

**191CS61**

**CLOUD COMPUTING**

**L T P C**  
**3 0 0 3**

**Programme:** B.E. Computer Science and Engineering

**Sem: 6 Category: PC**

**Prerequisites:** 191CS54 – Computer Networks

**Aim:** To provide an in-depth and comprehensive knowledge of the Cloud Computing fundamental issues, technologies, applications and implementations.

**Course Outcomes:** The Students will be able to

**CO1:** Recall the concepts, characteristics, delivery models and benefits of cloud computing

**CO2:** Apply and design suitable virtualization concepts.

**CO3:** Identify the architecture and infrastructure of cloud computing, including SaaS, PaaS, IaaS, public cloud, private cloud, hybrid cloud, etc.

**CO4:** Provide the appropriate cloud computing solutions and recommendations according to the applications used.

**CO5:** Explain the core issues of cloud computing such as security, privacy, and interoperability

**CO6:** Test programming and experiment with the various cloud computing environments

**INTRODUCTION TO CLOUD COMPUTING** **9**

Distributed Computing-Cluster computing- Grid computing-Characteristics of Cloud Computing-Cloud deployment models-Cloud service models-Cloud services-Cloud Applications.

**CONCEPTS & TECHNOLOGIES** **9**

Virtualization-Load balancing-Scalability&Elasticity-Deployment-Replication-Monitoring-MapReduce-Identify and Access Management-Service Level Agreements-Billing.

**CLOUD SERVICES & PLATFORMS** **9**

Compute services-Storage services-Database services-Application services-Content delivery services-Analytics services-Deployment& Management services, Identify and Access Management

**CLOUD APPLICATION DESIGN & BENCHMARKING** **9**

Cloud application design consideration-Cloud application reference architectures-Design methodologies-Data Storage-Data analytics-Deployment & Management-Performance metrics for cloud applications-Cloud application testing

**CLOUD SECURITY** **9**

Cloud Security challenges –Authentication-Authorization-identify& Access Management-Data Security-Data Integrity-Encryption& Key Management

**Total Periods: 45**

**Text Book:**

1. Cloud Computing: “A Hands-On Approach” ArshdeepBahga and Vijay K. Madiseti, published 2013.

**References:**

1. Kai Hwang, Geoffrey C. Fox and Jack J. Dongarra, “Distributed and Cloud Computing from Parallel Processing to the Internet of Things”, Morgan Kaufmann, Elsevier, 2012
2. Barrie Sosinsky, “Cloud Computing Bible” John Wiley & Sons, Wiley publishing, Inc. 2011
3. Tim Mather, Subra Kumaraswamy, and Shahed Latif, “Cloud Security and Privacy Enterprise Perspective on Risks and Compliance”, O'Reilly 2009
4. Bernard Golden, “Amazon Web Services for Dummies”, John Wiley & Sons, 2013.

Course Outcomes	Program Outcomes (POs)												Program Specific Outcomes (PSOs)			
	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3	PSO4
CO1	3													3		
CO2	3	2	3	3									3	2		
CO3	3	2	2	2												
CO4	3	3	3	3							2		2	2		
CO5	3	3	2										3	2	3	3
CO6	2	2	3	3	2							3	3	2	3	3

1: Slight (Low) 2: Moderate (Medium) 3: Substantial (High)

# UNIT I

---

INTRODUCTION TO CLOUD COMPUTING

# Distributed Computing

---

- ❖ Distributed computing refers to the use of multiple computers or servers to work together on solving a problem or performing a task.
- ❖ In a distributed computing system, the processing and storage resources are spread across multiple machines, and they communicate with each other to achieve a common goal.
- ❖ This approach offers several advantages
  - Improved performance
  - Fault tolerance
  - Scalability

## **Here are some key concepts and components associated with distributed computing:**

### **Nodes/Computers:**

- Nodes are individual computers or servers that participate in the distributed system.
- Each node typically has its own processing power, memory, and storage.

### **Communication:**

---

- Nodes in a distributed system communicate with each other through a network. This communication can occur via message passing, remote procedure calls (RPC), or other communication protocols.
- Efficient and reliable communication is essential for the proper functioning of distributed systems.

### **Concurrency:**

- Distributed systems often involve concurrent processing, where multiple nodes are working on different parts of a problem simultaneously.
- Coordination mechanisms are employed to manage concurrency and ensure that nodes work together harmoniously.

### **Fault Tolerance:**

- Distributed systems are designed to be resilient to failures. If one node fails, the system should be able to continue functioning without significant disruption.
- Replication and redundancy are common techniques used to achieve fault tolerance.

### **Scalability:**

- Distributed systems can scale horizontally by adding more nodes to handle increased load or demand.
- This makes it easier to accommodate growing workloads by simply adding more hardware.

### **Middleware:**

- Middleware is software that facilitates communication and data management between different nodes in a distributed system.
- It provides a layer of abstraction that simplifies the development of distributed applications.

### **Distributed Databases:**

- Distributed databases store data across multiple nodes, providing advantages in terms of performance, fault tolerance, and scalability.
- NoSQL databases like Cassandra and MongoDB are examples of distributed databases.

### **MapReduce and Big Data:**

- MapReduce is a programming model used for processing and generating large datasets in parallel across a distributed computing cluster.
- Distributed computing plays a crucial role in handling big data analytics tasks.

### **Cloud Computing:**

- Cloud computing often involves distributed systems where resources are provided on-demand over a network.
- Services like Amazon Web Services (AWS) and Microsoft Azure are examples of cloud platforms that leverage distributed computing.

Distributed computing is a broad field with applications ranging from scientific simulations and data analysis to web services and cloud computing. It enables the efficient use of resources and improved performance for computationally intensive tasks.

# Cluster computing

---

Cluster computing refers to the use of multiple interconnected computers or servers that work together as a single system to perform tasks or process data.

These individual computers, often referred to as nodes, are networked together and operate collaboratively to achieve a common goal.

Cluster computing is a form of **parallel computing**, where computations are **divided into smaller tasks** that can be executed simultaneously.

Key features of cluster computing include:

**Parallel Processing:** The primary purpose of clustering is to enable parallel processing. Tasks are divided among the nodes, and each node processes its **assigned portion of the workload concurrently**.

---

**High Performance:** By **distributing tasks across multiple nodes**, cluster computing can achieve **higher performance and throughput than a single computer**. This is particularly useful for computationally intensive tasks, such as scientific simulations, data analysis, and rendering.

**Scalability:** Clusters can be easily scaled by **adding more nodes to the network**. This scalability allows organizations to increase computing power as needed **to handle larger workloads**.

**Fault Tolerance:** Many cluster computing systems are designed to be fault-tolerant. If **one node fails**, the others can continue the computation, ensuring that the overall task is not disrupted.

**Load Balancing:** Cluster computing systems often include mechanisms for distributing the workload **evenly among the nodes to ensure efficient resource utilization**.

**Distributed Storage:** In addition to distributed processing, cluster computing often involves distributed storage systems. Data is distributed across multiple nodes, reducing the risk of data loss and improving access times.

There are different types of clusters, such as:

**High-Performance Computing (HPC) Clusters:** Designed for scientific and engineering applications that require significant computational power.

---

**Load Balancing Clusters:** Focus on distributing network traffic or computational tasks to ensure that **no single node is overwhelmed**.

**High Availability Clusters:** Emphasize **continuous operation and fault tolerance**, ensuring that critical services remain available **even if some nodes fail**.

**Data Clusters:** Emphasize distributed storage and processing of large datasets.

Cluster computing is widely used in various fields, including **scientific research, financial modeling, weather prediction, and data analysis**. Popular cluster computing frameworks include **Apache Hadoop, Apache Spark, and Kubernetes**, among others.



# Grid computing

---

Grid computing is a **form of distributed computing** that involves pooling together the computational resources of multiple interconnected computers to work on a task or solve a problem.

Unlike traditional computing models where a **single supercomputer performs the entire task**, grid computing **distributes the workload across a network of computers**, often geographically dispersed.

## **Key features of grid computing include:**

**Resource Sharing:** Grid computing allows organizations to **share computing resources** such as **processing power, storage, and specialized software** across a network. This sharing enables more efficient utilization of resources and cost savings.

**Distributed Computing:** **Tasks are divided into smaller sub-tasks**, and these sub-tasks are distributed to different computers in the grid. Each computer works on its assigned portion of the task, and the results are then aggregated to obtain the final outcome.

**Parallel Processing:** Grids often leverage parallel processing, where multiple processors work on different parts of a problem simultaneously. This can significantly reduce the time required to complete complex computations.

**Dynamic Resource Allocation:** Grid computing systems can dynamically **allocate resources** based on **demand**. If a particular task requires more computational power, the system can allocate additional resources to speed up the processing.

**Heterogeneous Environments:** Grids can be **composed of a variety of hardware and software configurations**. This heterogeneity allows organizations to integrate existing infrastructure into the grid without the need for a homogeneous environment.

**High Performance Computing (HPC):** Grid computing is often associated with high-performance computing, as it enables the efficient use of distributed resources for computationally intensive tasks, such as scientific simulations, data analysis, and modeling.

Grid computing has been used in various fields, including **scientific research, weather forecasting, drug discovery, and financial modeling**. Some well-known grid computing projects include the Worldwide LHC Computing Grid (WLCG) used for processing data from the Large Hadron Collider (LHC) at CERN and the SETI@home project, which used volunteers' computers to analyze radio signals for signs of extraterrestrial intelligence.

---

While grid computing has its advantages, it also presents challenges such as security concerns, data management issues, and the need for standardized protocols to ensure interoperability between different systems and resources in the grid.

# Cloud computing

---

Cloud computing refers to the **delivery of computing services, including storage, processing power, and applications, over the internet.**

**Instead of owning and maintaining physical hardware and software,** users can access these resources on-demand from a cloud service provider.

This model has gained significant popularity due to its **flexibility, scalability, and cost-effectiveness.**

Here are some key components and characteristics of cloud computing:

### Service Models:

- **Infrastructure as a Service (IaaS):** Provides virtualized computing resources over the internet. **Users can rent virtual machines, storage, and networks on a pay-as-you-go basis.**
- **Platform as a Service (PaaS):** Offers a platform that includes the **tools and services** needed for application development. **Users can focus on building and deploying applications** without worrying about the underlying infrastructure.
- **Software as a Service (SaaS):** Delivers software applications over the internet. Users can access these applications through a web browser without needing **to install or maintain software locally.**

### Deployment Models:

- **Public Cloud:** Resources are owned and operated by a **third-party cloud service provider** and **shared among multiple customers.**
- **Private Cloud:** Resources are used **exclusively by a single organization.** It can be managed by the organization itself or by a third party.
- **Hybrid Cloud:** Combines **both public and private clouds,** allowing data and applications to be shared between them.

## Essential Characteristics:

- **On-Demand Self-Service:** Users can provision and manage computing resources as needed **without human intervention** from the service provider.
- **Broad Network Access:** Services are available over the network and can be accessed through standard mechanisms.
- **Resource Pooling:** Computing resources are pooled to **serve multiple customers**, with ~~different physical and virtual resources dynamically assigned and reassigned according to~~ **demand**.
- **Rapid Elasticity:** Resources can be rapidly and elastically provisioned to **quickly scale out** and rapidly **released to scale in** based on demand.
- **Measured Service:** Cloud systems automatically control and optimize resource use by leveraging a **metering capability** at some level of abstraction.

## Benefits of Cloud Computing:

- **Cost Savings:** Organizations can reduce capital expenditure on hardware and maintenance and **pay only for the resources they use**.
- **Scalability:** Easily **scale up or down based on demand**.
- **Flexibility and Accessibility:** Users can **access resources** and applications from **anywhere with an internet connection**.
- **Reliability and Redundancy:** Many cloud providers offer **high levels of uptime and data redundancy**.

## Challenges:

**Security Concerns:** Storing data and applications in the cloud raises security and privacy issues.

---

**Downtime:** Reliance on internet connectivity means that downtime or outages can impact access to cloud services.

**Data Transfer Bottlenecks:** Moving large amounts of data to and from the cloud can be time-consuming.

Popular cloud service providers include **Amazon Web Services (AWS)**, **Microsoft Azure**, **Google Cloud Platform (GCP)**, and others. Cloud computing has become a fundamental technology for businesses of all sizes, enabling them to be more agile and efficient in their operations.

# Difference between Distributed, Cluster, Grid and Cloud computing

---

## Distributed Computing

It is to solve a single large problem by breaking it down into several tasks where each task is computed in the individual computers of the distributed system.

## Cluster computing

- Tightly coupled systems
- Single system image
- Centralized Job management & scheduling system

## Grid Computing

- Loosely coupled(Decentralization)
- Diversity and Dynamism
- Distributed Job Management& scheduling

## Cloud computing

- Dynamic computing infrastructure
- IT service centric approach
- Self service based usage model
- Minimally or self managed platform



# Key Differences: Cluster Vs Grid & Cloud Computing

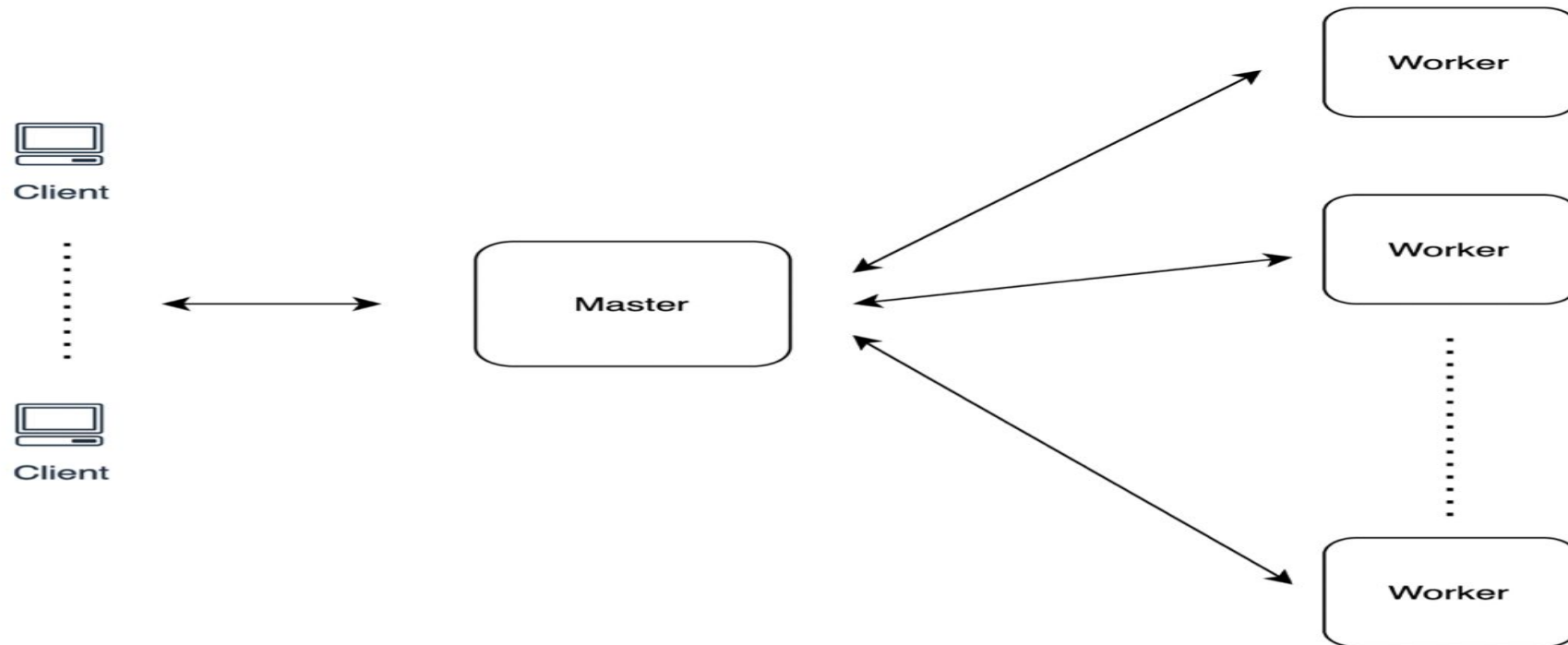
	Cluster	Cloud	Grid
1	Cluster connects computers in a Local Area Network	Cloud connects computers in a geographically distributed network	Grid also connects computers in a geographically distributed network
2	Tightly coupled network of computers	Loosely coupled network of computers	Loosely coupled network of computers
3	Made up of computers of homogenous hardware configurations	Possess Heterogeneous hardware configuration	Possess Heterogeneous hardware configuration

# Salient Differences: Cloud Computing Vs Grid Computing

	Cloud Computing	Grid Computing
Architecture	Centralized Resource Architecture	Distributed Resource Architecture
Resource	Resources are centrally managed.	Resources are managed on a collaboration pattern.
Flexibility	More flexible	Not as flexible as cloud
Payment	Users pay as per the usage model for Platform as a Service (PaaS), Infrastructure as a service (IaaS), or Software-as-a-Service (SaaS) services availed	Only set-up cost involved. Later no need to pay for anything.
Accessibility	highly accessible service. It can be accessed using conventional web protocols.	low on accessibility as compared to cloud computing. It can be accessed using grid middleware.

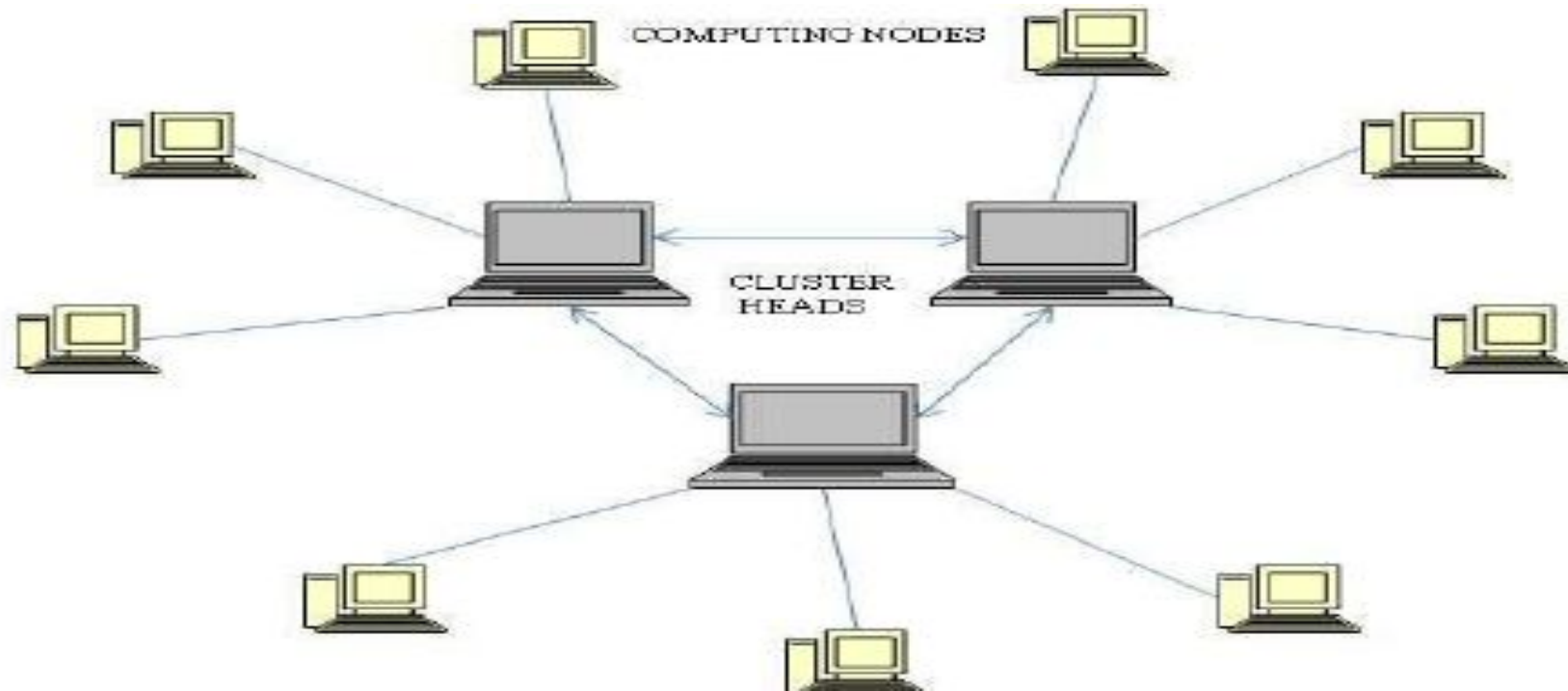
# Distributed Computing

---



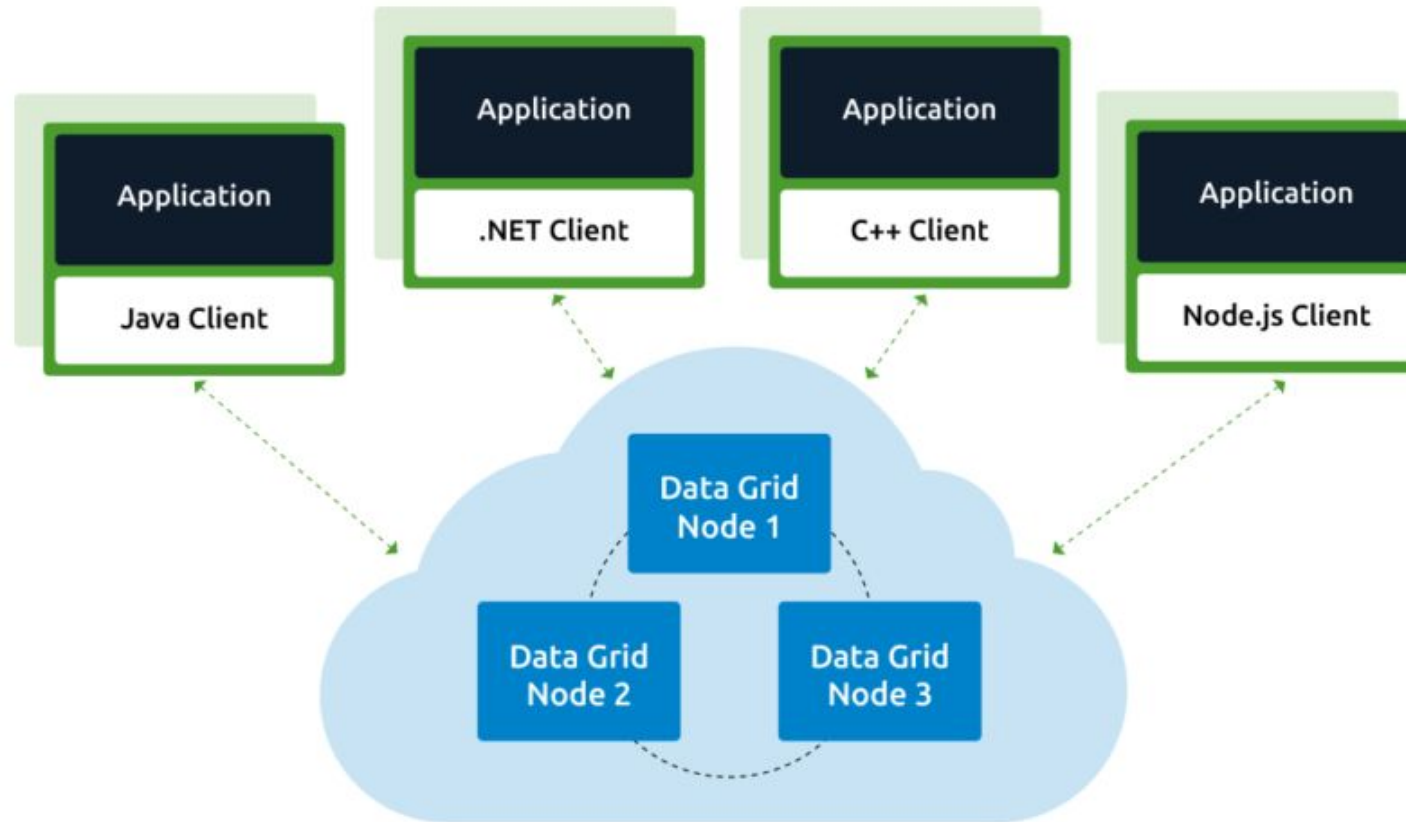
# Cluster Computing

---



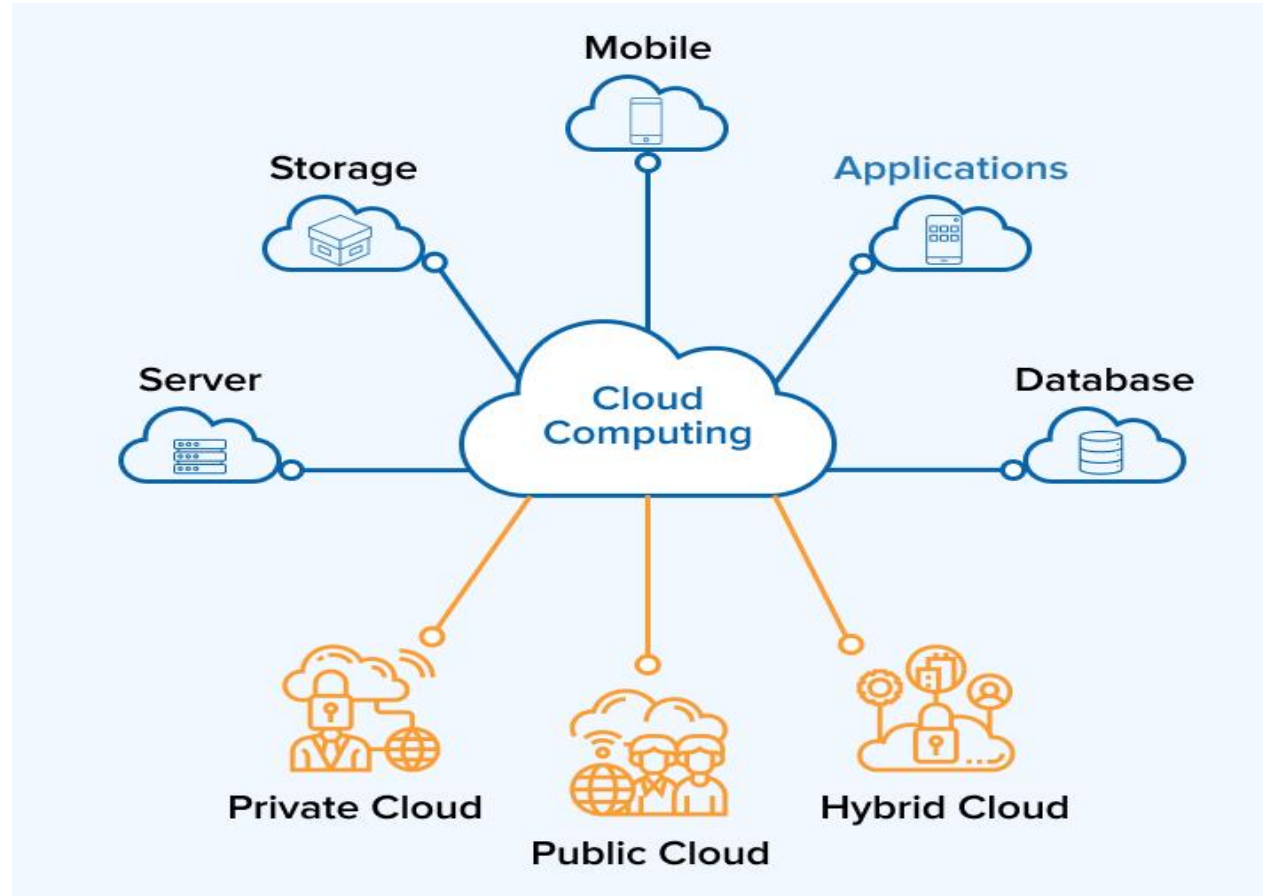
# Grid Computing

---



# Cloud Computing

---



# UNIT - II

## **CONCEPTS & TECHNOLOGIES**

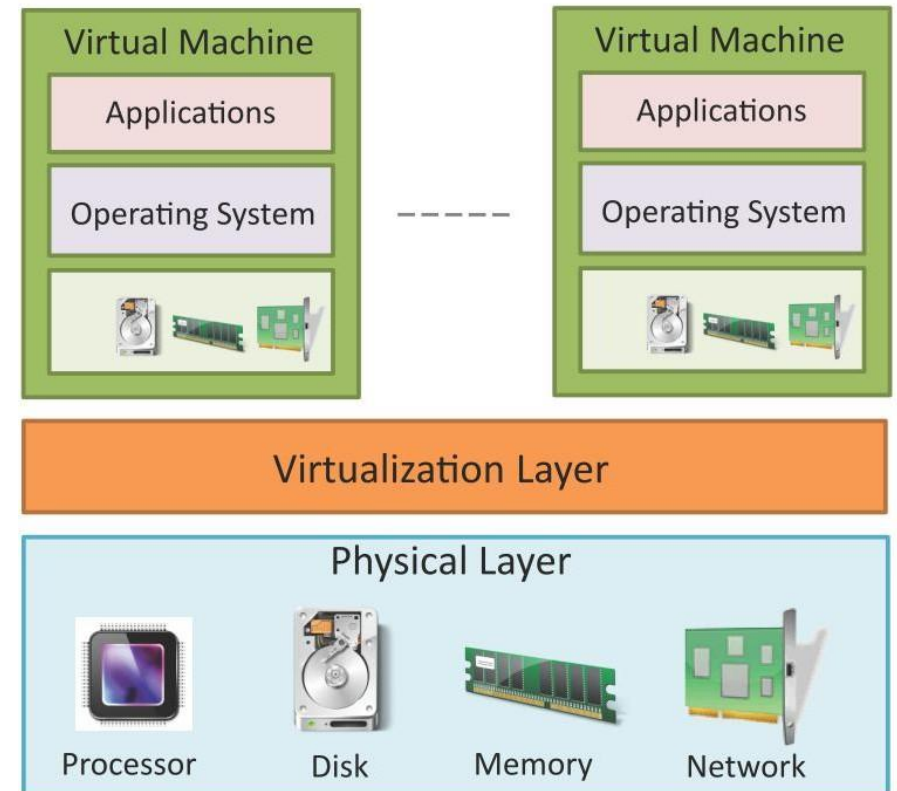
# Outline

- Concepts and enabling technologies of cloud computing
  - Virtualization
  - Load balancing
  - Scalability & Elasticity
  - Deployment
  - Replication
  - Monitoring
  - MapReduce
  - Identity and Access Management
  - Service Level Agreements
  - Billing



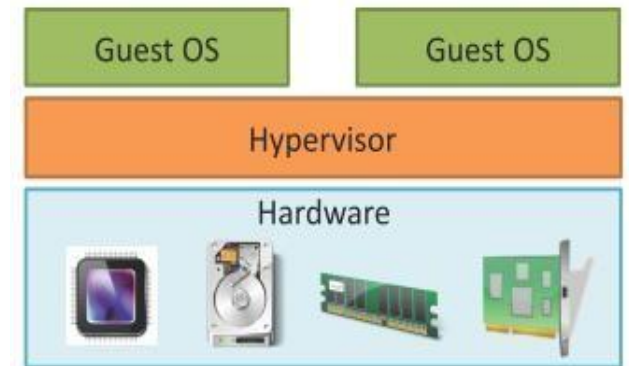
# Virtualization

- Virtualization refers to the partitioning the resources of a physical system (such as computing, storage, network and memory) into multiple virtual resources.
- Key enabling technology of cloud computing that allow pooling of resources.
- In cloud computing, resources are pooled to serve multiple users using multi-tenancy.

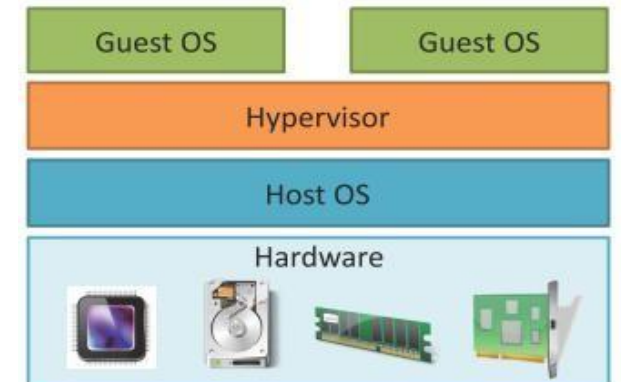


# Hypervisor

- The virtualization layer consists of a hypervisor or a virtual machine monitor (VMM).
- Hypervisor presents a virtual operating platform to a guest operating system (OS).
- Type-1 Hypervisor
  - Type-1 or the native hypervisors run directly on the host hardware and control the hardware and monitor the guest operating systems.
- Type-2 Hypervisor
  - Type 2 hypervisors or hosted hypervisors run on top of a conventional (main/host) operating system and monitor the guest operating systems.



Type-1 Hypervisor



Type-2 Hypervisor

# Types of Virtualization

- Full Virtualization

- In full virtualization, the virtualization layer completely decouples the guest OS from the underlying hardware. The guest OS requires no modification and is not aware that it is being virtualized. Full virtualization is enabled by direct execution of user requests and binary translation of OS requests.

- Para-Virtualization

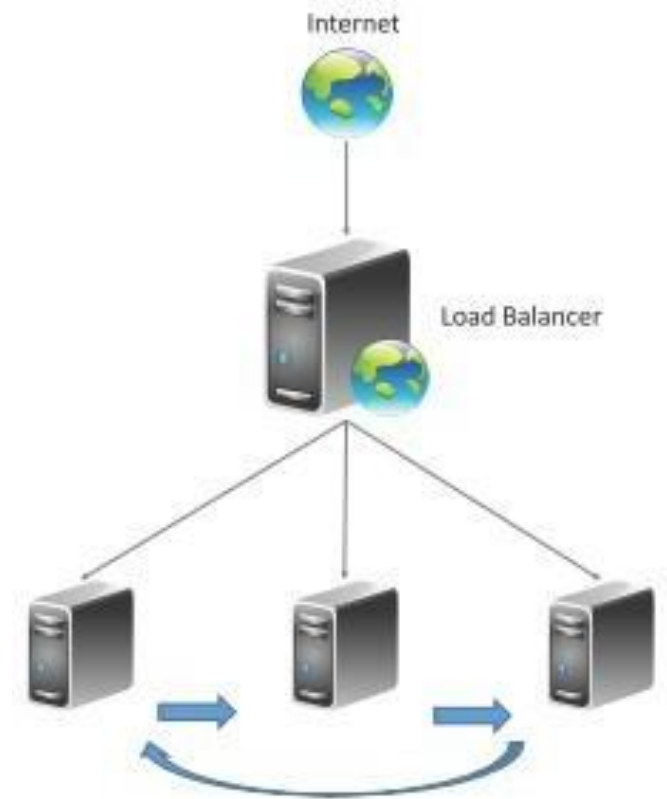
- In para-virtualization, the guest OS is modified to enable communication with the hypervisor to improve performance and efficiency. The guest OS kernel is modified to replace non-virtualizable instructions with hyper-calls that communicate directly with the virtualization layer hypervisor.

- Hardware Virtualization

- Hardware assisted virtualization is enabled by hardware features such as Intel's Virtualization Technology (VT-x) and AMD's AMD-V. In hardware assisted virtualization, privileged and sensitive calls are set to automatically trap to the hypervisor. Thus, there is no need for either binary translation or para-virtualization.

# Load Balancing

- Cloud computing resources can be scaled up on demand to meet the performance requirements of applications.
- Load balancing distributes workloads across multiple servers to meet the application workloads.
- The goals of load balancing techniques include:
  - Achieve maximum utilization of resources
  - Minimizing the response times
  - Maximizing throughput



# Load Balancing Algorithms

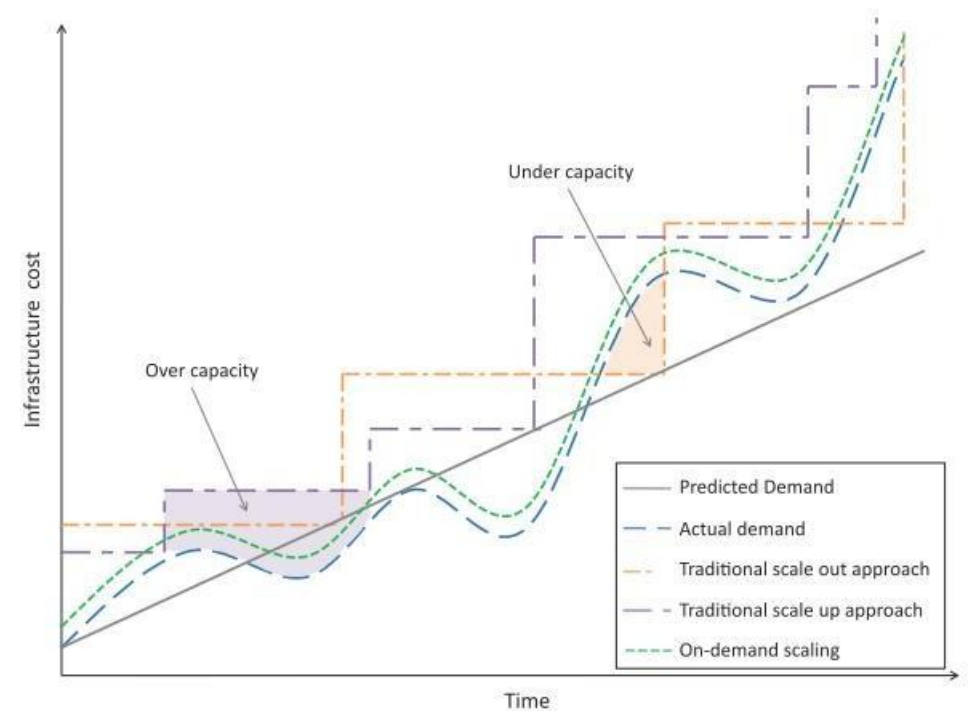
- Round Robin load balancing
- Weighted Round Robin load balancing
- Low Latency load balancing
- Least Connections load balancing
- Priority load balancing
- Overflow load balancing

# Load Balancing- Persistence Approaches

- Since load balancing can route successive requests from a user session to different servers, maintaining the state or the information of the session is important.
- Persistence Approaches
  - Sticky sessions
  - Session Database
  - Browser cookies
  - URL re-writing

# Scalability & Elasticity

- Multi-tier applications such as e-Commerce, social networking, business-to-business, etc. can experience rapid changes in their traffic.
- Capacity planning involves determining the right sizing of each tier of the deployment of an application in terms of the number of resources and the capacity of each resource.
- Capacity planning may be for computing, storage, memory or network resources.



# Scaling Approaches

- Vertical Scaling/Scaling up:
  - Involves upgrading the hardware resources (adding additional computing, memory, storage or network resources).
- Horizontal Scaling/Scaling out
  - Involves addition of more resources of the same type.



# Deployment

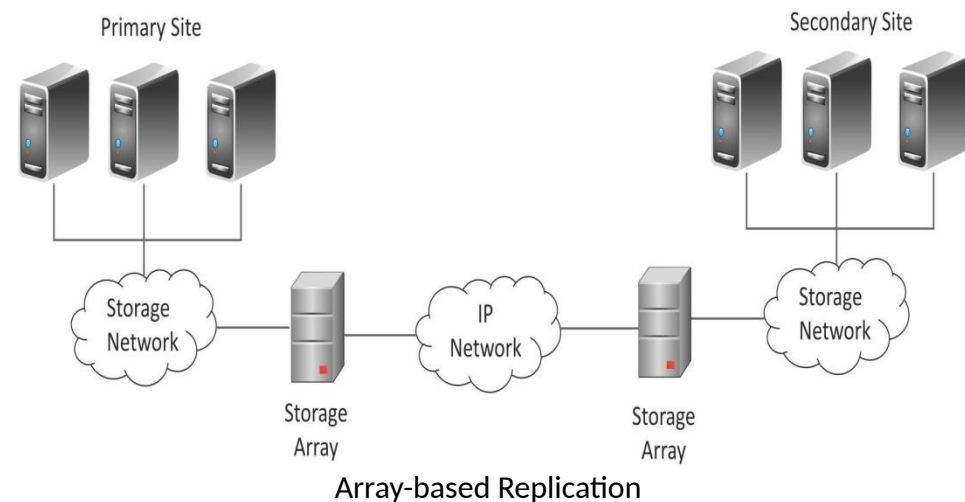
- Cloud application deployment design is an iterative process that involves:
  - Deployment Design
    - The variables in this step include the number of servers in each tier, computing, memory and storage capacities of servers, server interconnection, load balancing and replication strategies.
  - Performance Evaluation
    - To verify whether the application meets the performance requirements with the deployment.
    - Involves monitoring the workload on the application and measuring various workload parameters such as response time and throughput.
    - Utilization of servers (CPU, memory, disk, I/O, etc.) in each tier is also monitored.
  - Deployment Refinement
    - Various alternatives can exist in this step such as vertical scaling (or scaling up), horizontal scaling (or scaling out), alternative server interconnections, alternative load balancing and replication strategies, for instance.

# Replication

- Replication is used to create and maintain multiple copies of the data in the cloud.
- Cloud enables rapid implementation of replication solutions for disaster recovery for organizations.
- With cloud-based data replication organizations can plan for disaster recovery without making any capital expenditures on purchasing, configuring and managing secondary site locations.

- Types:

- Array-based Replication
- Network-based Replication
- Host-based Replication



# Monitoring

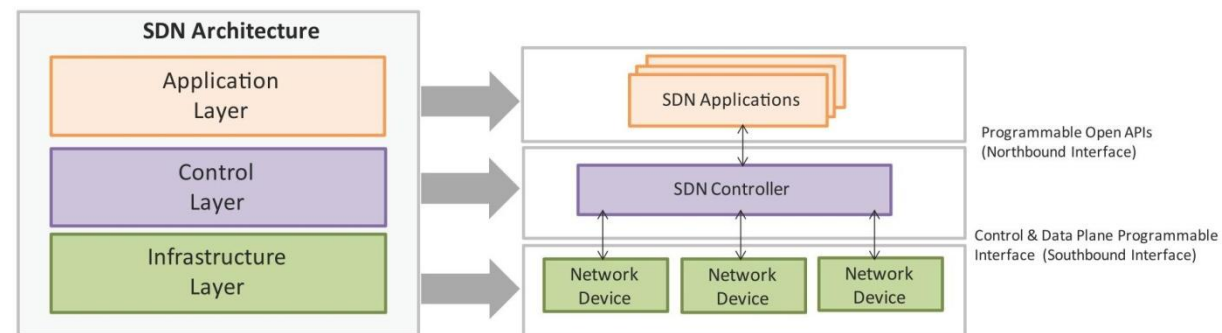
- Monitoring services allow cloud users to collect and analyze the data on various monitoring metrics.
- A monitoring service collects data on various system and application metrics from the cloud computing instances.
- Monitoring of cloud resources is important because it allows the users to keep track of the health of applications and services deployed in the cloud.

Examples of Monitoring Metrics

Type	Metrics
CPU	CPU-Usage, CPU-Idle
Disk	Disk-Usage, Bytes/sec (read/write), Operations/sec
Memory	Memory-Used, Memory-Free, Page-Cache
Interface	Packets/sec (incoming/outgoing), Octets/sec(incoming/outgoing)

# Software Defined Networking

- Software-Defined Networking (SDN) is a networking architecture that separates the control plane from the data plane and centralizes the network controller.
- Conventional network architecture
  - The control plane and data plane are coupled. Control plane is the part of the network that carries the signaling and routing message traffic while the data plane is the part of the network that carries the payload data traffic.
- SDN Architecture
  - The control and data planes are decoupled and the network controller is centralized.

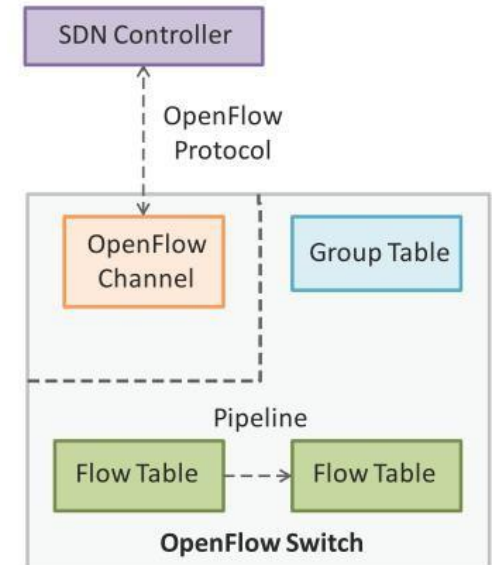


# SDN - Key Elements

- **Centralized Network Controller**
  - With decoupled the control and data planes and centralized network controller, the network administrators can rapidly configure the network.
- **Programmable Open APIs**
  - SDN architecture supports programmable open APIs for interface between the SDN application and control layers (Northbound interface). These open APIs that allow implementing various network services such as routing, quality of service (QoS), access control, etc.
- **Standard Communication Interface (OpenFlow)**
  - SDN architecture uses a standard communication interface between the control and infrastructure layers (Southbound interface). OpenFlow, which is defined by the Open Networking Foundation (ONF) is the broadly accepted SDN protocol for the Southbound interface.

# OpenFlow

- OpenFlow is the broadly accepted SDN protocol for the Southbound interface.
- With OpenFlow, the forwarding plane of the network devices can be directly accessed and manipulated.
- OpenFlow uses the concept of flows to identify network traffic based on pre-defined match rules.
- Flows can be programmed statically or dynamically by the SDN control software.
- OpenFlow protocol is implemented on both sides of the interface between the controller and the network devices.



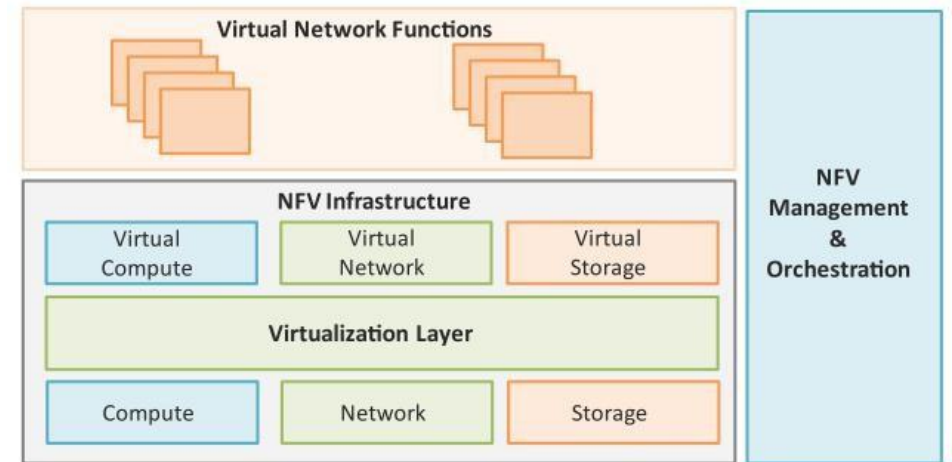
OpenFlow switch comprising of one or more flow tables and a group table, which perform packet lookups and forwarding, and OpenFlow channel to an external controller.

# Network Function Virtualization

- Network Function Virtualization (NFV) is a technology that leverages virtualization to consolidate the heterogeneous network devices onto industry standard high volume servers, switches and storage.
- Relationship to SDN
  - NFV is complementary to SDN as NFV can provide the infrastructure on which SDN can run.
  - NFV and SDN are mutually beneficial to each other but not dependent.
  - Network functions can be virtualized without SDN, similarly, SDN can run without NFV.
- NFV comprises of network functions implemented in software that run on virtualized resources in the cloud.
- NFV enables a separation the network functions which are implemented in software from the underlying hardware.

# NFV Architecture

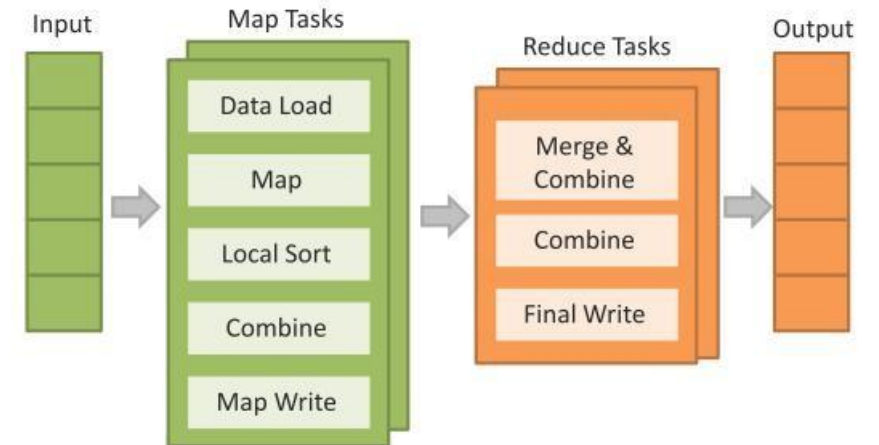
- Key elements of the NFV architecture are
  - Virtualized Network Function (VNF): VNF is a software implementation of a network function which is capable of running over the NFV Infrastructure (NFVI).
  - NFV Infrastructure (NFVI): NFVI includes compute, network and storage resources that are virtualized.
  - NFV Management and Orchestration: NFV Management and Orchestration focuses on all virtualization-specific management tasks and covers the orchestration and lifecycle management of physical and/or software resources that support the infrastructure virtualization, and the lifecycle management of VNFs.





# MapReduce

- MapReduce is a parallel data processing model for processing and analysis of massive scale data.
- MapReduce phases:
  - **Map Phase:** In the Map phase, data is read from a distributed file system, partitioned among a set of computing nodes in the cluster, and sent to the nodes as a set of key-value pairs.
  - The Map tasks process the input records independently of each other and produce intermediate results as key-value pairs.
  - The intermediate results are stored on the local disk of the node running the Map task.
  - **Reduce Phase:** When all the Map tasks are completed, the Reduce phase begins in which the intermediate data with the same key is aggregated.



# Identity and Access Management

- Identity and Access Management (IDAM) for cloud describes the authentication and authorization of users to provide secure access to cloud resources.
- Organizations with multiple users can use IDAM services provided by the cloud service provider for management of user identifiers and user permissions.
- IDAM services allow organizations to centrally manage users, access permissions, security credentials and access keys.
- Organizations can enable role-based access control to cloud resources and applications using the IDAM services.
- IDAM services allow creation of user groups where all the users in a group have the same access permissions.
- Identity and Access Management is enabled by a number of technologies such as OpenAuth, Role-based Access Control (RBAC), Digital Identities, Security Tokens, Identity Providers, etc.

# Billing

Cloud service providers offer a number of billing models described as follows:

- Elastic Pricing
  - In elastic pricing or pay-as-you-use pricing model, the customers are charged based on the usage of cloud resources.
- Fixed Pricing
  - In fixed pricing models, customers are charged a fixed amount per month for the cloud resources.
- Spot Pricing
  - Spot pricing models offer variable pricing for cloud resources which is driven by market demand.

# Further Reading

- Network Functions Virtualization, <http://www.etsi.org/technologies-clusters/technologies/nfv>, Retrieved 2013.
- OpenFlow Switch Specification, <https://www.opennetworking.org>, Retrieved 2013.
- J. Dean, S. Ghemawat, MapReduce: Simplified Data Processing on Large Clusters, OSDI 2004.
- VMware, Understanding Full Virtualization, Paravirtualization, and Hardware Assist, 2007.

# UNIT 3

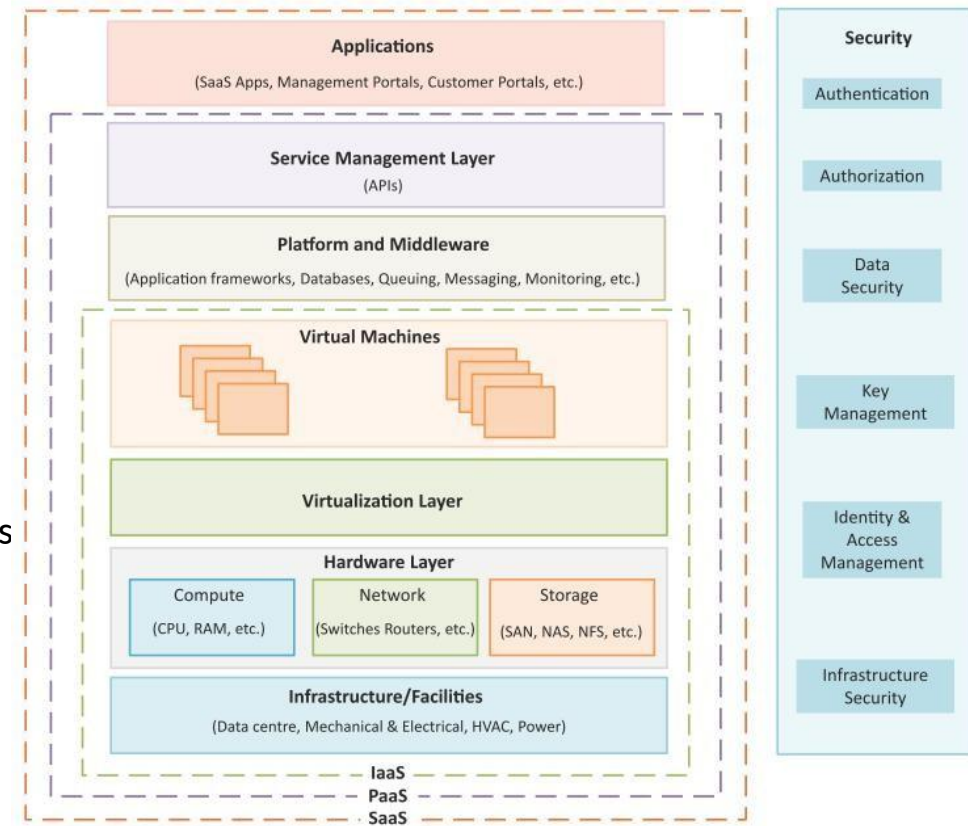
## Cloud Services & Platforms

# Outline

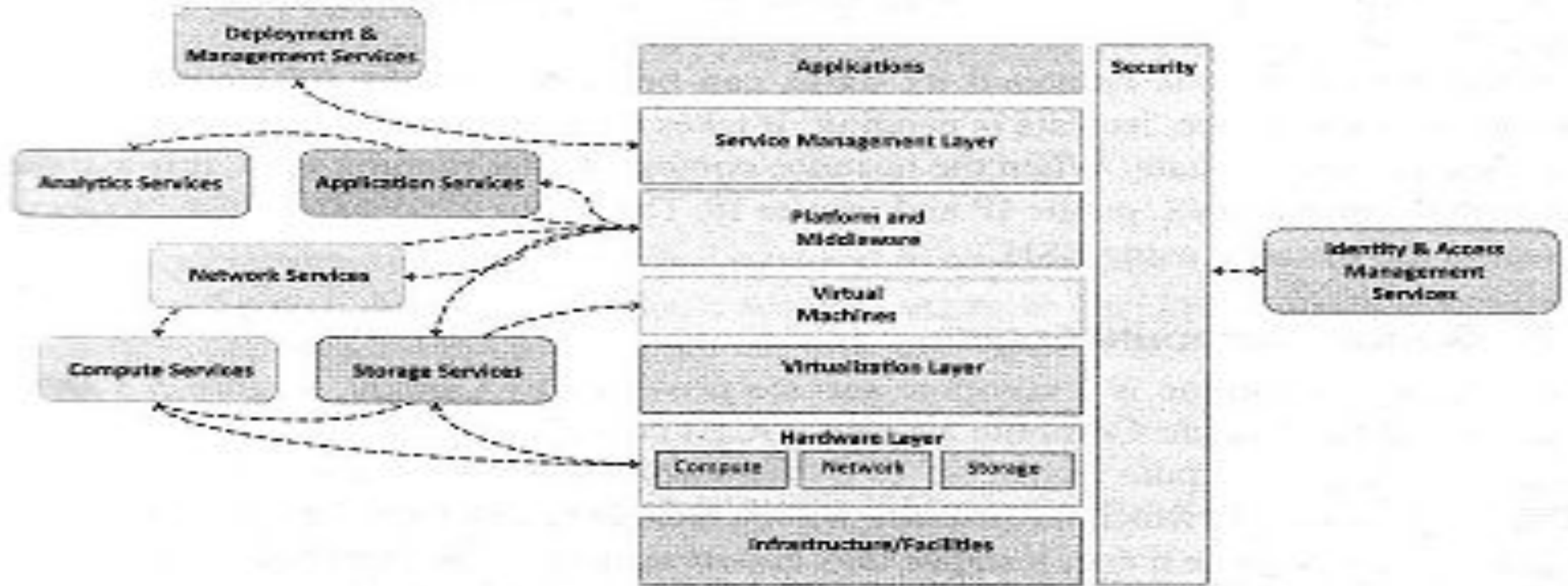
- Compute Services
- Storage Services
- Database Services
- Application Services
- Content Delivery Services
- Analytics Services
- Deployment & Management Services
- Identity & Access Management Services

# Cloud Reference Model

- **Infrastructure & Facilities Layer**
  - Includes the physical infrastructure such as datacenter facilities, electrical and mechanical equipment, etc.
- **Hardware Layer**
  - Includes physical compute, network and storage hardware.
- **Virtualization Layer**
  - Partitions the physical hardware resources into multiple virtual resources that enabling pooling of resources.
- **Platform & Middleware Layer**
  - Builds upon the IaaS layers below and provides standardized stacks of services such as database service, queuing service, application frameworks and run-time environments, messaging services, monitoring services, analytics services, etc.
- **Service Management Layer**
  - Provides APIs for requesting, managing and monitoring cloud resources.
- **Applications Layer**
  - Includes SaaS applications such as Email, cloud storage application, productivity applications, management portals, customer self-service portals, etc.



# Cloud services



(b) Cloud services



# Compute Services

- Compute services provide dynamically scalable compute capacity in the cloud.
- Compute resources can be provisioned on-demand in the form of virtual machines. Virtual machines can be created from standard images provided by the cloud service provider or custom images created by the users.
- Compute services can be accessed from the web consoles of these services that provide graphical user interfaces for provisioning, managing and monitoring these services.
- Cloud service providers also provide APIs for various programming languages that allow developers to access and manage these services programmatically.

# Features

- Scalable
- Flexible
- Secure
- Cost effective

# Compute Services – Amazon EC2

- Amazon Elastic Compute Cloud (EC2) is a compute service provided by Amazon.
- Launching EC2 Instances
  - To launch a new instance click on the launch instance button. This will open a wizard where you can select the Amazon machine image (AMI) with which you want to launch the instance. You can also create their own AMIs with custom applications, libraries and data. Instances can be launched with a variety of operating systems.
- Instance Sizes
  - When you launch an instance you specify the instance type (micro, small, medium, large, extra-large, etc.), the number of instances to launch based on the selected AMI and availability zones for the instances.
- Key-pairs
  - When launching a new instance, the user selects a key-pair from existing keypairs or creates a new keypair for the instance. Keypairs are used to securely connect to an instance after it launches.
- Security Groups
  - The security groups to be associated with the instance can be selected from the instance launch wizard. Security groups are used to open or block a specific network port for the launched instances.

The screenshot displays the Amazon EC2 console interface. On the left, a navigation sidebar lists categories like EC2 Dashboard, INSTANCES, IMAGES, ELASTIC BLOCK STORE, and NETWORK & SECURITY. The main content area is titled 'Resources' and shows a summary of EC2 resources in the US West (Oregon) region: 0 Running Instances, 0 Elastic IPs, 0 Volumes, 0 Snapshots, 0 Key Pairs, 0 Load Balancers, 0 Placement Groups, and 12 Security Groups. Below this is a 'Create Instance' section with a 'Launch Instance' button. A 'Service Health' section indicates that the service is operating normally in the US West (Oregon) region. An 'Availability Zone Status' section shows that two availability zones (us-west-2a and us-west-2b) are also operating normally. On the right side, there are sections for 'Account Attributes' (including Supported Platforms and Default VPC) and 'Additional Information' (including Getting Started Guide, Documentation, and Popular AMIs on AWS Marketplace).

# Compute Services – Google Compute Engine

- Google Compute Engine is a compute service provided by Google.
- Launching Instances
  - To create a new instance, the user selects an instance machine type, a zone in which the instance will be launched, a machine image for the instance and provides an instance name, instance tags and meta-data.
- Disk Resources
  - Every instance is launched with a disk resource. Depending on the instance type, the disk resource can be a scratch disk space or persistent disk space. The scratch disk space is deleted when the instance terminates. Whereas, persistent disks live beyond the life of an instance.
- Network Options
  - Network option allows you to control the traffic to and from the instances. By default, traffic between instances in the same network, over any port and any protocol and incoming SSH connections from anywhere are enabled.

Google Cloud Console

Cloud Project ID: cloud Compute Engine

Instances NEW INSTANCE

Disks  
Snapshots  
Images  
Networks  
Metadata  
Zones  
Operations  
Quotas

### Create a new Instance

Name

Description   
Optional

Tags   
Optional

Metadata   -- +

#### Summary

**myinstance**  
My instance

**debian-7-wheezy-v20130723**  
Debian GNU/Linux 7.1 (wheezy) b...

**us-central1-b**

**1 vCPU, 3.75 GB RAM**

**default**  
Default network for the project

Note: per-minute charges will begin now

#### Location and Resources

Zone

Machine Type

Boot Source

Image

Additional Disks   
Optional

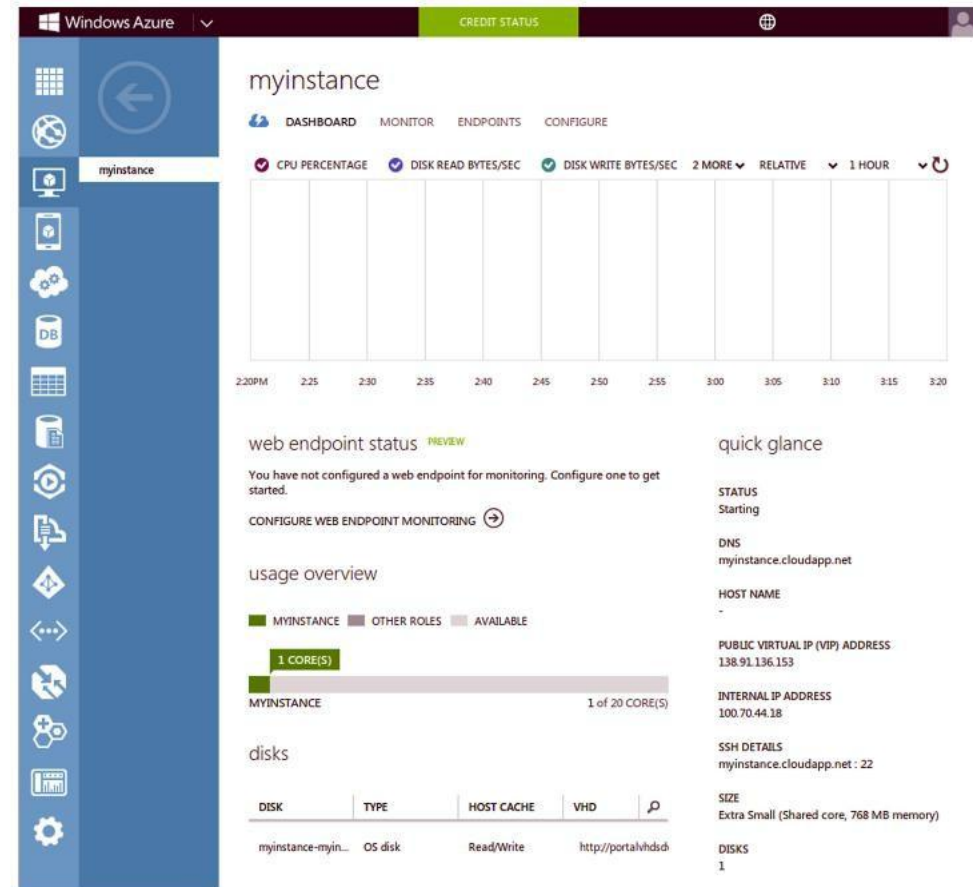
#### Networking

Network

External IP

# Compute Services – Windows Azure VMs

- Windows Azure Virtual Machines is the compute service from Microsoft.
- Launching Instances:
  - To create a new instance, you select the instance type and the machine image.
  - You can either provide a user name and password or upload a certificate file for securely connecting to the instance.
  - Any changes made to the VM are persistently stored and new VMs can be created from the previously stored machine images.

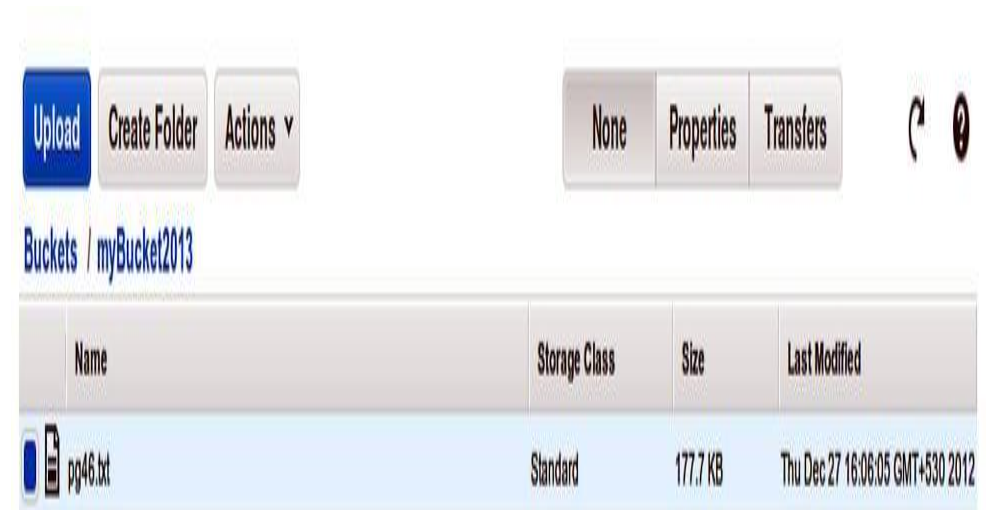


# Storage Services

- Cloud storage services allow storage and retrieval of any amount of data, at any time from anywhere on the web.
- Most cloud storage services organize data into buckets or containers.
- Scalability
  - Cloud storage services provide high capacity and scalability. Objects upto several tera-bytes in size can be uploaded and multiple buckets/containers can be created on cloud storages.
- Replication
  - When an object is uploaded it is replicated at multiple facilities and/or on multiple devices within each facility.
- Access Policies
  - Cloud storage services provide several security features such as Access Control Lists (ACLs), bucket/container level policies, etc. ACLs can be used to selectively grant access permissions on individual objects. Bucket/container level policies can also be defined to allow or deny permissions across some or all of the objects within a single bucket/container.
- Encryption
  - Cloud storage services provide Server Side Encryption (SSE) options to encrypt all data stored in the cloud storage.
- Consistency
  - Strong data consistency is provided for all upload and delete operations. Therefore, any object that is uploaded can be immediately downloaded after the upload is complete.

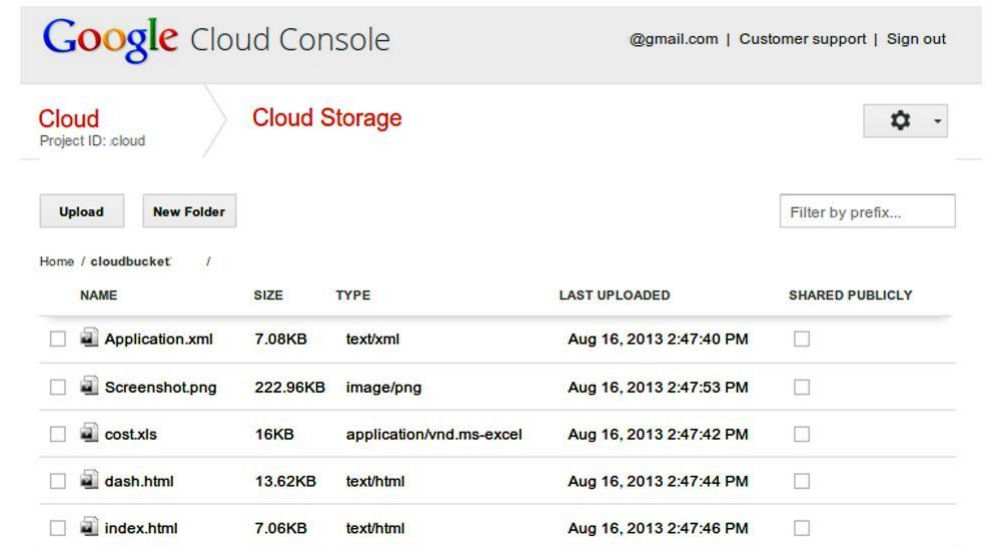
# Storage Simple Services – Amazon S3

- Amazon Simple Storage Service(S3) is an online cloud-based data storage infrastructure for storing and retrieving any amount of data.
- S3 provides highly reliable, scalable, fast, fully redundant and affordable storage infrastructure.
- Buckets
  - Data stored on S3 is organized in the form of buckets. You must create a bucket before you can store data on S3.
- Uploading Files to Buckets
  - S3 console provides simple wizards for creating a new bucket and uploading files.
  - You can upload any kind of file to S3.
  - While uploading a file, you can specify the redundancy and encryption options and access permissions.



# Storage Services – Google Cloud Storage

- GCS is the Cloud storage service from Google
- Buckets
  - Objects in GCS are organized into buckets.
- Access Control Lists
  - ACLs are used to control access to objects and buckets. ACLs can be configured to share objects and buckets with the entire world, a Google group, a Google-hosted domain, or specific Google account holders.



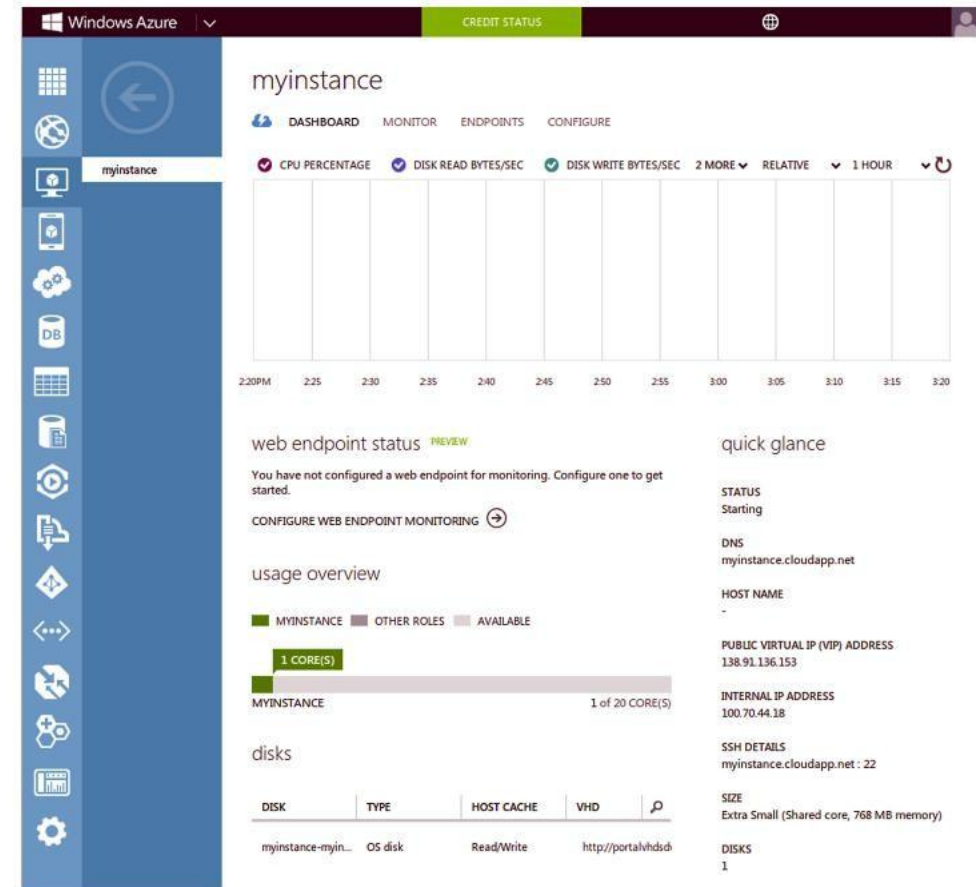
The screenshot displays the Google Cloud Console interface for the Cloud Storage service. At the top, the Google Cloud Console logo is visible, along with the user's email address (@gmail.com), a link to Customer support, and a Sign out button. Below the header, the 'Cloud' section is active, showing the Project ID as 'cloud'. The 'Cloud Storage' section is also visible, with a settings gear icon. The main content area shows the 'Upload' and 'New Folder' buttons, and a 'Filter by prefix...' search box. The breadcrumb path is 'Home / cloudbucket /'. A table lists the objects in the bucket, with columns for NAME, SIZE, TYPE, LAST UPLOADED, and SHARED PUBLICLY. The objects listed are:

NAME	SIZE	TYPE	LAST UPLOADED	SHARED PUBLICLY
<input type="checkbox"/> Application.xml	7.08KB	text/xml	Aug 16, 2013 2:47:40 PM	<input type="checkbox"/>
<input type="checkbox"/> Screenshot.png	222.96KB	image/png	Aug 16, 2013 2:47:53 PM	<input type="checkbox"/>
<input type="checkbox"/> cost.xls	16KB	application/vnd.ms-excel	Aug 16, 2013 2:47:42 PM	<input type="checkbox"/>
<input type="checkbox"/> dash.html	13.62KB	text/html	Aug 16, 2013 2:47:44 PM	<input type="checkbox"/>
<input type="checkbox"/> index.html	7.06KB	text/html	Aug 16, 2013 2:47:46 PM	<input type="checkbox"/>



# Storage Services – Windows Azure Storage

- Windows Azure Storage is the cloud storage service from Microsoft.
- Windows Azure Storage provides various storage services such as blob storage service, table service and queue service.
- Blob storage service
  - The blob storage service allows storing unstructured binary data or binary large objects (blobs).
  - Blobs are organized into containers.
  - Block blobs - can be subdivided into some number of blocks. If a failure occurs while transferring a block blob, retransmission can resume with the most recent block rather than sending the entire blob again.
  - Page blobs - are divided into number of pages and are designed for random access. Applications can read and write individual pages at random in a page blob.

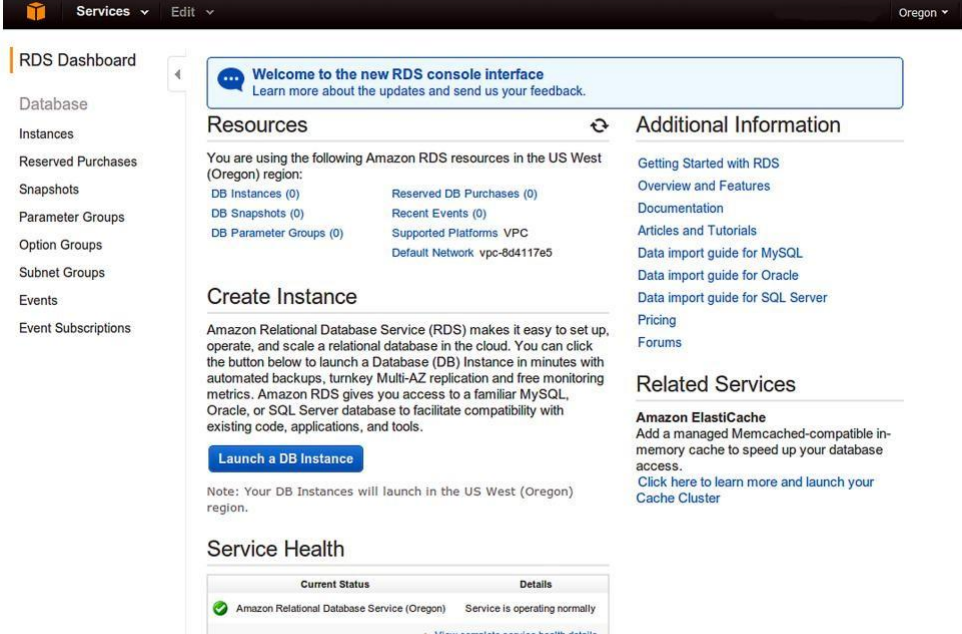


# Database Services

- Cloud database services allow you to set-up and operate relational or non-relational databases in the cloud.
- Relational Databases
  - Popular relational databases provided by various cloud service providers include MySQL, Oracle, SQL Server, etc.
- Non-relational Databases
  - The non-relational (No-SQL) databases provided by cloud service providers are mostly proprietary solutions.
- Scalability
  - Cloud database services allow provisioning as much compute and storage resources as required to meet the application workload levels. Provisioned capacity can be scaled-up or down. For read-heavy workloads, read-replicas can be created.
- Reliability
  - Cloud database services are reliable and provide automated backup and snapshot options.
- Performance
  - Cloud database services provide guaranteed performance with options such as guaranteed input/output operations per second (IOPS) which can be provisioned upfront.
- Security
  - Cloud database services provide several security features to restrict the access to the database instances and stored data, such as network firewalls and authentication mechanisms.

# Database Services – Amazon RDS

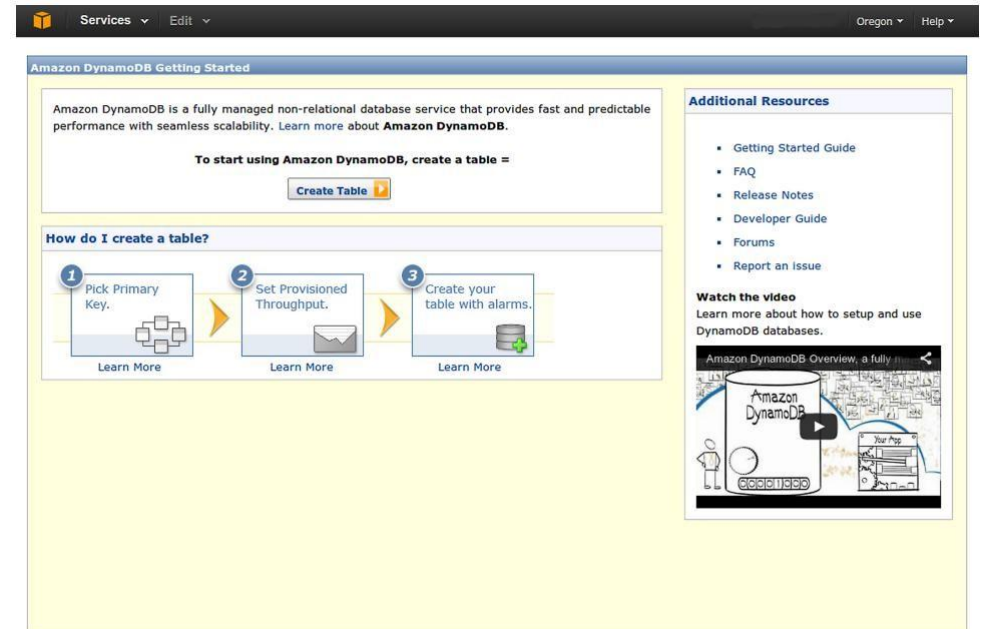
- Amazon Relational Database Service (RDS) is a web service that makes it easy to setup, operate and scale a relational database in the cloud.
- Launching DB Instances
  - The console provides an instance launch wizard that allows you to select the type of database to create (MySQL, Oracle or SQL Server) database instance size, allocated storage, DB instance identifier, DB username and password. The status of the launched DB instances can be viewed from the console.
- Connecting to a DB Instance
  - Once the instance is available, you can note the instance end point from the instance properties tab. This end point can then be used for securely connecting to the instance.



The screenshot displays the Amazon RDS console interface. At the top, there is a navigation bar with 'Services' and 'Edit' dropdown menus, and the region 'Oregon' is selected. The main content area is titled 'RDS Dashboard' and includes a sidebar with navigation options: Database, Instances, Reserved Purchases, Snapshots, Parameter Groups, Option Groups, Subnet Groups, Events, and Event Subscriptions. A welcome message states: 'Welcome to the new RDS console interface. Learn more about the updates and send us your feedback.' The 'Resources' section lists Amazon RDS resources in the US West (Oregon) region: DB Instances (0), Reserved DB Purchases (0), DB Snapshots (0), Recent Events (0), DB Parameter Groups (0), Supported Platforms VPC, and Default Network vpc-8d4117e5. The 'Create Instance' section provides a description of Amazon RDS and a 'Launch a DB Instance' button. A note indicates that DB Instances will launch in the US West (Oregon) region. The 'Service Health' section shows the current status of Amazon Relational Database Service (Oregon) as 'Service is operating normally' with a link to view complete service health details. The 'Additional Information' section includes links for Getting Started with RDS, Overview and Features, Documentation, Articles and Tutorials, Data import guides for MySQL, Oracle, and SQL Server, Pricing, and Forums. The 'Related Services' section features Amazon ElastiCache, described as a managed Memcached-compatible in-memory cache to speed up database access, with a link to learn more and launch a Cache Cluster.

# Database Services – Amazon DynamoDB

- Amazon DynamoDB is the non-relational (No-SQL) database service from Amazon.
- Data Model
  - The DynamoDB data model includes include tables, items and attributes.
  - A table is a collection of items and each item is a collection of attributes.
  - To store data in DynamoDB you have to create a one or more tables and specify how much throughput capacity you want to provision and reserve for reads and writes.
- Fully Managed Service
  - DynamoDB is a fully managed service that automatically spreads the data and traffic for the stored tables over a number of servers to meet the throughput requirements specified by the users.
- Replication
  - All stored data is automatically replicated across multiple availability zones to provide data durability.



# Storage Services – Google Cloud SQL

- Google SQL is the relational database service from Google.
- Google Cloud SQL service allows you to host MySQL databases in the Google's cloud.
- Launching DB Instances
  - You can create new database instances from the console and manage existing instances. To create a new instance you select a region, database tier, billing plan and replication mode.
- Backups
  - You can schedule daily backups for your Google Cloud SQL instances, and also restore backed-up databases.
- Replication
  - Cloud SQL provides both synchronous or asynchronous geographic replication and the ability to import/ export databases.

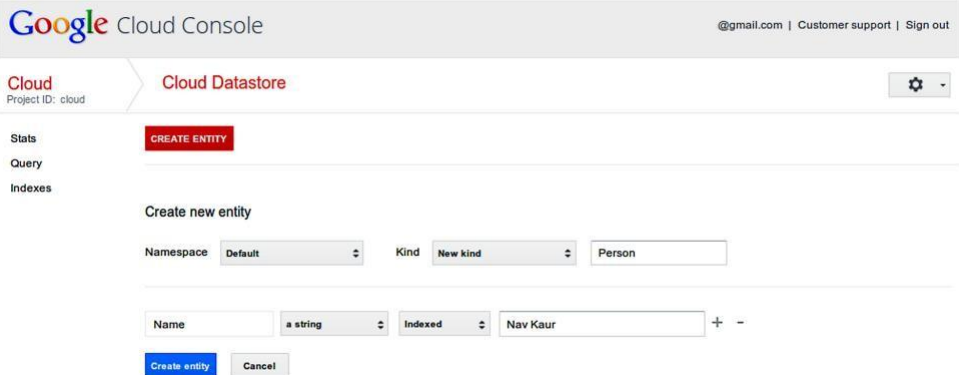
The screenshot displays the Google Cloud Console interface for creating a new Cloud SQL instance. The breadcrumb navigation shows 'Cloud SQL' and 'cloud:mydbinstance'. The main heading is 'Create a New Cloud SQL Instance'. The form includes the following sections:

- Instance ID:** mydbinstance
- Region:** United States
- Resources and Billing:**
  - Tier:** D0 - 128M RAM
  - Billing Plan:** Per Use (selected), Package
- Options:**
  - Backup Window:** 7:30 AM - 11:30 AM
  - Replication:** Synchronous (selected), Asynchronous

The Summary sidebar on the right provides a quick overview: mydbinstance, United States, D0 (\$0.025 per hour - 128M RAM), Per Use, Backups at 7:30 AM - 11:30 AM, and Synchronous replication. It includes 'Confirm' and 'Cancel' buttons.

# Storage Services – Google Cloud Datastore

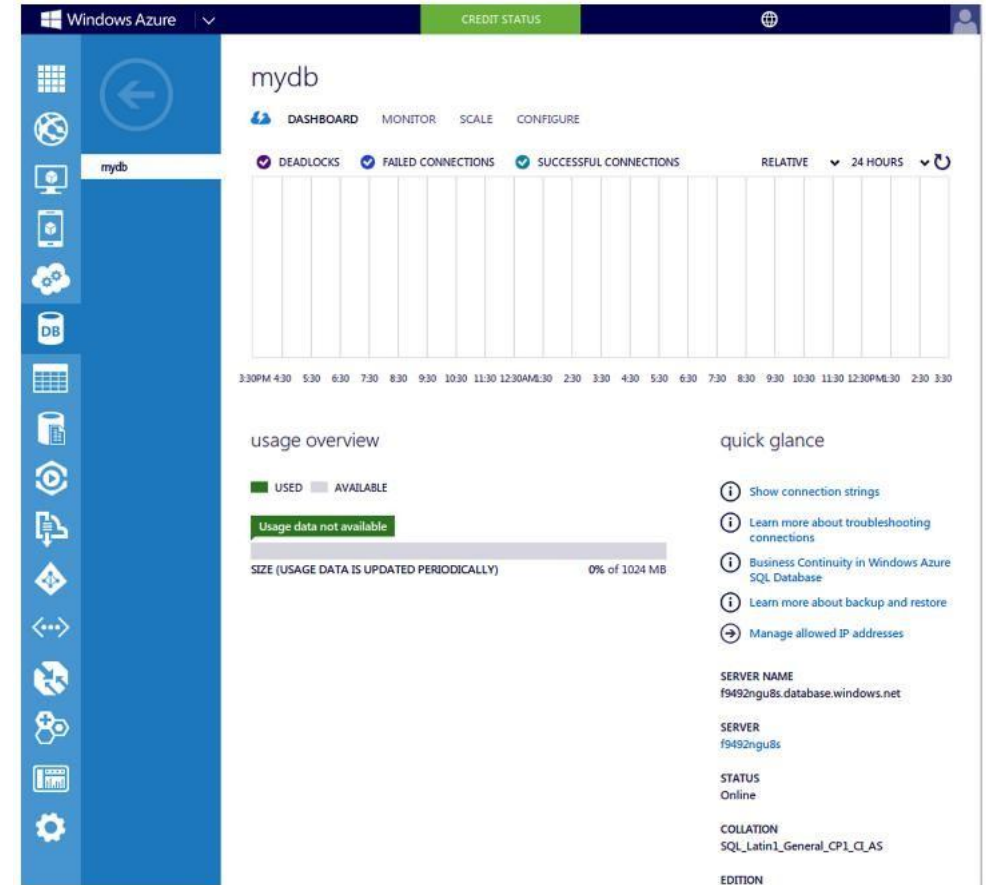
- Google Cloud Datastore is a fully managed non-relational database from Google.
- Cloud Datastore offers ACID transactions and high availability of reads and writes.
- Data Model
  - The Cloud Datastore data model consists of entities. Each entity has one or more properties (key-value pairs) which can be of one of several supported data types, such as strings and integers. Each entity has a kind and a key. The entity kind is used for categorizing the entity for the purpose of queries and the entity key uniquely identifies the entity.



The screenshot shows the Google Cloud Console interface for Cloud Datastore. The top navigation bar includes the Google Cloud Console logo, a user profile icon, and links for Customer support and Sign out. The main header shows 'Cloud Datastore' with a settings icon. A sidebar on the left lists 'Stats', 'Query', and 'Indexes'. A red 'CREATE ENTITY' button is visible. Below, the 'Create new entity' section has a 'Namespace' dropdown set to 'Default', a 'Kind' dropdown set to 'New kind', and a text input field containing 'Person'. A table below shows a property named 'Name' with a data type of 'a string', an 'Indexed' checkbox, and a value of 'Nav Kaur'. At the bottom, there are 'Create entity' and 'Cancel' buttons.

# Storage Services – Windows Azure SQL DB

- Windows Azure SQL Database is the relational database service from Microsoft.
- Azure SQL Database is based on the SQL server, but it does not give each customer a separate instance of SQL server.
- Multi-tenant Service
  - SQL Database is a multi-tenant service, with a logical SQL Database server for each customer.



# Storage Services – Windows Azure Table Service

- Windows Azure Table Service is a non-relational (No-SQL) database service from Microsoft.
- Data Model
  - The Azure Table Service data model consists of tables having multiple entities.
  - Tables are divided into some number of partitions, each of which can be stored on a separate machine.
  - Each partition in a table holds a specified number of entities, each containing as many as 255 properties.
  - Each property can be one of the several supported data types such as integers and strings.
- No Fixed Schema
  - Tables do not have a fixed schema and different entities in a table can have different properties.

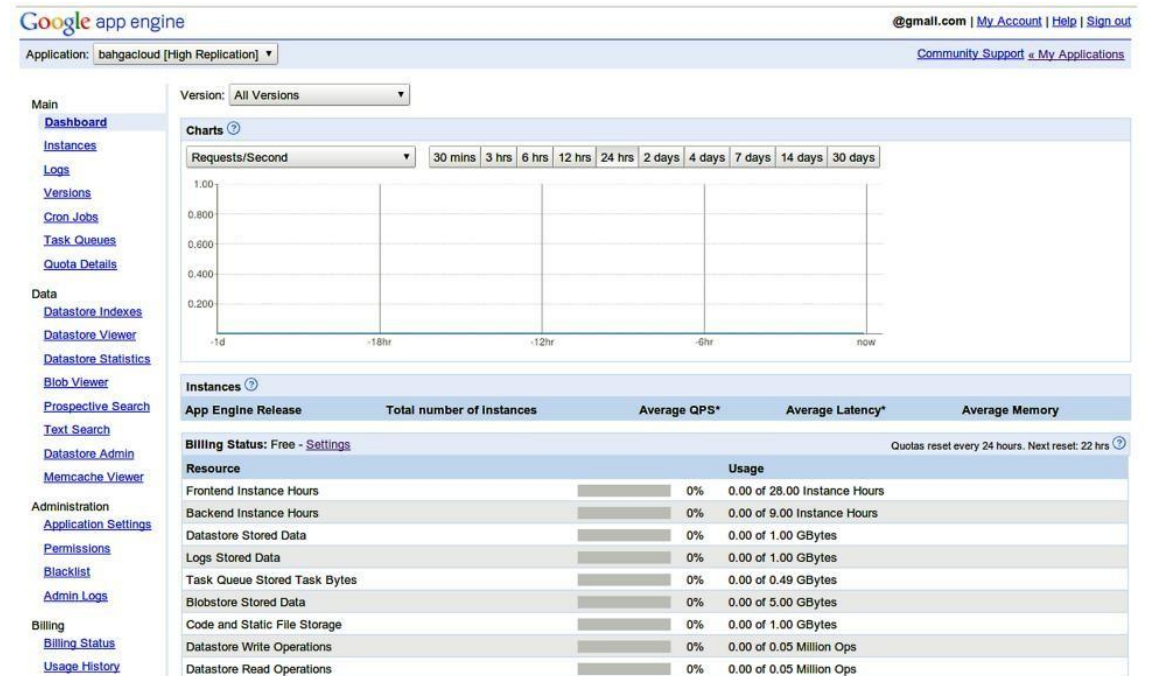


# Application Runtimes & Frameworks

- Cloud-based application runtimes and frameworks allow developers to develop and host applications in the cloud.
- Support for various programming languages
  - Application runtimes provide support for programming languages (e.g., Java, Python, or Ruby).
- Resource Allocation
  - Application runtimes automatically allocate resources for applications and handle the application scaling, without the need to run and maintain servers.

# Google App Engine

- Google App Engine is the platform-as-a-service (PaaS) from Google, which includes both an application runtime and web frameworks.
- Runtimes
  - App Engine provides runtime environments for Java, Python, PHP and Go programming language.
- Sandbox
  - Applications run in a secure sandbox environment isolated from other applications.
  - The sandbox environment provides a limited access to the underlying operating system.



# Google App Engine

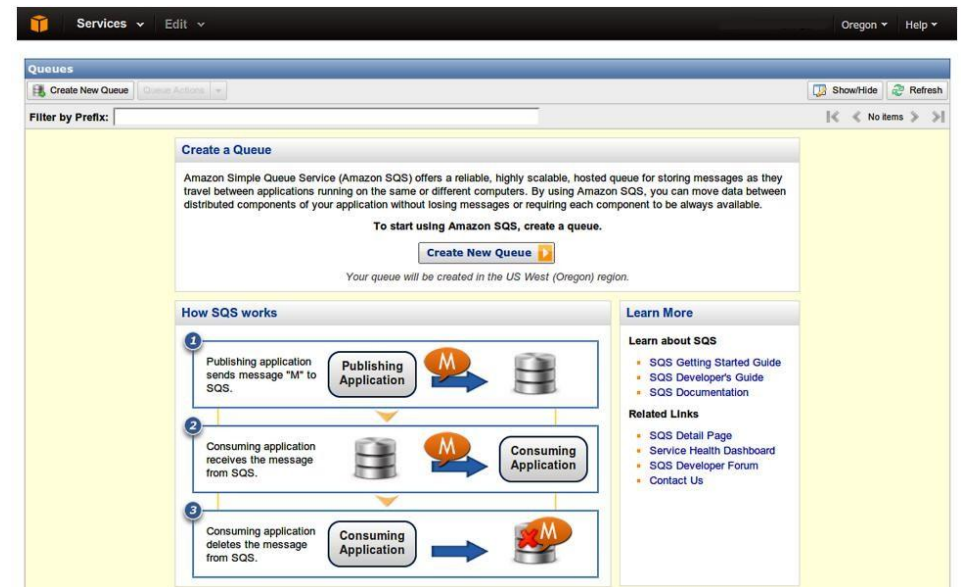
- Web Frameworks
  - App Engine provides a simple Python web application framework called webapp2. App Engine also supports any framework written in pure Python that speaks WSGI, including Django, CherryPy, Pylons, web.py, and web2py.
- Datastore
  - App Engine provides a no-SQL data storage service.
- Authentication
  - App Engine applications can be integrated with Google Accounts for user authentication.
- URL Fetch service
  - URL Fetch service allows applications to access resources on the Internet, such as web services or other data.
- Other services
  - Email service
  - Image Manipulation service
  - Memcache
  - Task Queues
  - Scheduled Tasks service

# Windows Azure Web Sites

- Windows Azure Web Sites is a Platform-as-a-Service (PaaS) from Microsoft.
- Azure Web Sites allows you to host web applications in the Azure cloud.
- Shared & Standard Options.
  - In the shared option, Azure Web Sites run on a set of virtual machines that may contain multiple web sites created by multiple users.
  - In the standard option, Azure Web Sites run on virtual machines (VMs) that belong to an individual user.
- Azure Web Sites supports applications created in ASP.NET, PHP, Node.js and Python programming languages.
- Multiple copies of an application can be run in different VMs, with Web Sites automatically load balancing requests across them.

# Queuing Services - Amazon Simple Queue Service

- Amazon Simple Queue Service (SQS) is a queuing service from Amazon.
- Short Messages
  - SQS is a distributed queue that supports messages of up to 256 KB in size.
- Multiple Writers/Readers
  - SQS supports multiple writers and readers and locks messages while they are being processed.
- High Availability
  - To ensure high availability for delivering messages, SQS service trade-offs on the first in, first out capability and does not guarantee that messages will be delivered in FIFO order.
  - Applications that require FIFO ordering of messages can place additional sequencing information in each message so that they can be re-ordered after retrieving from a queue.



# Queuing Services - Google Task Queue Service

- Google Task Queues service is a queuing service from Google and is a part of the Google App Engine platform.
- Task queues allow applications to execute tasks in background.
- Tasks
  - Task is a unit of work to be performed by an application. The task objects consist of application-specific URL with a request handler for the task, and an optional data payload that parameterizes the task.
- Push Queue
  - Push Queue is the default queue that processes tasks based on the processing rate configured in the queue definition.
- Pull Queue
  - Pull Queues allow task consumers to lease a specific number of tasks for a specific duration. The tasks are processed and deleted before the lease ends.

# Queuing Services - Windows Azure Queue Service

- Windows Azure Queue service is a queuing service from Microsoft.
- Azure Queue service allows storing large numbers of messages that can be accessed from anywhere in the world via authenticated calls using HTTP or HTTPS.
- Short Messages
  - The size of a single message can be up to 64KB.

# Email Services

- Cloud-based email services allow applications hosted in the cloud to send emails.
- Amazon Simple Email Service
  - Amazon Simple Email Service is bulk and transactional email-sending service from Amazon
  - SES is an outbound-only email-sending service that allows applications hosted in the Amazon cloud to send emails such as marketing emails, transactional emails and other types of correspondence
  - To ensure high email deliverability, SES uses content filtering technologies to scan the outgoing email messages
  - SES service can be accessed and used from the SES console, the Simple Mail Transfer Protocol (SMTP) interface, or the SES API
- Google Email Service
  - Google Email service is part of the Google App Engine platform that allows App Engine applications to send email messages on behalf of the app's administrators, and on behalf of users with Google Accounts.
  - App Engine apps can also receive emails. Apps send messages using the Mail service and receive messages in the form of HTTP requests initiated by App Engine and posted to the app.

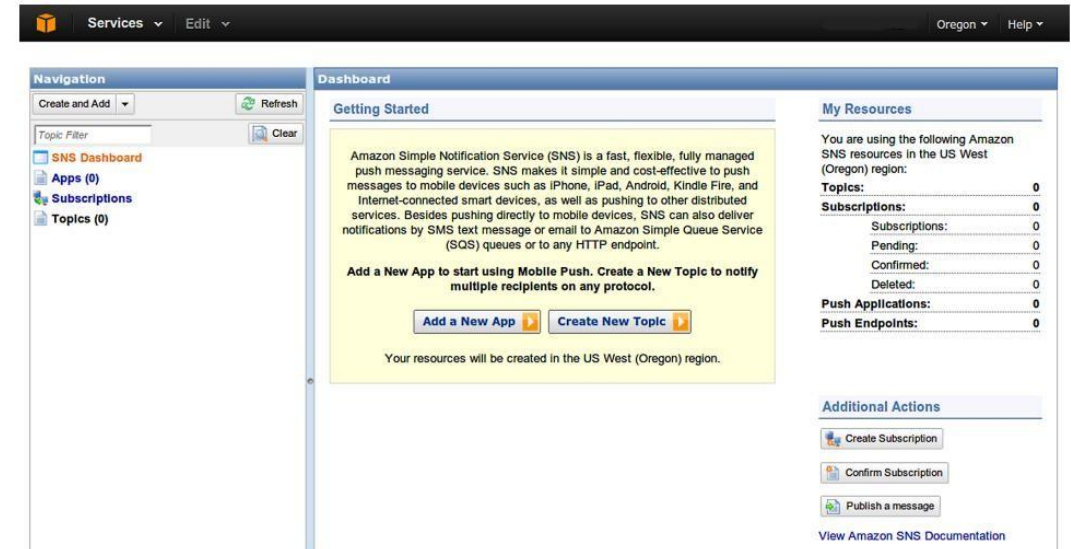


# Notification Services

- Cloud-based notification services or push messaging services allow applications to push messages to internet connected smart devices such as smartphones, tablets, etc.
- Push messaging services are based on publish-subscribe model in which consumers subscribe to various topics/channels provided by a publisher/producer.
- Whenever new content is available on one of those topics/channels, the notification service pushes that information out to the consumer.
- Push notifications are used for such smart devices as they help in displaying the latest information while remaining energy efficient.
- Consumer applications on such devices can increase their consumer engagement with the help of push notifications.

# Notification Services - Amazon Simple Notification Service

- Amazon Simple Notification Service is a push messaging service from Amazon.
- SNS has two types of clients:
  - Publishers
    - Publishers communicate asynchronously with subscribers by producing and sending messages to topics. A topic is a logical access point and a communication channel.
  - Subscribers.
    - Subscribers are the consumers who subscribe to topics to receive notifications.
- SNS can deliver notifications as SMS, email, or to SQS queues, or any HTTP endpoint.



# Google Cloud Messaging

- Google Cloud Messaging for Android provides push messaging for Android devices.
- GCM allows applications to send data from the application servers to their users' Android devices, and also to receive messages from devices on the same connection.
- Notifying Android Apps
  - GCM is useful for notifying applications on Android devices that there is new data to be fetched from the application servers.
- Short Messages
  - GCM supports messages with payload data upto 4 KB.
- Send-to-Sync
  - GCM provides a 'send-to-sync' message capability that can be used to inform an application to sync data from the server.
- GCM for Chrome
  - Google Cloud Messaging for Chrome is another notification service from Google that allows messages to be delivered from the cloud to apps and extensions running in Chrome.

# Windows Azure Notification Hubs

- Windows Azure Notification Hubs is a push notification service from Microsoft.
- Common Interface
  - Provides a common interface to send notifications to all major mobile platforms including Windows Store/Windows Phone 8, iOS, and Android.
- Platform Notification Systems
  - Platform specific infrastructures called Platform Notification Systems (PNS) are used to deliver notification messages.
  - Devices register their PNS handles with the Notification Hub.
  - Each notification hub contains credentials for each supported PNS.
  - These credentials are used to connect to the PNSs and send push notifications to the applications.

# Media Services

- Cloud service providers provide various types of media services that can be used by applications for manipulating, transforming or transcoding media such as images, videos, etc.
- Amazon Elastic Transcoder
  - Amazon Elastic Transcoder is a cloud-based video transcoding service from Amazon.
  - Elastic Transcoder can be used to convert video files from their source format into various other formats that can be played on devices such as desktops, mobiles, tablets, etc.
- Google Images Manipulation Service
  - Google Images Manipulation service is a part of the Google App Engine platform. Image Manipulation service provides the capability to resize, crop, rotate, flip and enhance images.
- Windows Azure Media Services
  - Windows Azure Media Services provides the various media services such as encoding & format conversion, content protection and on-demand and live streaming capabilities.

# Content Delivery Services

- Cloud-based content delivery services include Content Delivery Networks (CDNs).
- CDN is a distributed system of servers located across multiple geographic locations to serve content to end-users with high availability and high performance.
- CDNs are useful for serving static content such as text, images, scripts, etc., and streaming media.
- CDNs have a number of edge locations deployed in multiple locations, often over multiple backbones.
- Requests for static or streaming media content that is served by a CDN are directed to the nearest edge location.
- **Amazon Cloud Front**
  - Amazon Cloud Front is a content delivery service from Amazon. CloudFront can be used to deliver dynamic, static and streaming content using a global network of edge locations.
- **Windows Azure Content Delivery Network**
  - Windows Azure Content Delivery Network (CDN) is the content delivery service from Microsoft.

# Analytics Services

- Cloud-based analytics services allow analyzing massive data sets stored in the cloud either in cloud storages or in cloud databases using programming models such as MapReduce.
- Amazon Elastic MapReduce
  - Amazon Elastic MapReduce is the MapReduce service from Amazon based the Hadoop framework running on Amazon EC2 and S3
  - EMR supports various job types such as Custom JAR, Hive program, Streaming job, Pig programs and Hbase
- Google MapReduce Service
  - Google MapReduce Service is a part of the App Engine platform and can be accessed using the Google MapReduce API.
- Google BigQuery
  - Google BigQuery is a service for querying massive datasets. BigQuery allows querying datasets using SQL-like queries.
- Windows Azure HDInsight
  - Windows Azure HDInsight is an analytics service from Microsoft. HDInsight deploys and provisions Hadoop clusters in the Azure cloud and makes Hadoop available as a service.

# Deployment & Management Services

- Cloud-based deployment & management services allow you to easily deploy and manage applications in the cloud. These services automatically handle deployment tasks such as capacity provisioning, load balancing, auto-scaling, and application health monitoring.
- Amazon Elastic Beanstalk
  - Amazon provides a deployment service called Elastic Beanstalk that allows you to quickly deploy and manage applications in the AWS cloud.
  - Elastic Beanstalk supports Java, PHP, .NET, Node.js, Python, and Ruby applications.
  - With Elastic Beanstalk you just need to upload the application and specify configuration settings in a simple wizard and the service automatically handles instance provisioning, server configuration, load balancing and monitoring.
- Amazon CloudFormation
  - Amazon CloudFormation is a deployment management service from Amazon.
  - With CloudFormation you can create deployments from a collection of AWS resources such as Amazon Elastic Compute Cloud, Amazon Elastic Block Store, Amazon Simple Notification Service, Elastic Load Balancing and Auto Scaling.
  - A collection of AWS resources that you want to manage together are organized into a stack.

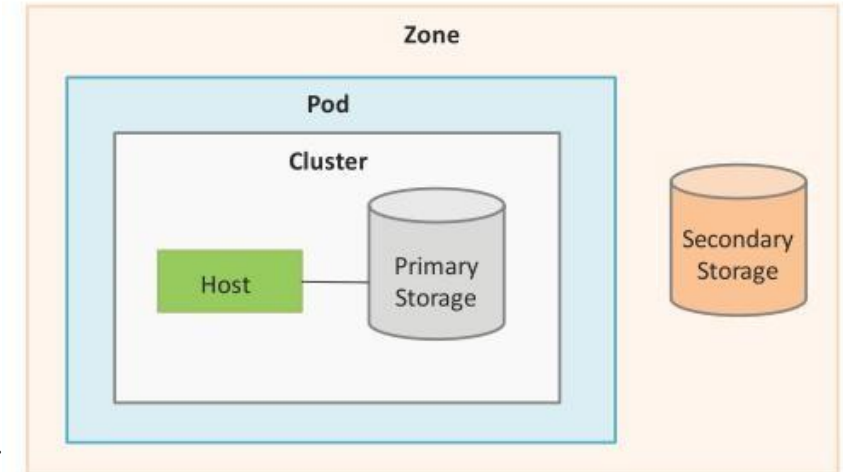


# Identity & Access Management Services

- Identity & Access Management (IDAM) services allow managing the authentication and authorization of users to provide secure access to cloud resources.
- Using IDAM services you can manage user identifiers, user permissions, security credentials and access keys.
- Amazon Identity & Access Management
  - AWS Identity and Access Management (IAM) allows you to manage users and user permissions for an AWS account.
  - With IAM you can manage users, security credentials such as access keys, and permissions that control which AWS resources users can access.
  - Using IAM you can control what data users can access and what resources users can create.
  - IAM also allows you to control creation, rotation, and revocation security credentials of users.
- Windows Azure Active Directory
  - Windows Azure Active Directory is an Identity & Access Management Service from Microsoft.
  - Azure Active Directory provides a cloud-based identity provider that easily integrates with your on-premises active directory deployments and also provides support for third party identity providers.
  - With Azure Active Directory you can control access to your applications in Windows Azure.

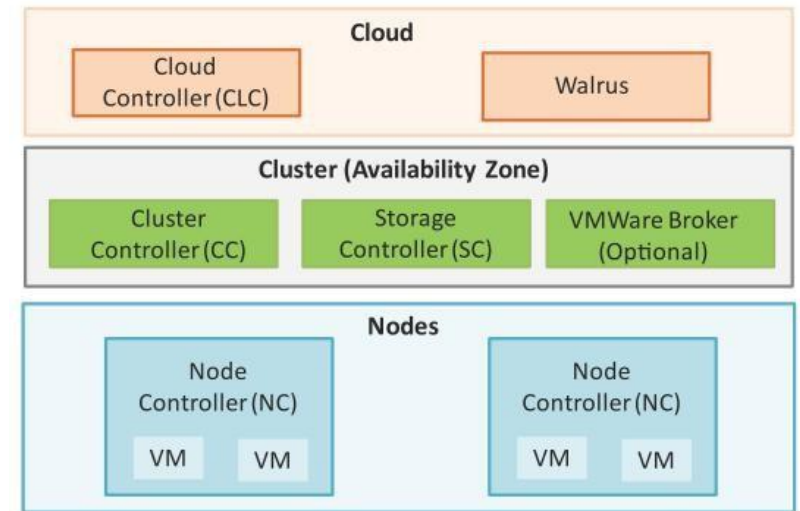
# Open Source Private Cloud Software - CloudStack

- Apache Cloud Stack is an open source cloud software that can be used for creating private cloud offerings.
- Cloud Stack manages the network, storage, and compute nodes that make up a cloud infrastructure.
- A CloudStack installation consists of a Management Server and the cloud infrastructure that it manages.
- Zones
  - The Management Server manages one or more zones where each zone is typically a single datacenter.
- Pods
  - Each zone has one or more pods. A pod is a rack of hardware comprising of a switch and one or more clusters.
- Cluster
  - A cluster consists of one or more hosts and a primary storage. A host is a compute node that runs guest virtual machines.
- Primary Storage
  - The primary storage of a cluster stores the disk volumes for all the virtual machines running on the hosts in that cluster.
- Secondary Storage
  - Each zone has a secondary storage that stores templates, ISO images, and disk volume snapshots.



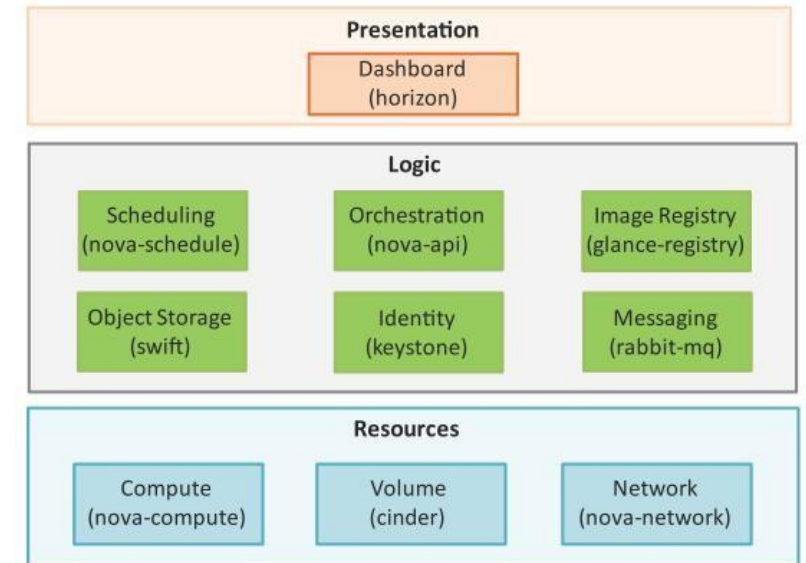
# Open Source Private Cloud Software - Eucalyptus

- Apache Cloud Stack is an open source cloud software that can be used for creating private cloud offerings.
- CloudStack manages the network, storage, and compute nodes that make up a cloud infrastructure.
- A CloudStack installation consists of a Management Server and the cloud infrastructure that it manages.
- Zones
  - The Management Server manages one or more zones where each zone is typically a single datacenter.
- Pods
  - Each zone has one or more pods. A pod is a rack of hardware comprising of a switch and one or more clusters.
- Cluster
  - A cluster consists of one or more hosts and a primary storage. A host is a compute node that runs guest virtual machines.
- Primary Storage
  - The primary storage of a cluster stores the disk volumes for all the virtual machines running on the hosts in that cluster.
- Secondary Storage
  - Each zone has a secondary storage that stores templates, ISO images, and disk volume snapshots.



# Open Source Private Cloud Software - OpenStack

- Eucalyptus is an open source private cloud software for building private and hybrid clouds that are compatible with Amazon Web Services (AWS) APIs.
- Node Controller
  - NC hosts the virtual machine instances and manages the virtual network endpoints.
- The cluster-level (availability-zone) consists of three components
  - Cluster Controller - which manages the virtual machines and is the front-end for a cluster.
  - Storage Controller - which manages the Eucalyptus block volumes and snapshots to the instances within its specific cluster. SC is equivalent to AWS Elastic Block Store (EBS).
  - VMWare Broker - which is an optional component that provides an AWS-compatible interface for VMware environments.
- At the cloud-level there are two components:
  - Cloud Controller - which provides an administrative interface for cloud management and performs high-level resource scheduling, system accounting, authentication and quota management.
  - Walrus - which is equivalent to Amazon S3 and serves as a persistent storage to all of the virtual machines in the Eucalyptus cloud. Walrus can be used as a simple Storage-as-a-Service



# Further Reading

- Amazon Elastic Compute Cloud, <http://aws.amazon.com/ec2>
- Google Compute Engine, <https://developers.google.com/compute/>
- Windows Azure, <http://www.windowsazure.com/>
- Google App Engine, <http://appengine.google.com>
- Google App Engine, <https://developers.google.com/appengine/>
- Google Cloud Storage, <https://developers.google.com/storage/>
- Google BigQuery, <https://developers.google.com/bigquery/>
- Google Cloud Datastore, <http://developers.google.com/datastore/>
- Google Cloud SQL, <https://developers.google.com/cloud-sql/>
- CloudStack, <http://cloudstack.apache.org>
- Eucalyptus, <http://www.eucalyptus.com>
- OpenStack, <http://www.openstack.org>

# UNIT-4

## **CLOUD APPLICATION DESIGN & BENCHMARKING**

# Outline

- Cloud Application Design Considerations
- Cloud Application Reference Architectures
- Design Methodologies
- Data Storage
- Data Analytics
- Deployment & Management

# Design Considerations for Cloud Applications

- **Scalability**

- Scalability is an important factor that drives the application designers to move to cloud computing environments. Building applications that can serve millions of users without taking a hit on their performance has always been challenging. With the growth of cloud computing application designers can provision adequate resources to meet their workload levels.
  - Loosely coupling of components
  - Asynchronous Communication
  - Stateless Design
  - Database Choice and Design

- **Reliability & Availability**

Reliability of a system is defined as the probability that a system will perform the intended functions under stated conditions for a specified amount of time. Availability is the probability that a system will perform a specified function under given conditions at a prescribed time.

- No Single point of failure
- Trigger automated action on failures
- Graceful Degradation
- Logging
- Replication



# Design Considerations for Cloud Applications

- **Security**

Security is an important design consideration for cloud applications given the outsourced nature of cloud computing environments.

- Securing data at rest
- Securing data in motion
- Authentication
- Identity and access management
- Key Management
- Data integrity
- Auditing.

- **Maintenance & Upgradation**

To achieve a rapid time-to-market, businesses typically launch their applications with a core set of features ready and then incrementally add new features as and when they are complete. In such scenarios, it is important to design applications with low maintenance and upgradation costs.

- **Performance**

Applications should be designed while keeping the performance requirements in mind.

# Reference Architectures – e-Commerce, Business-to-Business, Banking and Financial apps

- **Load Balancing Tier**

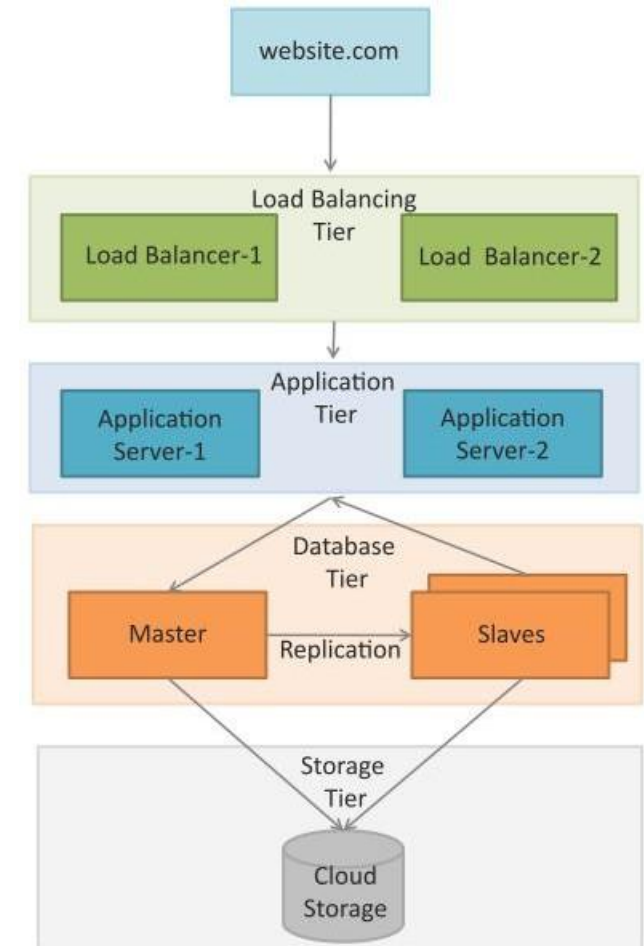
- Load balancing tier consists of one or more load balancers.

- **Application Tier**

- For this tier, it is recommended to configure auto scaling.
- Auto scaling can be triggered when the recorded values for any of the specified metrics such as CPU usage, memory usage, etc. goes above defined thresholds.

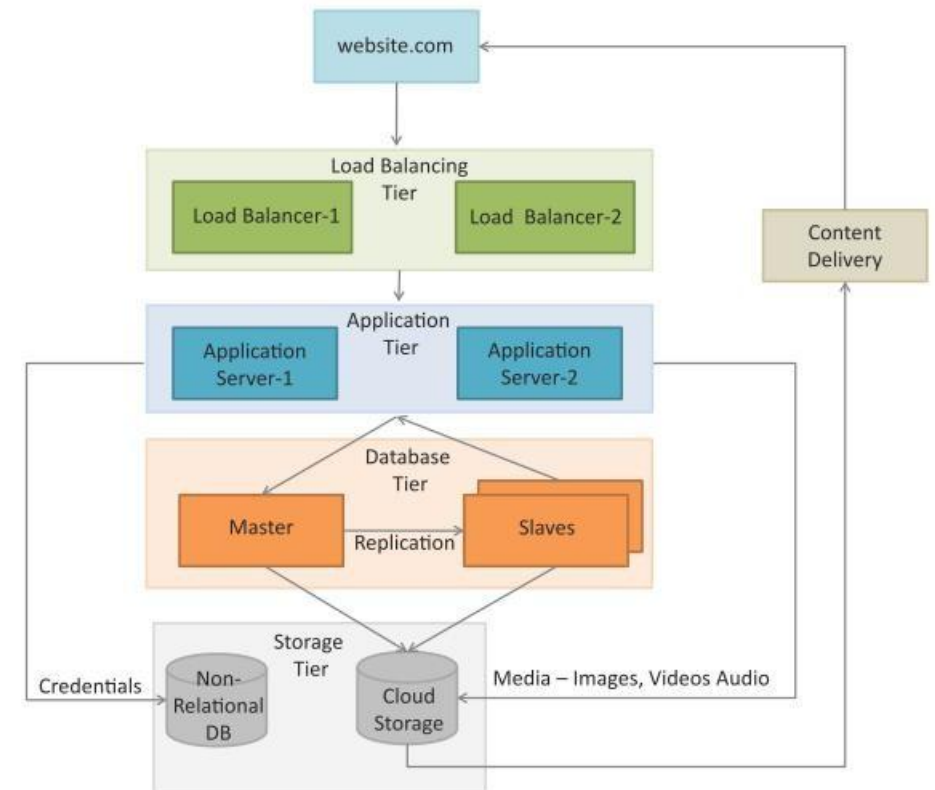
- **Database Tier**

- The database tier includes a master database instance and multiple slave instances.
- The master node serves all the write requests and the read requests are served from the slave nodes.
- This improves the throughput for the database tier since most applications have a higher number of read requests than write requests.



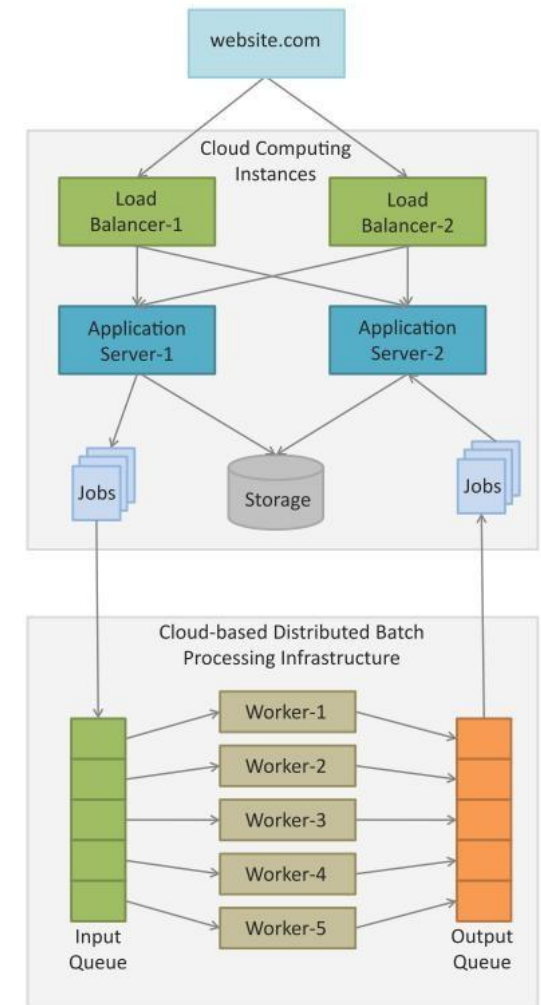
# Reference Architectures – Content delivery apps

- Figure shows a typical deployment architecture for content delivery applications such as online photo albums, video webcasting, etc.
- Both relational and non-relational data stores are shown in this deployment.
- A content delivery network (CDN) which consists of a global network of edge locations is used for media delivery.
- CDN is used to speed up the delivery of static content such as images and videos.



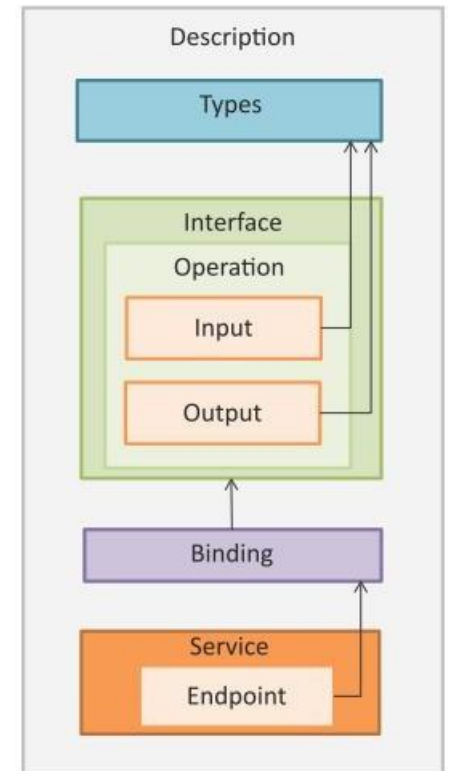
# Reference Architectures – Analytics apps

- Figure shows a typical deployment architecture for compute intensive applications such as Data Analytics, Media Transcoding, etc.
- Comprises of web, application, storage, computing/analytics and database tiers.
- The analytics tier consists of cloud-based distributed batch processing frameworks such as Hadoop which are suitable for analyzing big data.
- Data analysis jobs (such as MapReduce) jobs are submitted to the analytics tier from the application servers.
- The jobs are queued for execution and upon completion the analyzed data is presented from the application servers.



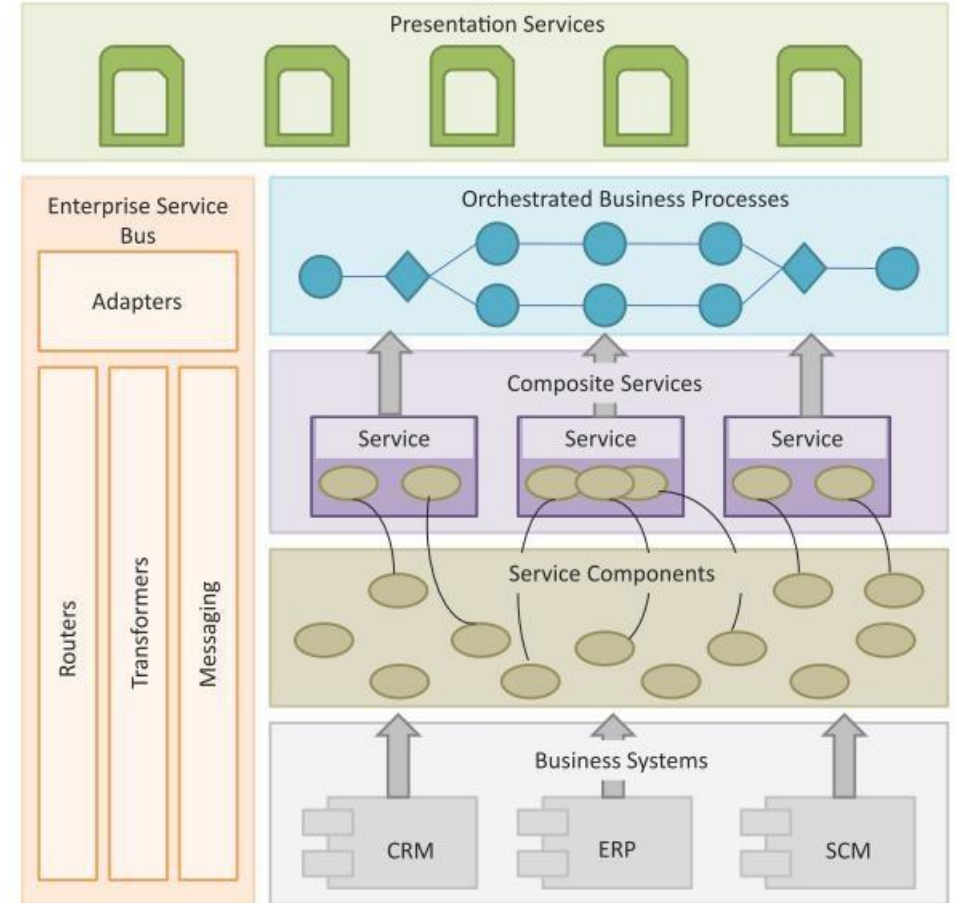
# Service Oriented Architecture

- Service Oriented Architecture (SOA) is a well established architectural approach for designing and developing applications in the form services that can be shared and reused.
- SOA is a collection of discrete software modules or services that form a part of an application and collectively provide the functionality of an application.
- SOA services are developed as loosely coupled modules with no hard-wired calls embedded in the services.
- The services communicate with each other by passing messages.
- Services are described using the Web Services Description Language (WSDL).
- WSDL is an XML-based web services description language that is used to create service descriptions containing information on the functions performed by a service and the inputs and outputs of the service.



# SOA Layers

- **Business Systems**
  - This layer consists of custom built applications and legacy systems such as Enterprise Resource Planning (ERP), Customer Relationship Management (CRM), Supply Chain Management (SCM), etc.
- **Service Components**
  - The service components allow the layers above to interact with the business systems. The service components are responsible for realizing the functionality of the services exposed.
- **Composite Services**
  - These are coarse-grained services which are composed of two or more service components. Composite services can be used to create enterprise scale components or business-unit specific components.
- **Orchestrated Business Processes**
  - Composite services can be orchestrated to create higher level business processes. In this layers the compositions and orchestrations of the composite services are defined to create business processes.
- **Presentation Services**
  - This is the topmost layer that includes user interfaces that exposes the services and the orchestrated business processes to the users.
- **Enterprise Service Bus**
  - This layer integrates the services through adapters, routing, transformation and messaging mechanisms.

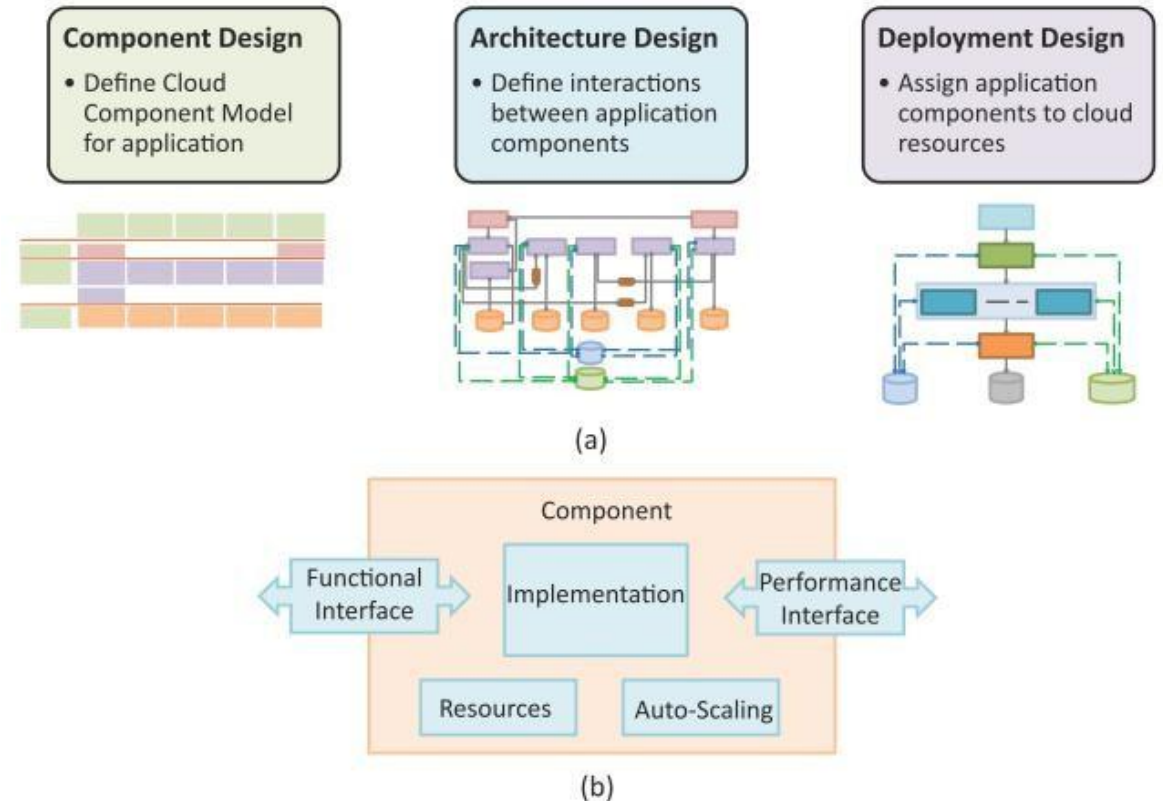


# Cloud Component Model

- Cloud Component Model is an application design methodology that provides a flexible way of creating cloud applications in a rapid, convenient and platform independent manner.
- CCM is an architectural approach for cloud applications that is not tied to any specific programming language or cloud platform.
- Cloud applications designed with CCM approach can have innovative hybrid deployments in which different components of an application can be deployed on cloud infrastructure and platforms of different cloud vendors.
- Applications designed using CCM have better portability and interoperability.
- CCM based applications have better scalability by decoupling application components and providing asynchronous

# CCM Application Design Methodology

- CCM approach for application design involves:
  - Component Design
  - Architecture Design
  - Deployment Design





# CCM Component Design

- Cloud Component Model is created for the application based on comprehensive analysis of the application's functions and building blocks.
- Cloud component model allows identifying the building blocks of a cloud application which are classified based on the functions performed and type of cloud resources required.
- Each building block performs a set of actions to produce the desired outputs for other components.
- Each component takes specific inputs, performs a pre-defined set of actions and produces the desired outputs.
- Components offer their functions as services through a functional interface which can be used by other components.
- Components report their performance to a performance database through a performance interface.



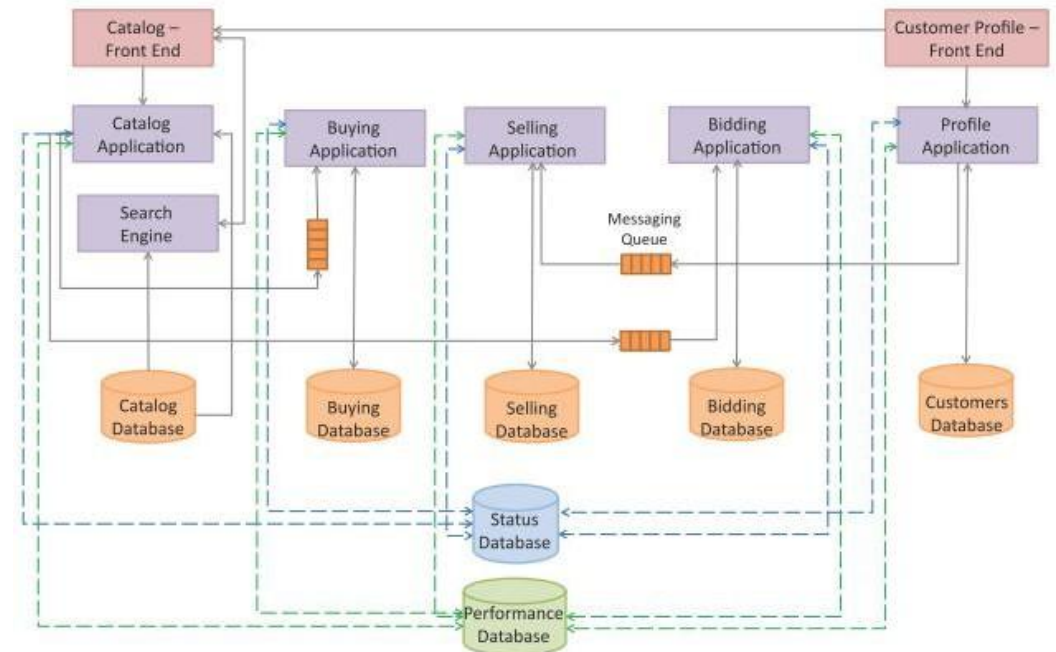
CCM map for an e-Commerce application

# CCM Architecture Design

- In Architecture Design step, interactions between the application components are defined.
- **CCM components have the following characteristics:**
  - **Loose Coupling**
    - Components in the Cloud Component Model are loosely coupled.
  - **Asynchronous Communication**
    - By allowing asynchronous communication between components, it is possible to add capacity by adding additional servers when the application load increases. Asynchronous communication is made possible by using messaging queues.
  - **Stateless**
  - **Design**

Components in the Cloud Component Model are stateless.

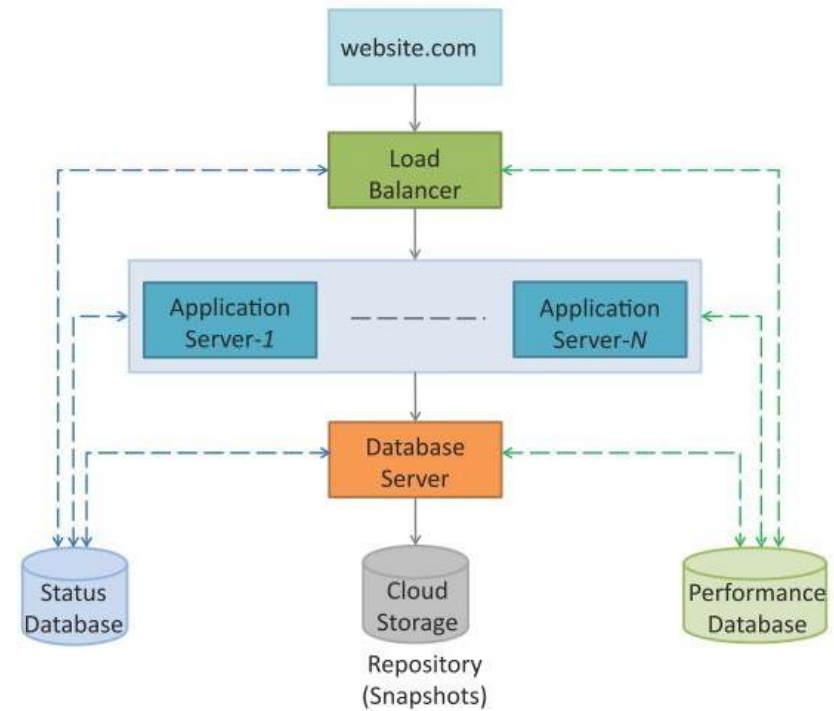
By storing session state outside of the component (e.g. in a database), stateless component design enables distribution and horizontal scaling.



**Figure: Architecture** design of an e-Commerce application.

# CCM Deployment Design

- In Deployment Design step, application components are mapped to specific cloud resources such as web servers, application servers, database servers, etc.
- Since the application components are designed to be loosely coupled and stateless with asynchronous communication, components can be deployed independently of each other.
- This approach makes it easy to migrate application components from one cloud to the other.
- With this flexibility in application design and deployment, the application developers can ensure that the applications meet the performance and cost requirements with changing contexts.



Deployment design of an e-Commerce application

# SOA vs CCM

## Similarities

	SOA	CCM
Standardization & Re-use	SOA advocates principles of reuse and well defined relationship between service provider and service consumer.	CCM is based on reusable components which can be used by multiple cloud applications.
Loose coupling	SOA is based on loosely coupled services that minimize dependencies.	CCM is based on loosely coupled components that communicate asynchronously
Statelessness	SOA services minimize resource consumption by deferring the management of state information.	CCM components are stateless. State is stored outside of the components.

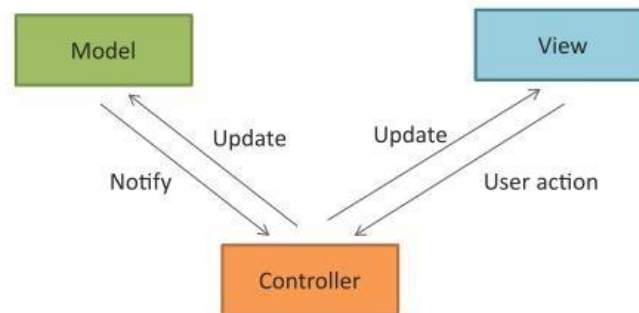
# SOA vs CCM

## Differences

	SOA	CCM
End points	SOA services have small and well-defined set of endpoints through which many types of data can pass.	CCM components have very large number of endpoints. There is an endpoint for each resource in a component, identified by a URI.
Messaging	SOA uses a messaging layer above HTTP by using SOAP which provide prohibitive constraints to developers.	CCM components use HTTP and REST for messaging.
Security	Uses WS-Security , SAML and other standards for security	CCM components use HTTPS for security.
Interfacing	SOA uses XML for interfacing.	CCM allows resources in components represent different formats for interfacing (HTML, XML, JSON, etc.).
Consumption	Consuming traditional SOA services in a browser is cumbersome.	CCM components and the underlying component resources are exposed as XML, JSON (and other formats) over HTTP or REST, thus easy to consume in the browser.

# Model View Controller

- Model View Controller (MVC) is a popular software design pattern for web applications.
- Model
  - Model manages the data and the behavior of the applications. Model processes events sent by the controller. Model has no information about the views and controllers. Model responds to the requests for information about its state (from the view) and responds to the instructions to change state (from controller).
- View
  - View prepares the interface which is shown to the user. Users interact with the application through views. Views present the information that the model or controller tell the view to present to the user and also handle user requests and sends them to the controller.
- Controller
  - Controller glues the model to the view. Controller processes user requests and updates the model when the user manipulates the view. Controller also updates the view when the model changes.



# RESTful Web Services

- Representational State Transfer (REST) is a set of architectural principles by which you can design web services and web APIs that focus on a system's resources and how resource states are addressed and transferred.
- The REST architectural constraints apply to the components, connectors, and data elements, within a distributed hypermedia system.
- A RESTful web service is a web API implemented using HTTP and REST principles.
- The REST architectural constraints are as follows:
  - Client-Server
  - Stateless
  - Cacheable
  - Layered System
  - Uniform Interface
  - Code on demand

# Relational Databases

- A relational database is database that conforms to the relational model that was popularized by Edgar Codd in 1970.
- The 12 rules that Codd introduced for relational databases include:
  - Information rule
  - Guaranteed access rule
  - Systematic treatment of null values
  - Dynamic online catalog based on relational model
  - Comprehensive sub-language rule
  - View updating rule
  - High level insert, update, delete
  - Physical data independence
  - Logical data independence
  - Integrity independence
  - Distribution independence
  - Non-subversion rule



# Relational Databases

- **Relations**

- A relational database has a collection of relations (or tables). A relation is a set of tuples (or rows).

- **Schema**

- Each relation has a fixed schema that defines the set of attributes (or columns in a table) and the constraints on the attributes.

- **Tuples**

- Each tuple in a relation has the same attributes (columns). The tuples in a relation can have any order and the relation is not sensitive to the ordering of the tuples.

- **Attributes**

- Each attribute has a domain, which is the set of possible values for the attribute.

- **Insert/Update/Delete**

- Relations can be modified using insert, update and delete operations. Every relation has a primary key that uniquely identifies each tuple in the relation.

- **Primary Key**

- An attribute can be made a primary key if it does not have repeated values in different tuples.

# ACID Guarantees

- Relational databases provide ACID guarantees.
- **Atomicity**
  - Atomicity property ensures that each transaction is either “all or nothing”. An atomic transaction ensures that all parts of the transaction complete or the database state is left unchanged.
- **Consistency**
  - Consistency property ensures that each transaction brings the database from one valid state to another. In other words, the data in a database always conforms to the defined schema and constraints.
- **Isolation**
  - Isolation property ensures that the database state obtained after a set of concurrent transactions is the same as would have been if the transactions were executed serially. This provides concurrency control, i.e. the results of incomplete transactions are not visible to other transactions. The transactions are isolated from each other until they finish.
- **Durability**
  - Durability property ensures that once a transaction is committed, the data remains as it is, i.e. it is not affected by system outages such as power loss. Durability guarantees that the database can keep track of changes and can recover from abnormal terminations.

# Non-Relational Databases

- Non-relational databases (or popularly called No-SQL databases) are becoming popular with the growth of cloud computing.
- Non-relational databases have better horizontal scaling capability and improved performance for big data at the cost of less rigorous consistency models.
- Unlike relational databases, non-relational databases do not provide ACID guarantees.
- Most non-relational databases offer “eventual” consistency, which means that given a sufficiently long period of time over which no updates are made, all updates can be expected to propagate eventually through the system and the replicas will be consistent.
- The driving force behind the non-relational databases is the need for databases that can achieve high scalability, fault tolerance and availability.
- These databases can be distributed on a large cluster of machines. Fault tolerance is provided by storing multiple replicas of data on different machines.

# Non-Relational Databases - Types

- **Key-value store**

- Key-value store databases are suited for applications that require storing unstructured data without a fixed schema. Most key-value stores have support for native programming language data types.

- **Document store**

- Document store databases store semi-structured data in the form of documents which are encoded in different standards such as JSON, XML, BSON, YAML, etc.

- **Graph store**

- Graph stores are designed for storing data that has graph structure (nodes and edges). These solutions are suitable for applications that involve graph data such as social networks, transportation systems, etc.

- **Object store**

- Object store solutions are designed for storing data in the form of objects defined in an object-oriented programming language.

# Unit - 5

## **Cloud Security**

# Cloud Security

- Cloud security challenges
- Authorization
- Authentication
- Identify & Access Management
- Data Security
- Data Integrity
- Encryption & Key Management

# Cloud Security Challenges

- **Authentication**
  - Authentication refers to digitally confirming the identity of the entity requesting access to some protected information.
  - In a traditional in-house IT environment authentication policies are under the control of the organization. However, in cloud computing environments, where applications and data are accessed over the internet, the complexity of digital authentication mechanisms increases rapidly.
- **Authorization**
  - Authorization refers to digitally specifying the access rights to the protected resources using access policies.
  - In a traditional in-house IT environment, the access policies are controlled by the organization and can be altered at their convenience.
  - Authorization in a cloud computing environment requires the use of the cloud service providers services for specifying the access policies.
- **Security of data at rest**
  - Due to the multi-tenant environments used in the cloud, the application and database servers of different applications belonging to different organizations can be provisioned side-by-side increasing the complexity of securing the data.
  - Appropriate separation mechanisms are required to ensure the isolation between applications and data from different organizations.

# Cloud Security Challenges

- Security of data in motion

- In traditional in-house IT environments all the data exchanged between the applications and users remains within the organization's control and geographical boundaries.
- With the adoption of the cloud model, the applications and the data are moved out of the in-house IT infrastructure to the cloud provider.
- Therefore, appropriate security mechanisms are required to ensure the security of data in, and while in, motion.

- Data Integrity

- Data integrity ensures that the data is not altered in an unauthorized manner after it is created, transmitted or stored. Due to the outsourcing of data storage in cloud computing environments, ensuring integrity of data is important.

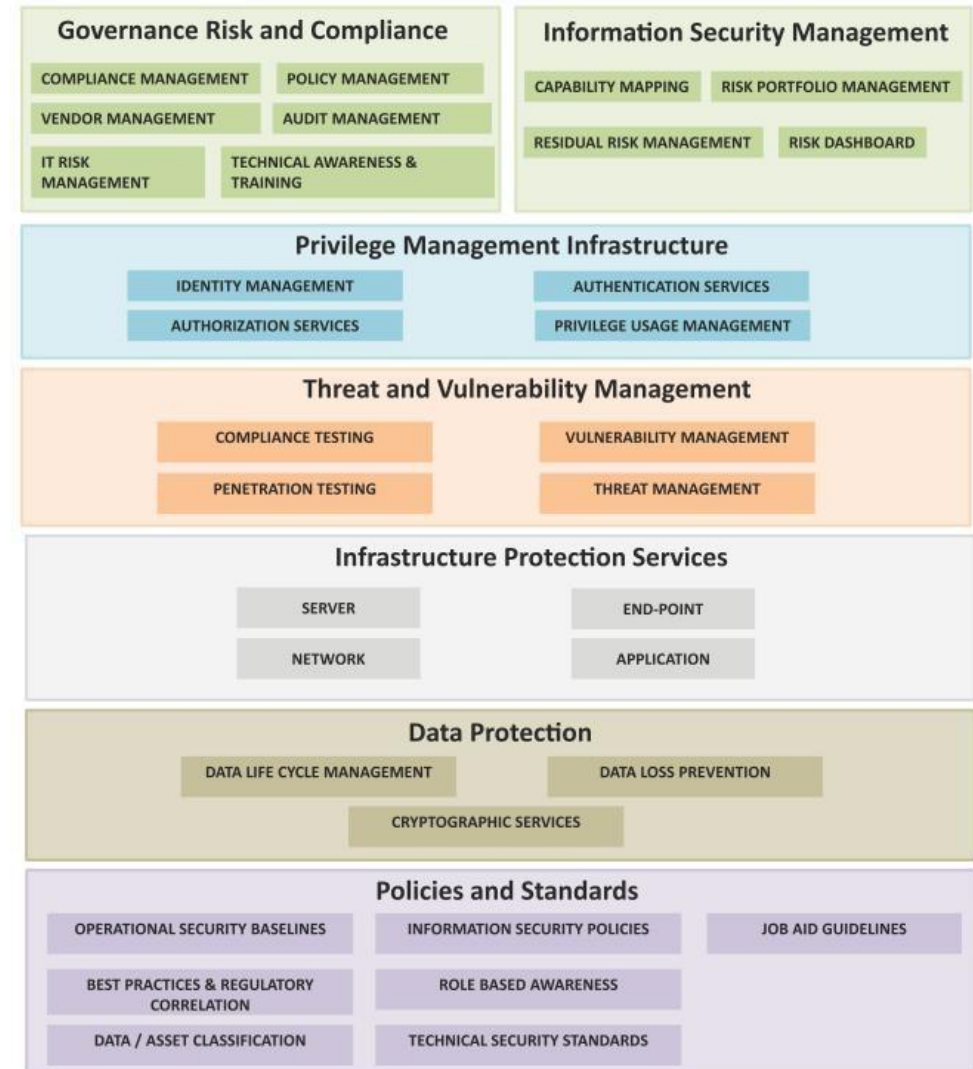
- Auditing

- Auditing is very important for applications deployed in cloud computing environments.
- In traditional in-house IT environments, organizations have complete visibility of their applications and accesses to the protected information.
- For cloud applications appropriate auditing mechanisms are required to get visibility into the application, data accesses and actions performed by the application users, including mobile users and devices such as wireless laptops and smartphones.



# CSA Cloud Security Architecture

- Cloud Security Alliance (CSA) provides a Trusted Cloud Initiative (TCI) Reference Architecture.
- TCI is a methodology and a set of tools that enable cloud application developers and security architects to assess where their internal IT and their cloud providers are in terms of security capabilities, and to plan a roadmap to meet the security needs of their business.
- Security and Risk Management (SRM) domain within the TCI Reference includes:
  - Governance, Risk Management, and Compliance
  - Information Security Management
  - Privilege Management Infrastructure
  - Threat and Vulnerability Management
  - Infrastructure Protection Services
  - Data Protection
  - Policies and Standards



# Authentication

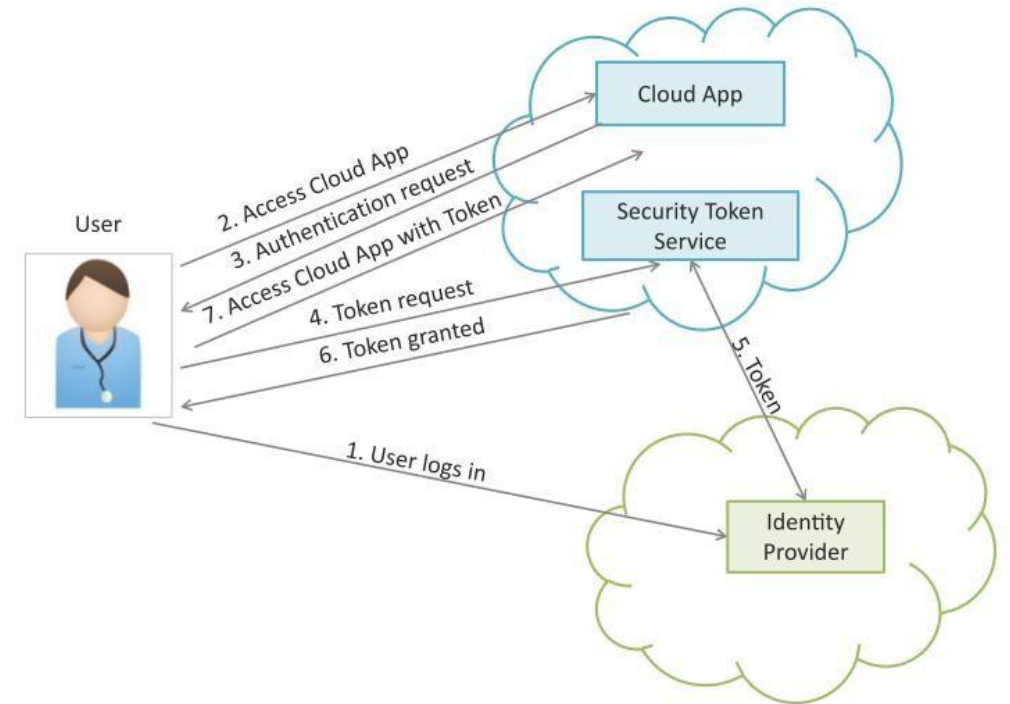
- Authentication refers to confirming the digital identity of the entity requesting access to some protected information.
- The process of authentication involves, but is not limited to, validating the at least one factor of identification of the entity to be authenticated.
- A factor can be something the entity or the user knows (password or pin), something the user has (such as a smart card), or something that can uniquely identify the user (such as fingerprints).
- In multifactor authentication more than one of these factors are used for authentication.
- There are various mechanisms for authentication including:
  - SSO
  - SAML-Token
  - OTP

# Single Sign-on (SSO)

- Single Sign-on (SSO) enables users to access multiple systems or applications after signing in only once, for the first time.
- When a user signs in, the user identity is recognized and there is no need to sign in again and again to access related systems or applications.
- Since different systems or applications may be internally using different authentication mechanisms, SSO upon receiving initial credential translates to different credentials for different systems or applications.
- The benefit of using SSO is that it reduces human error and saves time spent in authenticating with different systems or applications for the same identity.
- There are different implementation mechanisms:
  - SAML-Token
  - Kerberos

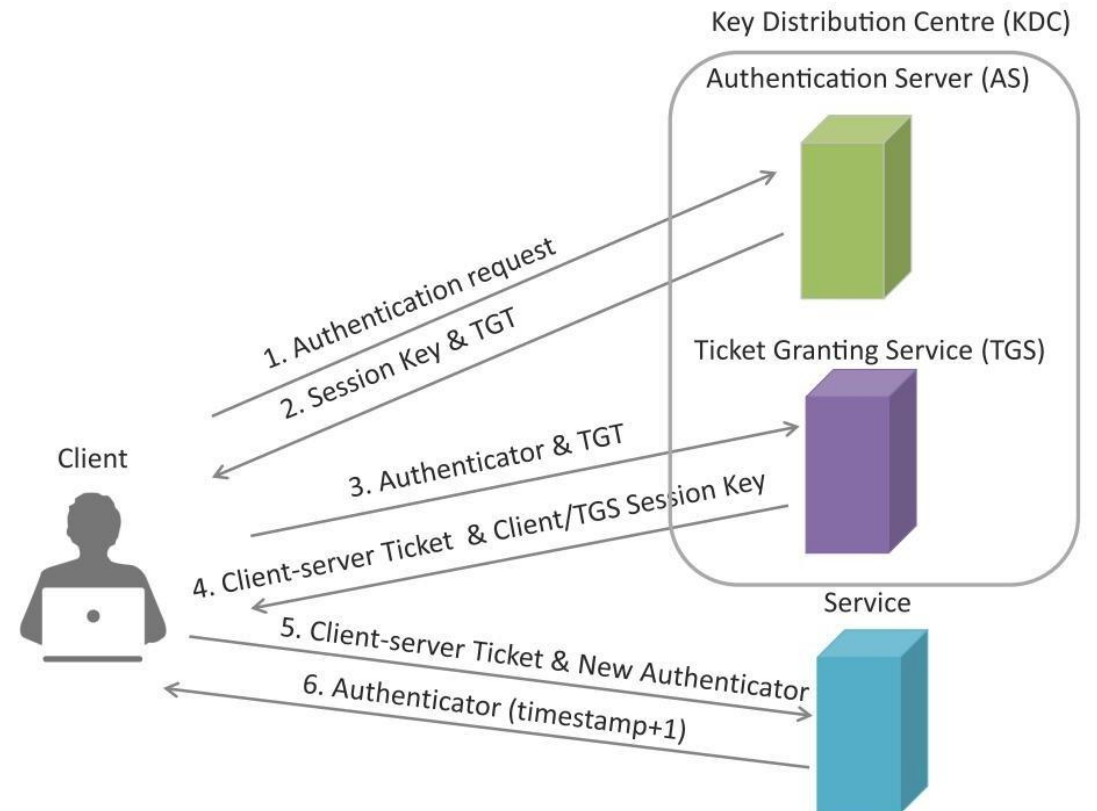
# SAML-Token

- Security Assertion Markup Language (SAML) is an XML-based open standard data format for exchanging security information (authentication and authorization data) between an identity provider and a service provider.
- SAML-token based SSO authentication
  - When a user tries to access the cloud application, a SAML request is generated and the user is redirected to the identity provider.
  - The identity provider parses the SAML request and authenticates the user. A SAML token is returned to the user, who then accesses the cloud application with the token.
  - SAML prevents man-in-the-middle and replay attacks by requiring the use of SSL encryption when transmitting assertions and messages.
  - SAML also provides a digital signature mechanism that enables the assertion to have a validity time range to prevent replay attacks.



# Kerberos

- Kerberos is an open authentication protocol that was developed At MIT.
- Kerberos uses tickets for authenticating client to a service that communicate over an un-secure network.
- Kerberos provides mutual authentication, i.e. both the client and the server authenticate with each other.

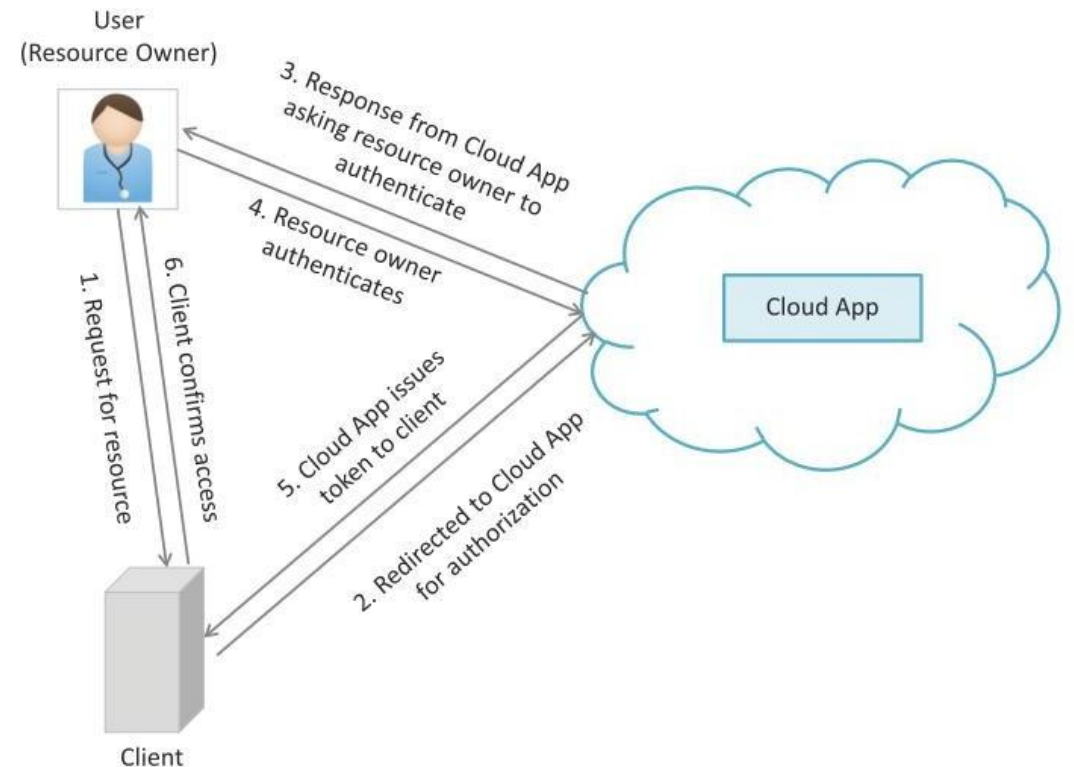


# One Time Password (OTP)

- One time password is another authentication mechanism that uses passwords which are valid for single use only for a single transaction or session.
- Authentication mechanism based on OTP tokens are more secure because they are not vulnerable to replay attacks.
- Text messaging (SMS) is the most common delivery mode for OTP tokens.
- The most common approach for generating OTP tokens is time synchronization.
- Time-based OTP algorithm (TOTP) is a popular time synchronization based algorithm for generating OTPs.

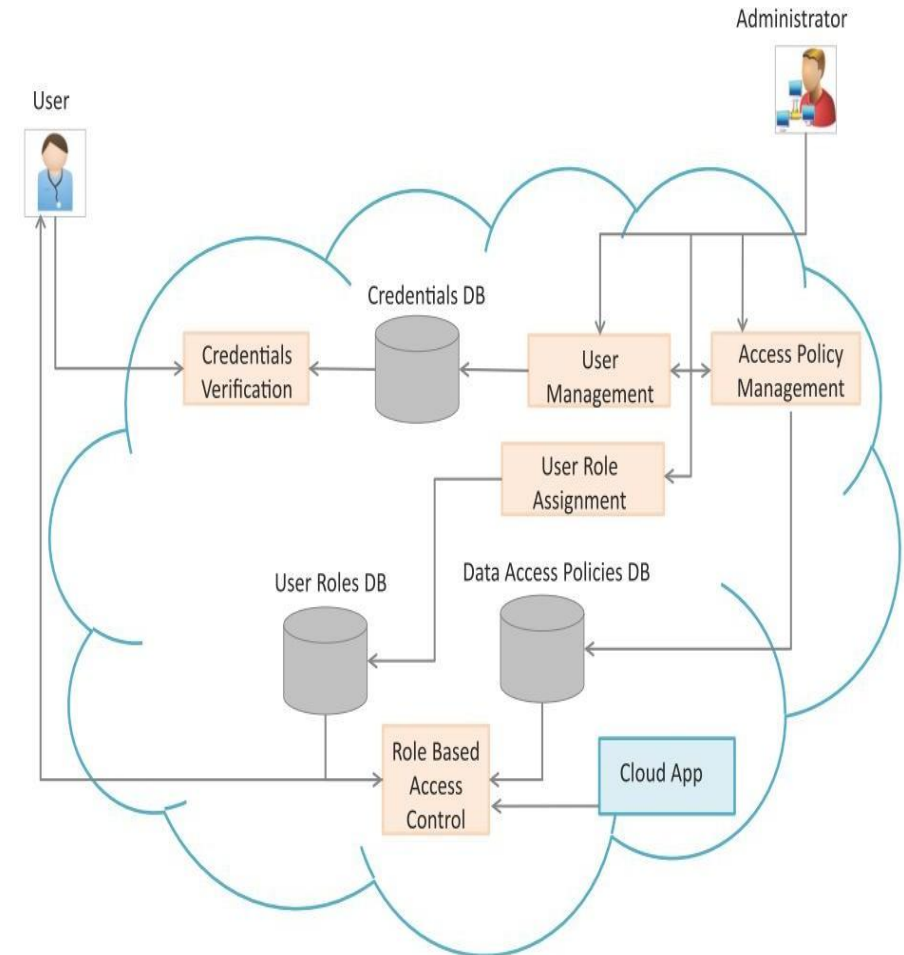
# Authorization

- Authorization refers to specifying the access rights to the protected resources using access policies.
- 
- OAuth
  - OAuth is an open standard for authorization that allows resource owners to share their private resources stored on one site with another site without handing out the credentials.
  - In the OAuth model, an application (which is not the resource owner) requests access to resources controlled by the resource owner (but hosted by the server).
  - The resource owner grants permission to access the resources in the form of a token and matching shared-secret.
  - Tokens make it unnecessary for the resource owner to share its credentials with the application.
  - Tokens can be issued with a restricted scope and limited lifetime, and revoked independently.



# Identity & Access Management

- Identity management provides consistent methods for digitally identifying persons and maintaining associated identity attributes for the users across multiple organizations.
- Access management deals with user privileges.
- Identity and access management deal with user identities, their authentication, authorization and access policies.
- Federated Identity Management
  - Federated identity management allows users of one domain to securely access data or systems of another domain seamlessly without the need for maintaining identity information separately for multiple domains.
  - Federation is enabled through the use single sign-on mechanisms such as SAML token and Kerberos.
- Role-based access control
  - Used for restricting access to confidential information to authorized users.
  - These access control policies allow defining different roles for different users.





# Securing Data at Rest

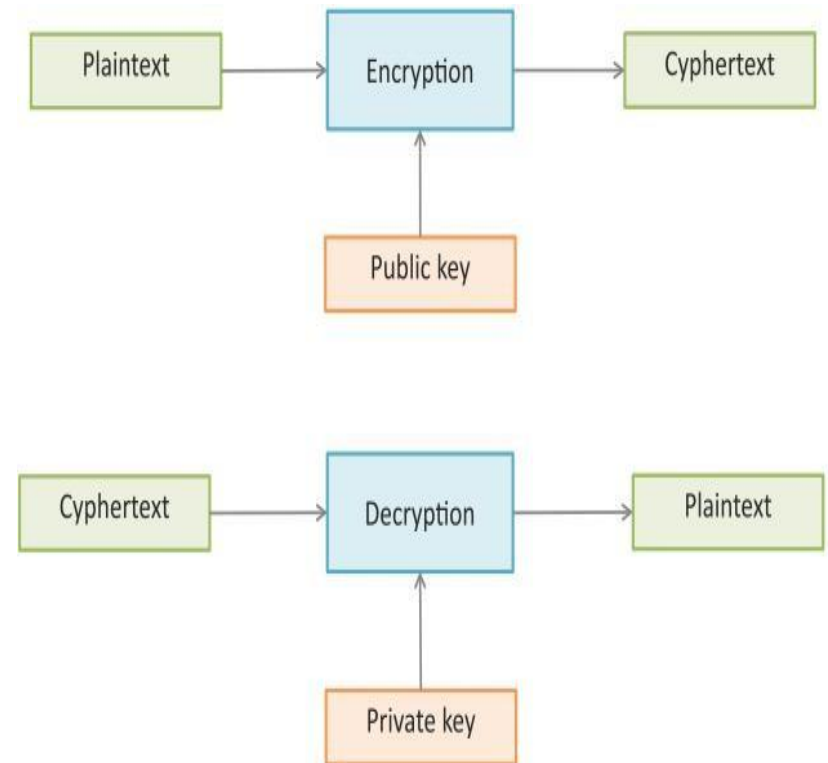
- Data at rest is the data that is stored in database in the form of tables/records, files on a file server or raw data on a distributed storage or storage area network (SAN).
- Data at rest is secured by encryption.
- Encryption is the process of converting data from its original form (i.e., plaintext) to a scrambled form (ciphertext) that is unintelligible. Decryption converts data from ciphertext to plaintext.
- Encryption can be of two types:
  - Symmetric Encryption (symmetric-key algorithms)
  - Asymmetric Encryption (public-key algorithms)

# Symmetric Encryption

- Symmetric encryption uses the same secret key for both encryption and decryption.
- The secret key is shared between the sender and the receiver.
- Symmetric encryption is best suited for securing data at rest since the data is accessed by known entities from known locations.
- Popular symmetric encryption algorithms include:
  - Advanced Encryption Standard (AES)
  - Twofish
  - Blowfish
  - Triple Data Encryption Standard (3DES)
  - Serpent
  - RC6
  - MARS

# Asymmetric Encryption

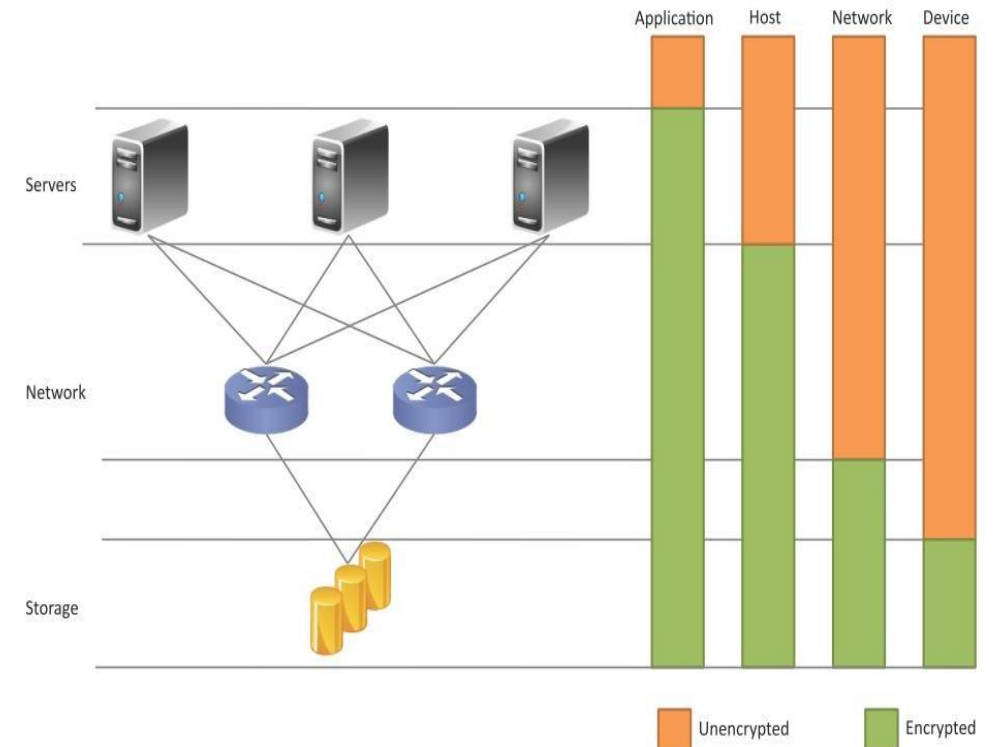
- Asymmetric encryption uses two keys, one for encryption (public key) and other for decryption (private key).
- The two keys are linked to each other such that one key encrypts plaintext to ciphertext and other decrypts ciphertext back to plaintext.
- Public key can be shared or published while the private key is known only to the user.
- Asymmetric encryption is best suited for securing data that is exchanged between two parties where symmetric encryption can be unsafe because the secret key has to be exchanged between the parties and anyone who manages to obtain the secret key can decrypt the data.
- In asymmetric encryption a separate key is used for decryption which is kept private.



# Encryption Levels

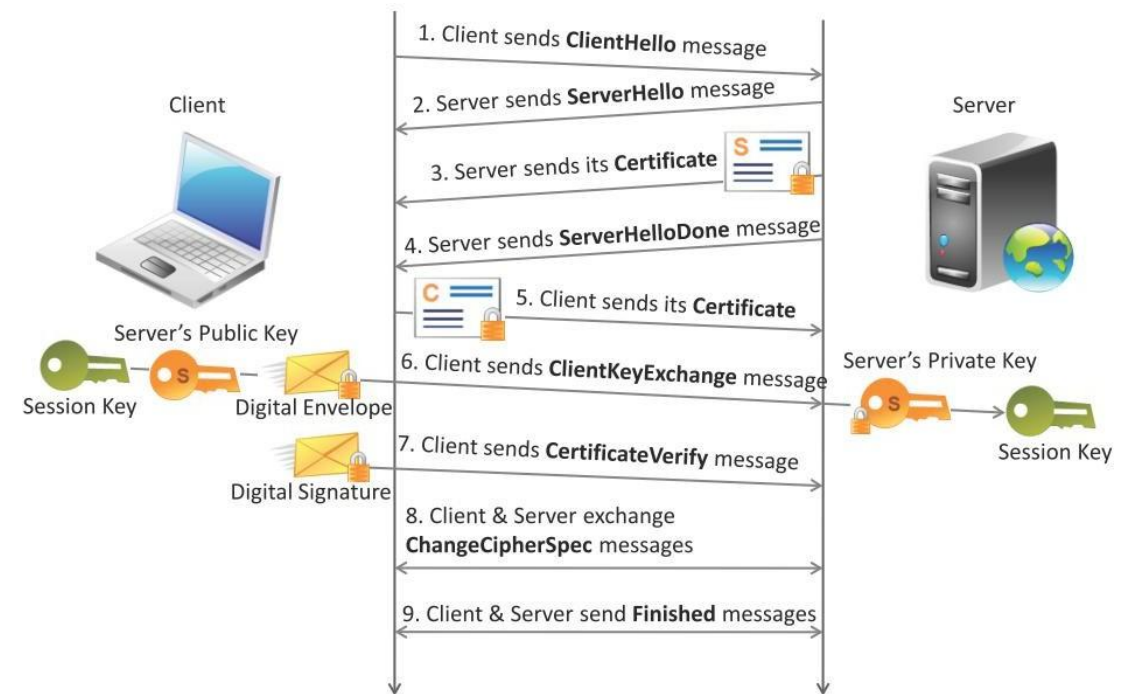
Encryption can be performed at various levels:

- **Application**
  - Application level encryption involves encrypting application data right at the point where it originates i.e. within the application.
  - Application level encryption provides security at the level of both the operating system and from other applications.
  - An application encrypts all data generated in the application before it flows to the lower levels and presents decrypted data to the user.
- **Host**
  - In host-level encryption, encryption is performed at the file-level for all applications running on the host.
  - Host level encryption can be done in software in which case additional computational resource is required for encryption or it can be performed with specialized hardware such as a cryptographic accelerator card.
- **Network**
  - Network-level encryption is best suited for cases where the threats to data are at the network or storage level and not at the application or host level.
  - Network-level encryption is performed when moving the data from a creation point to its destination using a specialized hardware that encrypts all incoming data in real-time.
- **Device**
  - Device-level encryption is performed on a disk controller or a storage server.
  - Device level encryption is easy to implement and is best suited for cases where the primary concern about data security is to protect data residing on storage media.



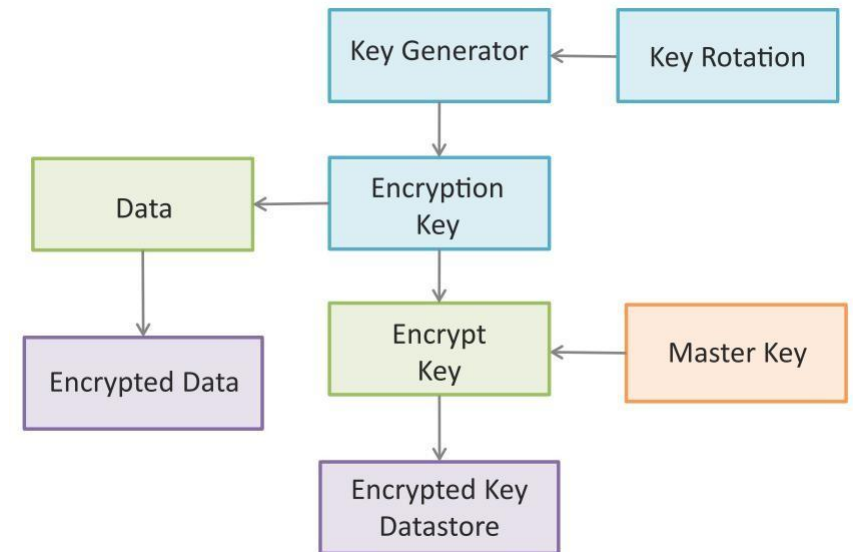
# Securing Data in Motion

- Securing data in motion, i.e., when the data flows between a client and a server over a potentially insecure network, is important to ensure data confidentiality and integrity.
- Data confidentiality means limiting the access to data so that only authorized recipients can access it.
- Data integrity means that the data remains unchanged when moving from sender to receiver.
- Data integrity ensures that the data is not altered in an unauthorized manner after it is created, transmitted or stored.
- Transport Layer Security (TLS) and Secure Socket Layer (SSL) are the mechanisms used for securing data in motion.
- TLS and SSL are used to encrypt web traffic using Hypertext Transfer Protocol (HTTP).
- TLS and SSL use asymmetric cryptography for authentication of key exchange, symmetric encryption



# Key Management

- Management of encryption keys is critical to ensure security of encrypted data.
- The key management lifecycle involves different phases including:
  - Creation
  - Backup
  - Deployment
  - Monitoring
  - Rotation
  - Expiration
  - Archival
  - Destruction
- Key Management Approach (example)
  - All keys for encryption must be stored in a data store which is separate and distinct from the actual data store.
  - Additional security features such as key rotation and key encrypting keys can be used.
  - Keys can be automatically or manually rotated.
  - In the automated key change approach, the key is changed after a certain number of transactions.
  - All keys can themselves be encrypted using a master key.



# Auditing

- Auditing is mandated by most data security regulations.
- Auditing requires that all read and write accesses to data be logged.
- Logs can include the user involved, type of access, timestamp, actions performed and records accessed.
- The main purpose of auditing is to find security breaches, so that necessary changes can be made in the application and deployment to prevent a further security breach.
- The objectives of auditing include:
  - Verify efficiency and compliance of identity and access management controls as per established access policies.
  - Verifying that authorized users are granted access to data and services based on their roles.
  - Verify whether access policies are updated in a timely manner upon change in the roles of the users.
  - Verify whether the data protection policies are sufficient.
  - Assessment of support activities such as problem management.