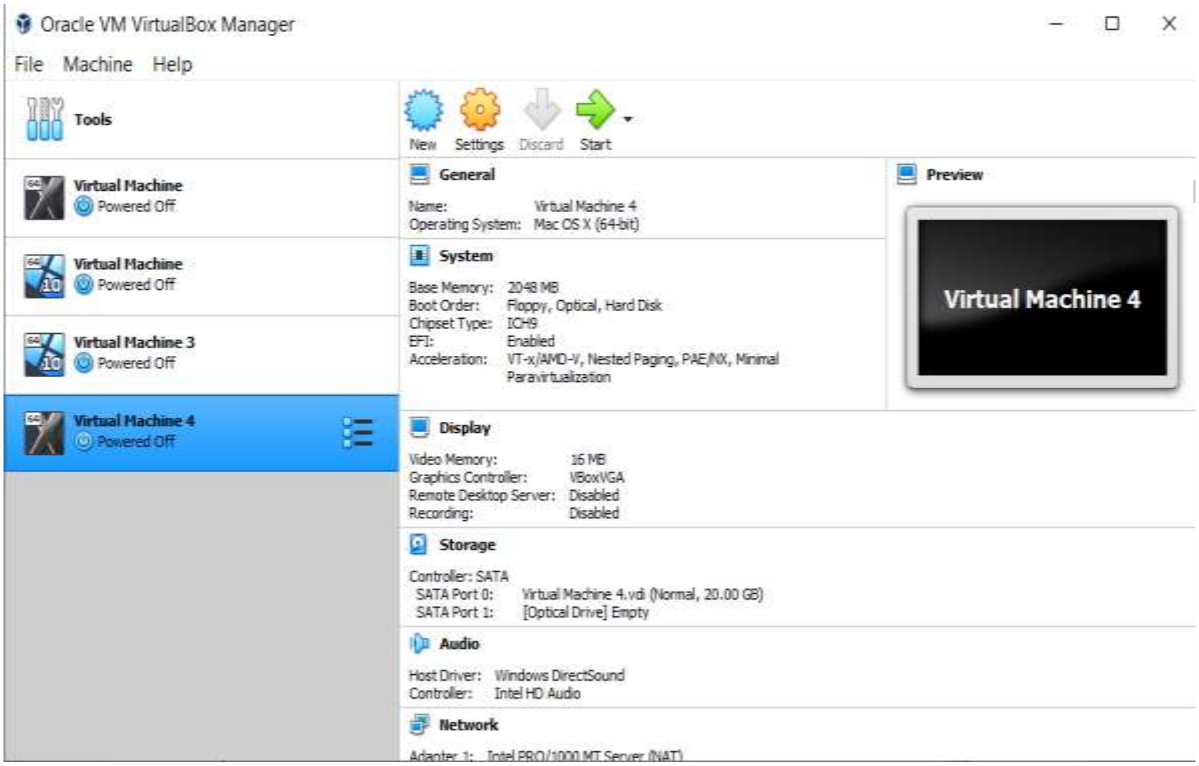
Please type the name of the new virtual hard disk file into the box below or click on the folder icon to select a different folder to create the file in.

C:Virtual Machine 4

Select the size of the virtual hard disk in megabytes. This size is the limit on the amount of file data that a virtual machine will be able to store on the hard disk.

20.00 GB

The virtual machine created will be as shown in the screenshot below.



General

Name: Virtual Machine 4
Operating System: Mac OS X (64-bit)

System

Base Memory: 2048 MB
Boot Order: Floppy, Optical, Hard Disk
Chipset Type: ICH9
EFI: Enabled
Acceleration: VT-x/AMD-V, Nested Paging, PAE/NX, Minimal Paravirtualization

Display

Video Memory: 16 MB
Graphics Controller: VBoxVGA
Remote Desktop Server: Disabled
Recording: Disabled

Storage

Controller: SATA
SATA Port 0: Virtual Machine 4.vdi (Normal, 20.00 GB)
SATA Port 1: [Optical Drive] Empty

Audio

Host Driver: Windows DirectSound
Controller: Intel HD Audio

Network

Adapter 1: Intel PRO/1000 MT Server (NAT)

5.4 Google App Engine (GAE):

- Google App Engine (GAE) is a service for developing and hosting Web applications in Google's data centers, belonging to the platform as a service (PaaS) category of cloud computing.
- It is Google's platform as a service offering that allows developers and businesses to build and run applications using Google's advanced infrastructure.
- Web applications hosted on GAE are sandboxed and run across multiple servers for redundancy and allowing for scaling of resources according to the traffic requirements of the moment. App Engine automatically allocates additional resources to the servers to accommodate increased load.

- These applications are required to be written in one of a few supported languages, namely: Java, Python, PHP and Go. It also requires the use of Google query language and that the database used is Google Big Table. Applications must abide by these standards, so applications either must be developed with GAE in mind or else modified to meet the requirements.

5.4.1 What is Google App Engine used for?

- **App Engine** is a fully managed, serverless platform for developing and hosting web applications at scale. You can choose from several popular languages, libraries, and frameworks to develop your **apps**, then let **App Engine** take care of provisioning servers and scaling your **app** instances based on demand.

5.4.2 Features of Google App Engine

1. Collection of Development Languages and Tools

The App Engine supports numerous programming languages for developers and offers the flexibility to import libraries and frameworks through docker containers. You can develop and test an app locally using the SDK containing tools for deploying apps. Every language has its SDK and runtime. Some of the languages offered include — Python, PHP, .NET, Java, Ruby, C#, Go, Node.Js.

2. Fully Managed

Google allows you to add your web application code to the platform while managing the infrastructure for you. The engine ensures that your web apps are secure and running and saves them from malware and threats by enabling the firewall.

3. Pay-as-you-Go

The app engine works on a pay-as-you-go model, i.e., you only pay for what you use. The app engine automatically scales up resources when the application traffic picks up and vice-versa.

4. Effective Diagnostic Services

Cloud Monitoring and Cloud Logging that helps run app scans to identify bugs. The app reporting document helps developers fix bugs on an immediate basis.

5. Traffic Splitting

The app engine automatically routes the incoming traffic to different versions of the apps as a part of A/B testing. You can plan the consecutive increments based on what version of the app works best.

5.4.3 Advantages of GAE include:

- Readily available servers with no configuration requirement
- Power scaling function all the way down to "free" when resource usage is minimal
- Automated cloud computing tools

5.4.4 Google Cloud vs AWS vs Azure:

Table 5.1 Google Cloud vs AWS vs Azure

Google Cloud	AWS	Azure
It uses GCE (Google Compute Engine) for computing purposes.	AWS EC2 offers core compute services.	It uses virtual machines for computation purposes.
It uses Google Cloud Storage for storage purposes.	It uses Amazon S3 for storing the data.	It uses a storage block bob that comprises blocks for storing the data.
It offers the lowest price to the customers to beat other cloud providers.	AWS pricing is generally keen to have inscrutable. The overall structure of granular pricing is a	Like AWS, Azure pricing structure is also difficult to understand unless you have

	bit complex.	considerable experience.
It uses Cloud Test labs for App Testing purposes.	It uses a device farm for App Testing purposes.	It uses DevTest labs for App Testing purposes.
It uses Subnet as a virtual network.	It uses VPC as a virtual network.	It uses VNet as a virtual Network.
It follows the Cloud Load Balancing configuration.	It follows the Elastic Load Balancing configuration.	It follows the Load-Balancer Application Gateway configuration.

5.5 Programming Environment for Google App Engine:

GAE Deployment using Travis CI

Travis CI is a hosted continuous integration service used to build and test software projects hosted at GitHub and Bitbucket. Travis CI was the first CI service which provided services to open-source projects for free.

Travis CI can automatically deploy your Google App Engine or Managed VMs application after a successful build.

For a minimal configuration, add the following to your .travis.yml:

```
deploy:
  provider: gae
  keyfile: "YOUR SERVICE ACCOUNT JSON FILE"
```

```
        project: "YOUR PROJECT ID"
```

YAML

Then go to the Google Cloud Console Dashboard and:

1. Enable “Google App Engine Admin API”,
2. Go to “Credentials”, click “Add Credential” and “Service account key”, finally click “JSON” to download your Service Account JSON file.

It is *strongly* recommended that you encrypt your key before committing it to a repo. First make sure you have the Travis command line tool installed. Then execute the following command from the terminal:

```
travis encrypt-file client-secret.json --add
```

Bash

The --add flag automatically adds the decryption step to the. travis file.

Keep in mind that the above command has to run in your project directory, so it can modify the.travis.yml for you.

More detailed instructions for encrypting keys using Travis can be found here.

Project to deploy

By default, the project will be deployed with the same name as the repository. Usually, you will want to explicilty configure the **project** option to match the project ID found in your Cloud console (note that this is sometimes, but not always, the same as the project name).

You can explicitly set the project id via the **project** option:

```
deploy:
        provider: gae
        keyfile: ...
  project: continuous-deployment-demo deploy:
  provider: gae
  keyfile: ...
  project: continuous-deployment-demo
  no_stop_previous_version: true
```

YAML

YAML

Version to deploy #

Either the **version** flag or the **default** option must be set. If default is true, the default version will be deployed to, which will be http://your-project-id.appspot.com. If the **version** flag is set instead, it will deploy to http://version-dot-your-project-id.appspot.com.

Branch to deploy from #

By default, Travis will only deploy from your **master** branch.

You can also explicitly specify the branch to deploy from with the **on** option:

```
deploy:
  provider: gae
  keyfile: ...
  project: ...
  on: production
```

YAML

Alternatively, you can also configure it to deploy from all branches:

```
deploy:
  provider: gae
  keyfile: ...
  project: ...
  on:
    all_branches: true
```

YAML

Builds triggered from Pull Requests will never trigger a deploy.

Deploying without Promoting

By default, when your application is deployed it will be promoted to receive all traffic. You can disable that using the no_promote option:

```
deploy:
  provider: gae
  keyfile: ...
  project: continuous-deployment-demo
  no_promote: true
```

YAML

In addition to that, and according to the Google Cloud SDK changelog, *“in a future Cloud SDK release, deployments that promote the new version to receive all traffic will stop the previous version by default”*.

You can prevent that from happening by setting the option no_stop_previous_version:

Skipping Cleanup #

Many App Engine apps use pip to vendor library requirements into the directory, and sometimes you need build artifacts or other credentials to deploy. If so, you want to avoid the Travis cleanup step that will clean you working directory before the deploy.

```
deploy:
  provider: gae
  skip_cleanup: true
```

5.5 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.

A dashboard is also available, giving administrators control while empowering their users to provision resources through a web interface.

Beyond standard infrastructure-as-a-service functionality, additional components provide orchestration, fault management and service management amongst other services to ensure high availability of user applications.

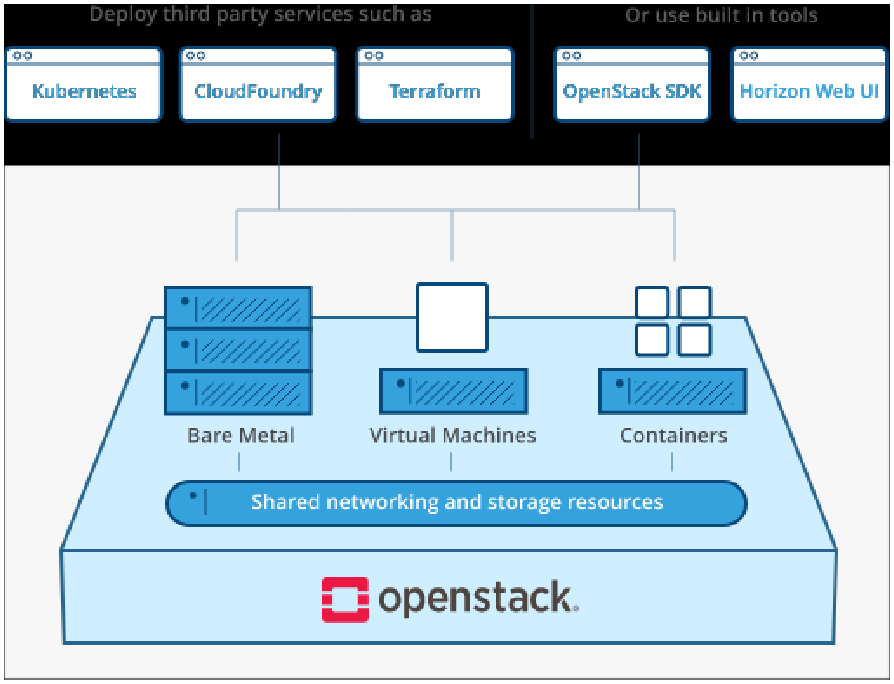


Fig 5.5 OpenStack Architectue

5.5.1 How is OpenStack used in a cloud environment?

The cloud is all about providing computing for end users in a remote environment, where the actual software runs as a service on reliable and scalable servers rather than on each end-user's computer. Cloud computing can refer to a lot of different things, but typically the industry talks about running different items "as a service"—software, platforms, and infrastructure. OpenStack falls into the latter category and is considered Infrastructure as a Service (IaaS). Providing infrastructure means that OpenStack makes it easy for users to quickly add new instance, upon which other cloud components can run. Typically, the infrastructure then runs a "platform" upon which a developer can create software applications that are delivered to the end users.

5.5.2 What are the components of OpenStack?

OpenStack is made up of many different moving parts. Because of its open nature, anyone can add additional components to OpenStack to help it to meet their needs. But the OpenStack community has collaboratively identified nine key components that are a part of the "core" of OpenStack, which are distributed as a part of any OpenStack system and officially maintained by the OpenStack community.

Nova is the primary computing engine behind OpenStack. It is used for deploying and managing large numbers of virtual machines and other instances to handle computing tasks.

Swift is a storage system for objects and files. Rather than the traditional idea of a referring to files by their location on a disk drive, developers can instead refer to a unique identifier referring to the file or piece of information and let OpenStack decide where to store this information. This makes scaling easy, as developers don't have the worry about the capacity on a single system behind the software. It also allows the system, rather than the developer, to worry about how best to make sure that data is backed up in case of the failure of a machine or network connection.

Cinder is a block storage component, which is more analogous to the traditional notion of a computer being able to access specific locations on a disk drive. This more traditional way of accessing files might be important in scenarios in which data access speed is the most important consideration.

Neutron provides the networking capability for OpenStack. It helps to ensure that each of the components of an OpenStack deployment can communicate with one another quickly and efficiently.

Horizon is the dashboard behind OpenStack. It is the only graphical interface to OpenStack, so for users wanting to give OpenStack a try, this may be the first component they actually "see." Developers can access all of the components of OpenStack individually through an application programming interface (API), but the dashboard provides system administrators a look at what is going on in the cloud, and to manage it as needed.

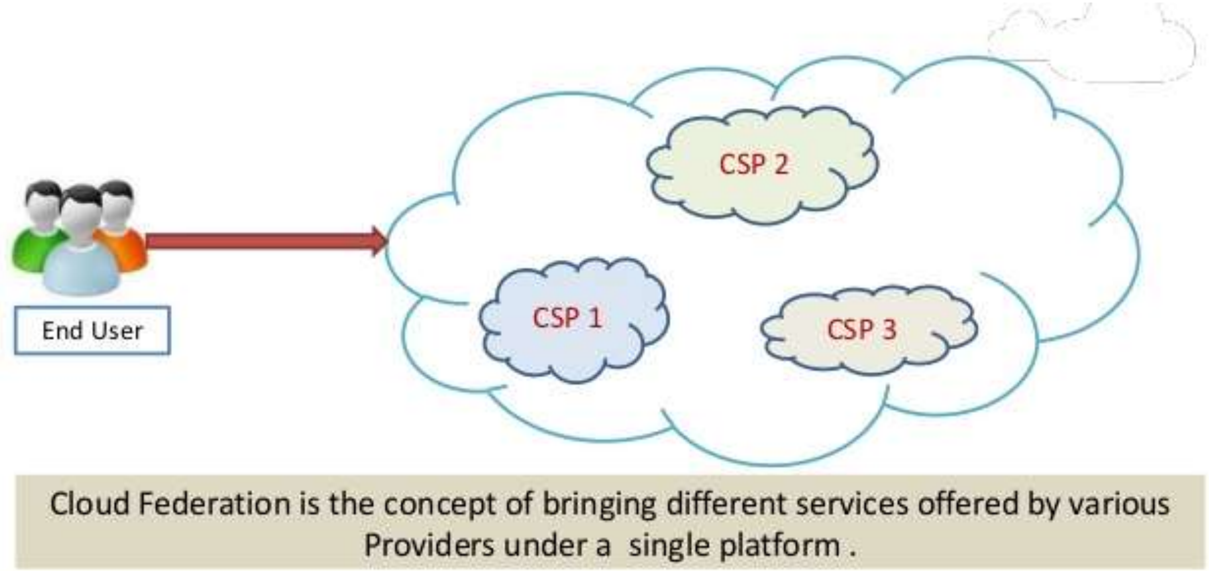
Keystone provides identity services for OpenStack. It is essentially a central list of all of the users of the OpenStack cloud, mapped against all of the services provided by the cloud, which they have permission to use. It provides multiple means of access, meaning developers can easily map their existing user access methods against Keystone.

Glance provides image services to OpenStack. In this case, "images" refers to images (or virtual copies) of hard disks. Glance allows these images to be used as templates when deploying new virtual machine instances.

Ceilometer provides telemetry services, which allow the cloud to provide billing services to individual users of the cloud. It also keeps a verifiable count of each user's system usage of each of the various components of an OpenStack cloud. Think metering and usage reporting.

Heat is the orchestration component of OpenStack, which allows developers to store the requirements of a cloud application in a file that defines what resources are necessary for that application. In this way, it helps to manage the infrastructure needed for a cloud service to run.

5.6 Federation in the Cloud:



Cloud Federation is the concept of bringing different services offered by various Providers under a single platform .

Fig 5.6 Federation in the Cloud

Cloud Federation refers to the unionization of software, infrastructure and platform services from disparate networks that can be accessed by a client via the internet. ... It is important to note that federated **cloud** computing services still rely on the existence of physical data centers.

Cloud federation requires one provider to wholesale or rent computing resources to another cloud provider. Those resources become a temporary or permanent extension of the buyer's cloud computing environment, depending on the specific federation agreement between providers.

Cloud federation offers two substantial benefits to cloud providers. First, it allows providers to earn revenue from computing resources that would otherwise be idle or underutilized. Second, cloud federation enables cloud providers to expand their geographic footprints and accommodate sudden spikes in demand without having to build new points-of-presence (POPs).

5.6.1 Driving Factors of Cloud-Federation:

- Choice of providers
- Mix and Match of Services
- Going Local
- Heterogeneity of Platform
- Ability to Scale
- Movement and Migration of Workloads

5.6.2 Architecture of Cloud Federation:

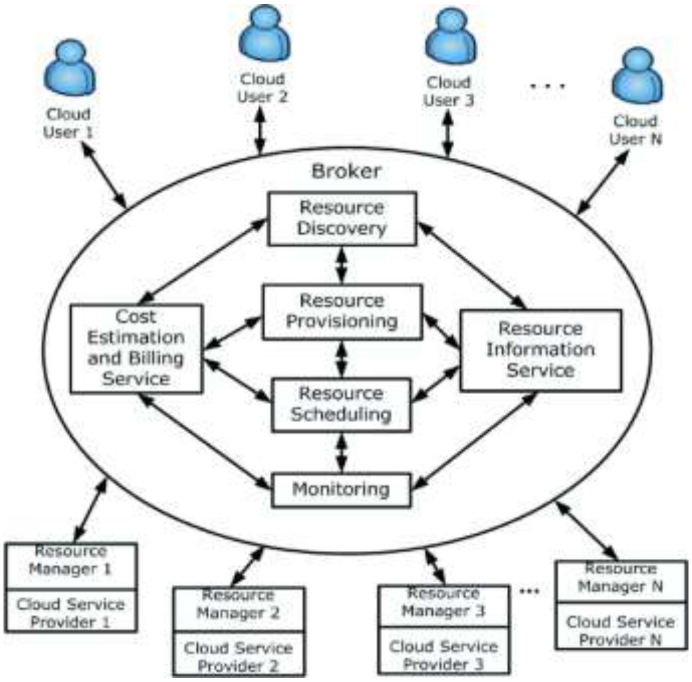


Fig 5.7 **Architecture of Cloud Federation**

A basic architecture for a Cloud Federation contains three main modules:

Fronted module:

The front-end module provides users with a single interface to choose their preferred service and access Cloud resources

Resource Broker module:

A resource broker with its main function being to match. the available resources to the user's requests. The use of. the resource broker provides a uniform interface to access. any of the available and appropriate resources using.

Cloud Interface module:

A cloud interface serves as a gateway or interface that provides direct and indirect cloud infrastructure and software services to users. A cloud interface is the core component behind any public cloud solution and is generally based primarily on the REST and SOAP frameworks, as well as cross-platform and vendor specific APIs.

5.7 Levels of Federation:

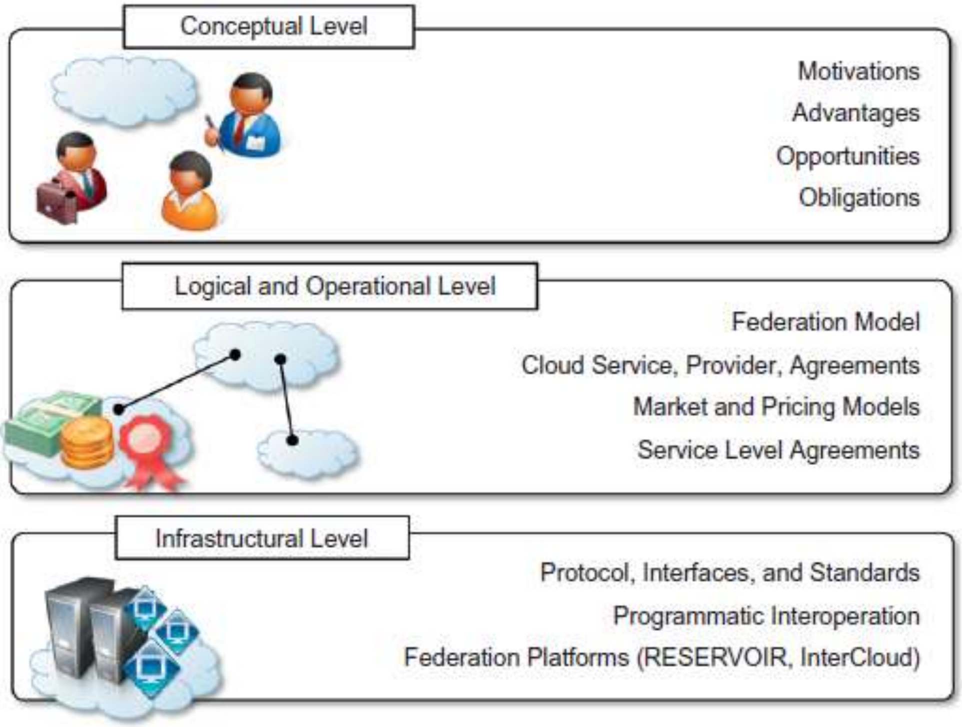

Conceptual Level
Motivations
Advantages
Opportunities
Obligations
Logical and Operational Level
Federation Model
Cloud Service, Provider, Agreements
Market and Pricing Models
Service Level Agreements
Infrastructural Level
Protocol, Interfaces, and Standards
Programmatic Interoperation
Federation Platforms (RESERVOIR, InterCloud)

Fig 5.8 Levels of Federation

Each cloud federation level presents different challenges and operates at a different layer of the IT stack. It then requires the use of different approaches and technologies. Taken together, the solutions to the challenges faced at each of these levels constitute a reference model for a cloud federation.

Conceptual Level:

The conceptual level addresses the challenges in presenting a cloud federation as a favourable solution with respect to the use of services leased by single cloud providers. In this level it is important to clearly identify the advantages for either service providers or service consumers in joining a federation and to delineate the new opportunities that a federated environment creates with respect to the single-provider solution.

Elements of concern at this level are:

- Motivations for cloud providers to join a federation.
- Motivations for service consumers to leverage a federation.
- Advantages for providers in leasing their services to other providers.
- Obligations of providers once they have joined the federation.
- Trust agreements between providers.
- Transparency versus consumers.

.

Logical & Operational Level:

The logical and operational level of a federated cloud identifies and addresses the challenges in devising a framework that enables the aggregation of providers that belong to different administrative domains within a context of a single overlay infrastructure, which is the cloud federation.

At this level, policies and rules for interoperation are defined. Moreover, this is the layer at which decisions are made as to how and when to lease a service to—or to leverage a service from— another provider.

The logical component defines a context in which agreements among providers are settled and services are negotiated, whereas the operational component characterizes and shapes the dynamic behaviour of the federation as a result of the single providers' choices.

This is the level where MOCC is implemented and realized. It is important at this level to address the following challenges:

- How should a federation be represented?
- How should we model and represent a cloud service, a cloud provider, or an agreement?
- How should we define the rules and policies that allow providers to join a federation?
- What are the mechanisms in place for settling agreements among providers?
- What are provider's responsibilities with respect to each other?
- When should providers and consumers take advantage of the federation?

• Which kinds of services are more likely to be leased or bought?

• How should we price resources that are leased, and which fraction of resources should we lease? The logical and operational level provides opportunities for both academia and industry.

Infrastructure Level:

The infrastructural level addresses the technical challenges involved in enabling heterogeneous cloud computing systems to interoperate seamlessly.

It deals with the technology barriers that keep separate cloud computing systems belonging to different administrative domains. By having standardized protocols and interfaces, these barriers can be overcome.

At this level it is important to address the following issues:

- What kind of standards should be used?
- How should design interfaces and protocols be designed for interoperation?
- Which are the technologies to use for interoperation?
- How can we realize a software system, design platform components, and services enabling interoperability?

5.8 Federated Services and Applications:

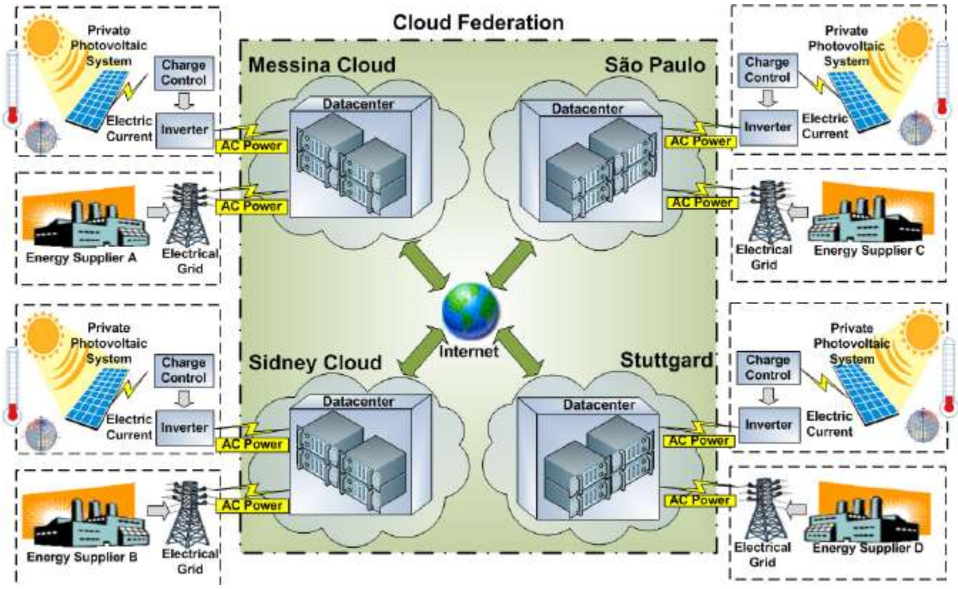


Fig 5.9 Federated Services and Applications

Above diagram shows the scenario where federated cloud is being used for providing services to the users.

It shows how different services are being collaborated and providing their services to cloud users.

5.9 Future of Cloud Federation:

- Additive benefits can be provided to the users by using Cloud-Federation.
- Risk can be minimized by using federated cloud as it will follow the security solutions provided by each service providers.
- Integrity could be improved.

Important Questions:

1. Define the term Hadoop. How it is implemented using map-reduce techniques**.**
2. Explain virtual box with example. Give the various steps to install virtual box.
3. Define Google App Engine (GAE). Give the various feature of google app engine.
4. What is open stack? How is OpenStack used in a cloud environment?
5. Draw and explain the block diagram of Cloud Federation:
6. Give the various services of cloud federation.
7. Explain the various levels of federation in cloud computing.
8. Give the future of cloud federation.