1.1 Introduction

Cloud computing is a transformative computing paradigm that involves delivering applications and services over the internet. Many of the underlying technologies that are the foundation of cloud computing have existed for quite some time. Cloud computing involves provisioning of computing, networking and storage resources on demand and providing these resources as metered services to the users, in a "pay as you go" model. In this chapter you will learn about the various deployment models, service models, characteristics, driving factors and challenges of cloud computing.

1.1.1 Definition of Cloud Computing

The U.S. National Institute of Standards and Technology (NIST) defines cloud computing as [1]:

> **Definition:** Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

1.2 Characteristics of Cloud Computing

NIST further identifies five essential characteristics of cloud computing:

On-demand self service

Cloud computing resources can be provisioned on-demand by the users, without requiring interactions with the cloud service provider. The process of provisioning resources is automated.

Broad network access

Cloud computing resources can be accessed over the network using standard access mechanisms that provide platform-independent access through the use of heterogeneous client platforms such as workstations, laptops, tablets and smartphones.

Resource pooling

The computing and storage resources provided by cloud service providers are pooled to serve multiple users using multi-tenancy. Multi-tenant aspects of the cloud allow multiple users to be served by the same physical hardware. Users are assigned virtual resources that run on top of the physical resources. Various forms of virtualization approaches such as full virtualization, para-virtualization and hardware virtualization are described in Chapter 2.

Rapid elasticity

Cloud computing resources can be provisioned rapidly and elastically. Cloud resources can be rapidly scaled up or down based on demand. Two types of scaling options exist:

- **Horizontal Scaling (scaling out):** Horizontal scaling or scaling-out involves launching and provisioning additional server resources.

- **Vertical Scaling (scaling up):** Vertical scaling or scaling-up involves changing the computing capacity assigned to the server resources while keeping the number of server resources constant.

Measured service

Cloud computing resources are provided to users on a pay-per-use model. The usage of the cloud resources is measured and the user is charged based on some specific metric. Metrics such as amount of CPU cycles used, amount of storage space used, number of network I/O requests, etc. are used to calculate the usage charges for the cloud resources.

In addition to these five essential characteristics of cloud computing, other characteristics that again highlight savings in cost include:

Performance

Cloud computing provides improved performance for applications since the resources available to the applications can be scaled up or down based on the dynamic application workloads.

Reduced costs

Cloud computing provides cost benefits for applications as only as much computing and storage resources as required can be provisioned dynamically, and upfront investment in purchase of computing assets to cover worst case requirements is avoid. This saves significant cost for organizations and individuals. Applications can experience large variations in the workloads which can be due to seasonal or other factors. For example, e-Commerce applications typically experience higher workloads in holiday seasons. To ensure market readiness of such applications, adequate resources need to be provisioned so that the applications can meet the demands of specified workload levels and at the same time ensure that service level agreements are met.

Outsourced Management

Cloud computing allows the users (individuals, large organizations, small and medium enterprises and governments) to outsource the IT infrastructure requirements to external cloud providers. Thus, the consumers can save large upfront capital expenditures in setting up the IT infrastructure and pay only for the operational expenses for the cloud resources used. The outsourced nature of the cloud services provides a reduction in the IT infrastructure management costs.

Reliability

Applications deployed in cloud computing environments generally have a higher reliability since the underlying IT infrastructure is professionally managed by the cloud service. Cloud service providers specify and guarantee the reliability and availability levels for their cloud resources in the form of service level agreements (SLAs). Most cloud providers promise 99.99% uptime guarantee for the cloud resources, which may often be expensive to achieve with in-house IT infrastructure.

Multi-tenancy

The multi-tenanted approach of the cloud allows multiple users to make use of the same shared resources. Modern applications such as e-Commerce, Business-to-Business, Banking and

Financial, Retail and Social Networking applications that are deployed in cloud computing environments are multi-tenanted applications. Multi-tenancy can be of different forms:

- **Virtual multi-tenancy:** In virtual multi-tenancy, computing and storage resources are shared among multiple users. Multiple tenants are served from virtual machines (VMs) that execute concurrently on top of the same computing and storage resources.
- **Organic multi-tenancy:** In organic multi-tenancy every component in the system architecture is shared among multiple tenants, including hardware, OS, database servers, application servers, load balancers, etc. Organic multi-tenancy exists when explicit multi-tenant design patterns are coded into the application.

1.3 Cloud Models

1.3.1 Service Models

Cloud computing services are offered to users in different forms. NIST defines at least three cloud service models as follows:

Infrastructure-as-a-Service (IaaS)

IaaS provides the users the capability to provision computing and storage resources. These resources are provided to the users as virtual machine instances and virtual storage. Users can start, stop, configure and manage the virtual machine instances and virtual storage. Users can deploy operating systems and applications of their choice on the virtual resources provisioned in the cloud. The cloud service provider manages the underlying infrastructure. Virtual resources provisioned by the users are billed based on a pay-per-use paradigm. Common metering metrics used are the number of virtual machine hours used and/or the amount of storage space provisioned.

Platform-as-a-Service (PaaS)

PaaS provides the users the capability to develop and deploy application in the cloud using the development tools, application programming interfaces (APIs), software libraries and services provided by the cloud service provider. The cloud service provider manages the underlying cloud infrastructure including servers, network, operating systems and storage. The users, themselves, are responsible for developing, deploying, configuring and managing applications on the cloud infrastructure.

Software-as-a-Service (SaaS)

SaaS provides the users a complete software application or the user interface to the the application itself. The cloud service provider manages the underlying cloud infrastructure including servers, network, operating systems, storage and application software, and the user is unaware of the underlying architecture of the cloud. Applications are provided to the user through a thin client interface (e.g., a browser). SaaS applications are platform independent and can be accessed from various client devices such as workstations, laptop, tablets and smartphones, running different operating systems. Since the cloud service provider manages both the application and data, the users are able to access the applications from anywhere.

Figure 1.1 shows the cloud computing service models and Figure 1.2 lists the benefits, characteristics and adoption of IaaS, PaaS and SaaS.

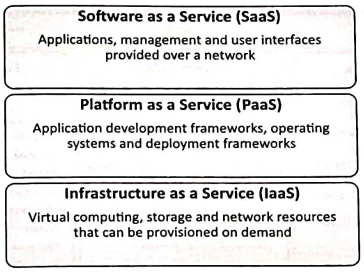

Figure 1.1: Cloud computing service models

1.3.2 Deployment Models

NIST also defines four cloud deployment models as follows:

Public cloud

In the public cloud deployment model, cloud services are available to the general public or a large group of companies. The cloud resources are shared among different users (individuals, large organizations, small and medium enterprises and governments). The cloud services are provided by a third-party cloud provider. Public clouds are best suited for users who want to use cloud infrastructure for development and testing of applications and host applications in the cloud to serve large workloads, without upfront investments in IT infrastructure.

Private cloud

In the private cloud deployment model, cloud infrastructure is operated for exclusive use of a single organization. Private cloud services are dedicated for a single organization. Cloud infrastructure can be setup on premise or off-premise and may be managed internally or by a third-party. Private clouds are best suited for applications where security is very important and organizations that want to have very tight control over their data.

Hybrid cloud

The hybrid cloud deployment model combines the services of multiple clouds (private or public). The individual clouds retain their unique identities but are bound by standardized or proprietary technology that enables data and application portability. Hybrid clouds are best suited for organizations that want to take advantage of secured application and data hosting on a private cloud, and at the same time benefit from cost savings by hosting shared applications and data in public clouds.

Community cloud

In the community cloud deployment model, the cloud services are shared by several organizations that have the same policy and compliance considerations. Community clouds are

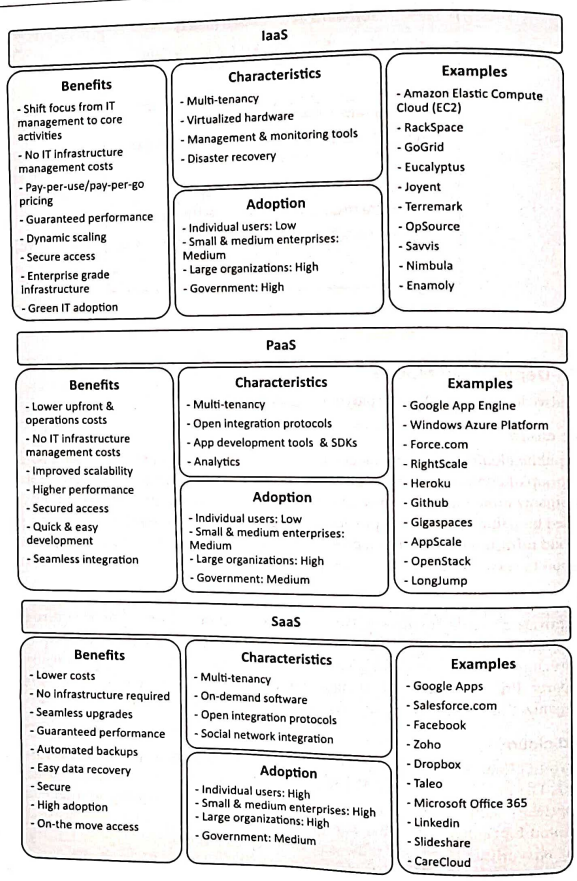

Figure 1.2: Benefits, characteristics and adoption of IaaS, PaaS and SaaS

best suited for organizations that want access to the same applications and data, and want the cloud costs to be shared with the larger group.

Figures 1.3 and 1.4 show the cloud deployment models.

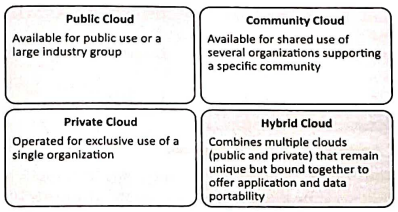

Figure 1.3: Cloud deployment models

1.4 Cloud Services Examples

1.4.1 IaaS: Amazon EC2, Google Compute Engine, Azure VMs

Amazon Elastic Compute Cloud (EC2) [3] is an Infrastructure as a Service (IaaS)n offering from Amazon.com. EC2 (TM) is a web service that provides computing capacity in the form of virtual machines that are launched in Amazon's cloud computing environment. Amazon EC2 allows users to launch instances on demand using a simple web-based interface. Amazon provides pre-configured Amazon Machine Images (AMIs) which are templates of cloud instances. Users can also create their own AMIs with custom applications, libraries and data. Instances can be launched with a variety of operating systems. Users can load their applications on running instances and rapidly and easily increase or decrease capacity to meet the dynamic application performance requirements. With EC2, users can even provision hundreds or thousands of server instances simultaneously, manage network access permissions, and monitor usage resources through a web interface. Amazon EC2 provides instances of various computing capacities ranging from small instances (e.g., 1 virtual core with 1EC2 compute unit, 1.7GB memory and 160GB instance storage) to extra large instances (e.g., 4 virtual cores with 2 EC2 compute units each, 15GB memory and 1690 GB instance storage). Amazon C2 also provides instances with high memory, high CPU resources, cluster compute instances, cluster graphical processor unit (GPU) instances and high Input/Output (I/O) instances. The pricing model for EC2 instances is based on a pay-per-use model. Users are billed based on the number on instance hours used for on-demand instances. EC2 provides the option of reserving instances by one-time payment for each instance that the user wants to reserve. In addition to these on-demand and reserved instances, EC2 also provides spot instances that allow users to bid on unused Amazon EC2 capacity and run those instances for as long as their bid exceeds the current spot price. Amazon EC2 provides a number of powerful

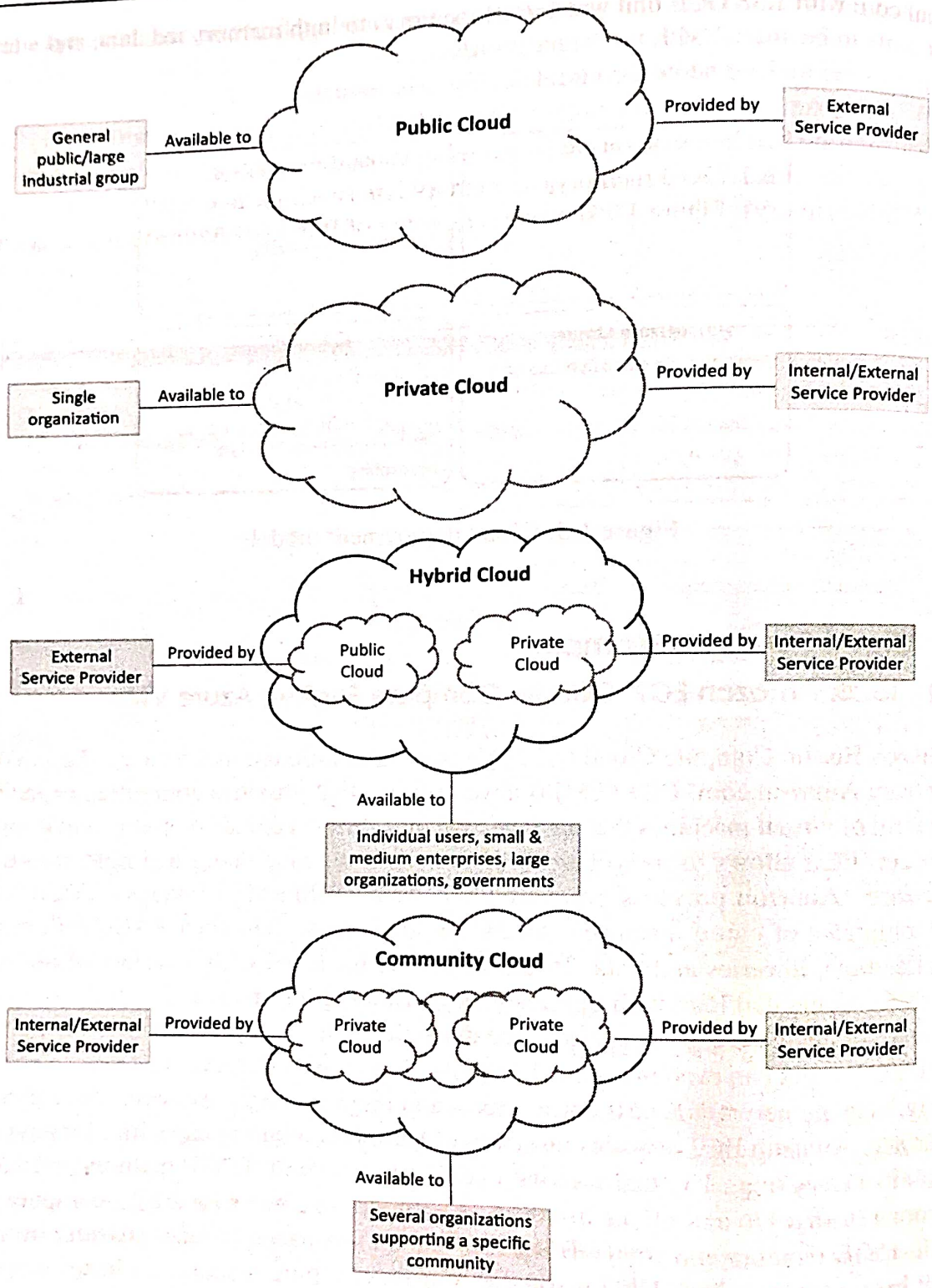

Figure 1.4: Cloud deployment models

features for building scalable and reliable applications such as auto scaling and elastic load balancing. Figure 1.5 shows a screenshot of Amazon EC2 dashboard.

Google Compute Engine (GCE) [4] is an IaaS offering from Google. GCE provides virtual machines of various computing capacities ranging from small instances (e.g., 1

In this chapter you will learn the key concepts and enabling technologies of cloud computing. We will introduce and build upon technologies such as virtualization, load balancing,and on-demand provisioning. A popular programming model, called MapReduce, will also be covered.

2.1 Virtualization

Virtualization refers to the partitioning the resources of a physical system (such as computing, storage, network and memory) into multiple virtual resources. Virtualization is the key enabling technology of cloud computing and allows pooling of resources. In cloud computing, resources are pooled to serve multiple users using multi-tenancy. Multi-tenant aspects of the cloud allow multiple users to be served by the same physical hardware. Users are assigned virtual resources that run on top of the physical resources. Figure 2.1 shows the architecture of a virtualization technology in cloud computing. The physical resources such as computing, storage memory and network resources are virtualized. The virtualization layer partitions the physical resources into multiple virtual machines. The virtualization layer allows multiple operating system instances to run currently as virtual machines on the same underlying physical resources.

Hypervisor

The virtualization layer consists of a hypervisor or a virtual machine monitor (VMM). The hypervisor presents a virtual operating platform to a guest operating system (OS). There are two types of hypervisors as shown in Figures 2.2 and 2.3 . Type-1 hypervisors or the native hypervisors run directly on the host hardware and control the hardware and monitor the guest operating systems. Type 2 hypervisors or hosted hypervisors run on top of a conventional (main/host) operating system and monitor the guest operating systems.

Guest OS

A guest OS is an operating system that is installed in a virtual machine in addition to the host or main OS. In virtualization, the guest OS can be different from the host OS.

Various forms of virtualization approaches exist:

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. Figure 2.4 shows the full virtualization approach.

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 hypercalls that communicate directly with the virtualization layer hypervisor. Figure 2.5 shows the para-virtualization approach.

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.

Table 2.1 lists some examples of popular hypervisors.

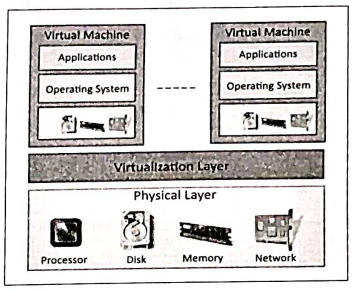

Figure 2.1: Virtualization architecture

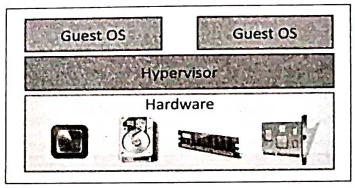

Figure 2.2: Hypervisor design: Type-1

100053

2.2 Load Balancing

One of the important features of cloud computing is scalability. 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 are to achieve maximum utilization of resources, minimizing the response times, maximizing throughput. Load balancing distributes the incoming user requests across multiple resources. With load balancing, cloud-based applications can achieve high availability and reliability. Since multiple resources under a load balancer are used to serve the user requests, in the event of failure of one or more of the resources, the load balancer can automatically reroute the user traffic to the healthy resources. To the end user accessing a cloud-based application, a load balancer makes the pool of servers under the

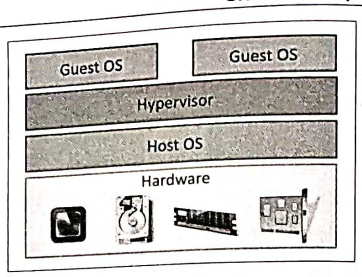

Figure 2.3: Hypervisor design: Type-2

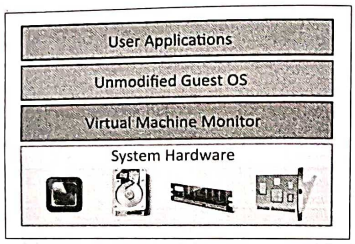

Figure 2.4: Full virtualization

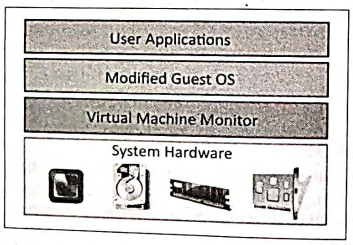

Figure 2.5: Para-virtualization

load balancer appear as a single server with high computing capacity. The routing of user requests is determined based on a load balancing algorithm. Commonly used load balancing algorithms include:

Hypervisor	Type
Citrix XenServer	Type-1
Oracle VM Server	Type-1
KVM	Type-1
VMWare ESX/ESXi	Type-1
Microsoft Hyper-V	Type-1
Xen Hypervisor	Type-1
VMWare Workstation	Type-2
VirtualBox	Type-2

Table 2.1: Examples of popular hypervisors

Round Robin
In round robin load balancing, the servers are selected one by one to serve the incoming requests in a non-hierarchical circular fashion with no priority assigned to a specific server.

Weighted Round Robin
In weighted round robin load balancing, severs are assigned some weights. The incoming requests are proportionally routed using a static or dynamic ratio of respective weights.

Low Latency
In low latency load balancing the load balancer monitors the latency of each server. Each incoming request is routed to the server which has the lowest latency.

Least Connections
In least connections load balancing, the incoming requests are routed to the server with the least number of connections.

Priority
In priority load balancing, each server is assigned a priority. The incoming traffic is routed to the highest priority server as long as the server is available. When the highest priority server fails, the incoming traffic is routed to a server with a lower priority.

Overflow
Overflow load balancing is similar to priority load balancing. When the incoming requests to highest priority server overflow, the requests are routed to a lower priority server.

Figure 2.6 depicts these various load balancing approaches. For session based applications, an important issue to handle during load balancing is the persistence of multiple requests from a particular user session. 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. Three commonly used persistence approaches are described below:

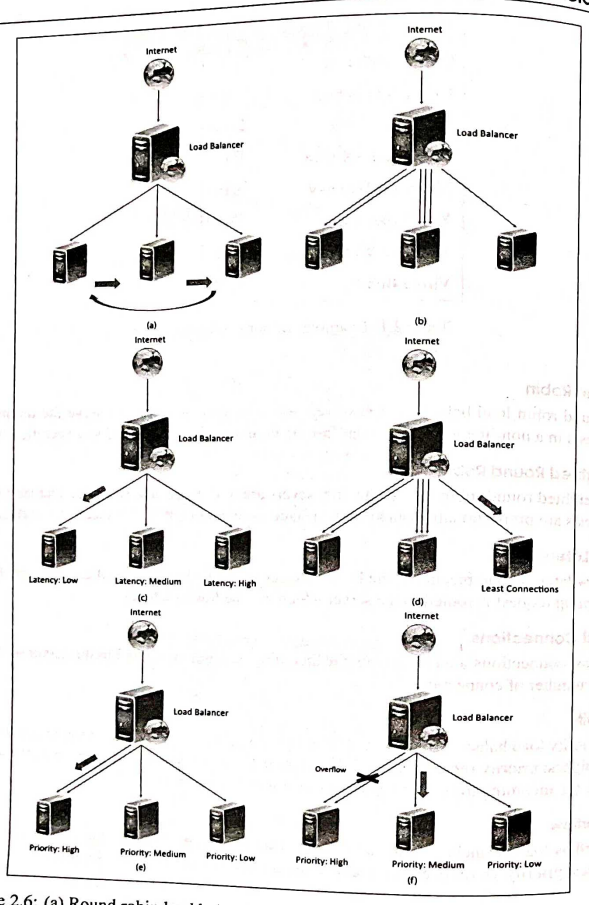

Figure 2.6: (a) Round robin load balancing, (b) Weighted round robin load balancing, (c) Low latency load balancing, (d) Least connections load balancing, (e) Priority load balancing, (f) Overload load balancing

Sticky sessions

In this approach all the requests belonging to a user session are routed to the same server. These sessions are called sticky sessions. The benefit of this approach is that it makes session management simple. However, a drawback of this approach is that if a server fails all the sessions belonging to that server are lost, since there is no automatic failover possible.

Session Database

In this approach, all the session information is stored externally in a separate session database, which is often replicated to avoid a single point of failure. Though, this approach involves additional overhead of storing the session information, however, unlike the sticky session approach, this approach allows automatic failover.

Browser cookies

In this approach, the session information is stored on the client side in the form of browser cookies. The benefit of this approach is that it makes the session management easy and has the least amount of overhead for the load balancer.

URL re-writing

In this approach, a URL re-write engine stores the session information by modifying the URLs on the client side. Though this approach avoids overhead on the load balancer, a drawback is that the amount of session information that can be stored is limited. For applications that require larger amounts of session information, this approach does not work.

Load balancing can be implemented in software or hardware. Software-based load balancers run on standard operating systems, and like other cloud resources, load balancers are also virtualized. Hardware-based load balancers implement load balancing algorithms in Application Specific Integrated Circuits (ASICs). In a hardware load balancer, the incoming user requests are routed to the underlying servers based on some pre-configured load balancing strategy and the response from the severs are sent back either directly to the user (at layer-4) or back to the load balancer (at layer-7) where it is manipulated before being sent back to the user. Table 2.2 lists some examples of load balancers.

2.3 Scalability & Elasticity

Multi-tier applications such as e-Commerce, social networking, business-to-business, etc. can experience rapid changes in their traffic. Each website has a different traffic pattern which is determined by a number of factors that are generally hard to predict beforehand. Modern web applications have multiple tiers of deployment with varying number of servers in each tier. Capacity planning is an important task for such applications. 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. Figure 2.7 shows the cost versus capacity curves for traditional and cloud approaches.

Traditional approaches for capacity planning are based on predicted demands for applications and account for worst case peak loads of applications. When the workloads of applications increase, the traditional approaches have been either to scale up or scale

Load Balancer	Type
Nginx	Software
HAProxy	Software
Pound	Software
Varish	Software
Cisco Systems Catalyst 6500	Hardware
Coyote Point Equalizer	Hardware
F5 Networks BIG-IP LTM	Hardware
Barracuda Load Balancer	Hardware

Table 2.2: Examples of popular load balancers

out. Scaling up involves upgrading the hardware resources (adding additional computing, memory, storage or network resources). Scaling out involves addition of more resources of the same type. Traditional scaling up and scaling out approaches are based on demand forecasts at regular intervals of time. When variations in workloads are rapid, traditional approaches are unable to keep track with the demand and lead to either over-provisioning or under-provisioning of resources. Over-provisioning of resources leads to higher capital expenditures than required. On the other hand, under-provisioning of resources leads to traffic overloads, slow response times, low throughputs and hence loss of opportunity to serve the customers. Analyzing the real traffic history plots for top websites shown in Figure 2.7 we observe that the off peak workloads are significantly lower than peak workloads. Traditional capacity planning approaches which are designed to meet the peak loads result in excess capacity and under utilization of resources. Moreover, the infrastructure resources for traditional applications are fixed, rigid and provisioned in advance. This involves up-front capital expenditures for setting up the infrastructure.

2.4 Deployment

Figure 2.8 shows the cloud application deployment lifecycle. Deployment prototyping can help in making deployment architecture design choices. By comparing performance of alternative deployment architectures, deployment prototyping can help in choosing the best and most cost effective deployment architecture that can meet the application performance requirements. Table 2.3 lists some popular cloud deployment management tools. Deployment design is an iterative process that involves the following steps:

Deployment Design

In this step the application deployment is created with various tiers as specified in the deployment configuration. The variables in this step include the number of servers in each tier, computing, memory and storage capacities of severs, server interconnection, load balancing and replication strategies. Deployment is created by provisioning the cloud

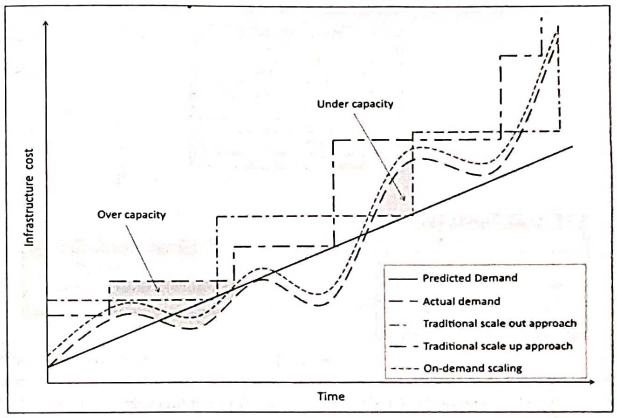

Figure 2.7: Cost versus capacity curves

resources as specified in the deployment configuration. The process of resource provisioning and deployment creation is often automated and involves a number of steps such as launching of server instances, configuration of servers, and deployment of various tiers of the application on the servers.

Performance Evaluation

Once the application is deployed in the cloud, the next step in the deployment lifecycle is to verify whether the application meets the performance requirements with the deployment. This step involves monitoring the workload on the application and measuring various workload parameters such as response time and throughput. In addition to this, the utilization of servers (CPU, memory, disk, I/O, etc.) in each tier is also monitored.

Deployment Refinement

After evaluating the performance of the application, deployments are refined so that the application can meet the performance requirements. 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.

2.5 Replication

Replication is used to create and maintain multiple copies of the data in the cloud. Replication of data is important for practical reasons such as business continuity and disaster recovery.

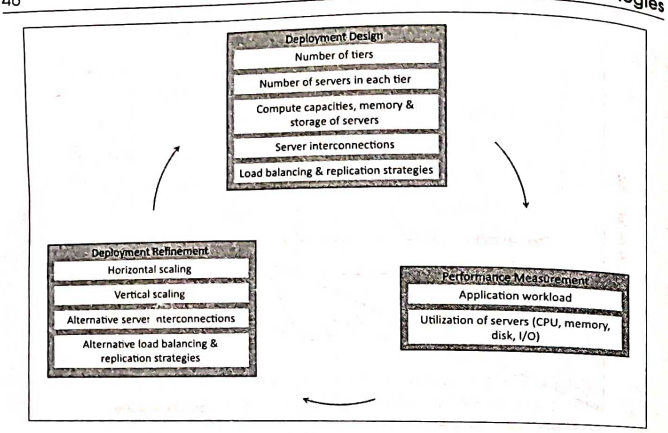

Figure 2.8: Cloud application deployment lifecycle

In the event of data loss at the primary location, organizations can continue to operate their applications from secondary data sources. With real-time replication of data, organizations can achieve faster recovery from failures. Traditional business continuity and disaster recovery approaches don't provide efficient, cost effective and automated recovery of data. Cloud based data replication approaches provide replication of data in multiple locations, automated recovery, low recovery point objective (RPO) and low recovery time objective (RTO). Cloud enables rapid implementation of replication solutions for disaster recovery for small and medium enterprises and large 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. Cloud provides affordable replication solutions with pay-per-use/pay-as-you-go pricing models. There are three types of replication approaches as shown in Figure 2.9 and described as follows:

Array-based Replication

Array-based replication uses compatible storage arrays to automatically copy data from a local storage array to a remote storage array. Arrays replicate data at the disk sub-system level, therefore the type of hosts accessing the data and the type of data is not important. Thus array-based replication can work in heterogeneous environments with different operating systems. Array-based replication uses Network Attached Storage (NAS) or Storage Area Network (SAN), to replicate. A drawback of this array-based replication is that it requires similar arrays at local and remote locations. Thus the costs for setting up array-based replication are higher than the other approaches.

Cloud Deployment Management Tool	Features
RightScale	Design, deploy and manage cloud deployments across multiple public or private clouds.
Scalr	Provides tools to automate the management of servers, monitors servers, replaces servers that fail, provides auto scaling and backups.
Kaavo	Allows deploying applications easily across multiple clouds, managing distributed applications and automating high availability.
CloudStack	Allows simple and cost effective deployment management and configuration of cloud computing environments.

Table 2.3: Examples of popular cloud deployment management tools

Network-based Replication

Network-based replication uses an appliance that sits on the network and intercepts packets that are sent from hosts and storage arrays. The intercepted packets are replicated to a secondary location. The benefits of this approach is that it supports heterogeneous environments and requires a single point of management. However, this approach involves higher initial costs due to replication hardware and software.

Host-based Replication

Host-based replication runs on standard servers and uses software to transfer data from a local to remote location. The host acts the replication control mechanism. An agent is installed on the hosts that communicates with the agents on the other hosts. Host-based replication can either be block-based or file-based. Block-based replication typically require dedicated volumes of the same size on both the local and remote servers. File-based replication requires less storage as compared to block-based storage. File-based replication gives additional allows the administrators to choose the files or folders to be replicated. Host-based replication with cloud-infrastructure provides affordable replication solutions. With host-based replication, entire virtual machines can be replicated in real-time.

2.6 Monitoring

Cloud resources can be monitored by monitoring services provided by the cloud service providers. Monitoring services allow cloud users to collect and analyze the data on various monitoring metrics. Figure 2.10 shows a generic architecture for a cloud monitoring service. A monitoring service collects data on various system and application metrics from the cloud computing instances. Monitoring services provide various pre-defined metrics. Users can also define their custom metrics for monitoring the cloud resources. Users can define various actions based on the monitoring data, for example, auto-scaling a cloud deployment when the CPU usage of monitored resources becomes high. Monitoring services also provide various statistics based on the monitoring data collected. Table 2.4 lists the commonly

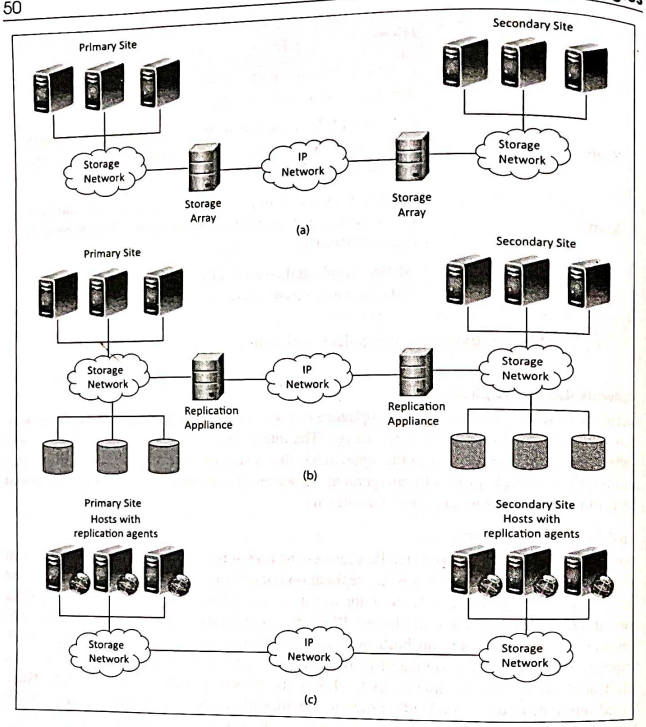

Figure 2.9: Replication approaches: (a) Array-based replication, (b) Network-based replication, (c) Host-based replication

used monitoring metrics for cloud computing resources. 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. For example, an organization which has its website hosted in the cloud can monitor the performance of the websit and also the website traffic. With the monitoring data available at run-time users can make operational decisions such as scaling up or scaling down cloud resources.

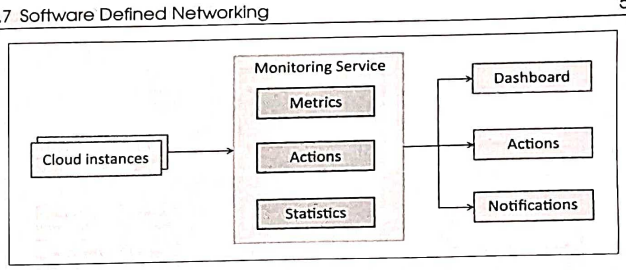

Figure 2.10: Typical cloud monitoring service architecture

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)

Table 2.4: Typical monitoring metrics

2.7 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. Figure 2.11 shows the conventional network architecture built with specialized hardware (switches, routers, etc.). Network devices in conventional network architectures are getting exceedingly complex with the increasing number of distributed protocols being implemented and the use of proprietary hardware and interfaces. In the 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.

The limitations of the conventional network architectures are as follows:

- **Complex Network Devices**: Conventional networks are getting increasingly complex with more and more protocols being implemented to improve link speeds and reliability. Interoperability is limited due to the lack of standard and open interfaces. Network devices use proprietary hardware and software and have slow product lifecycles limiting innovation. The conventional networks were well suited for static traffic patterns and had a large number of protocols designed for specific applications. With the emergence of cloud computing and proliferation of internet access devices, the traffic patterns are becoming more and more dynamic. Due to the complexity of conventional network devices, making changes in the networks to meet the dynamic traffic patterns has

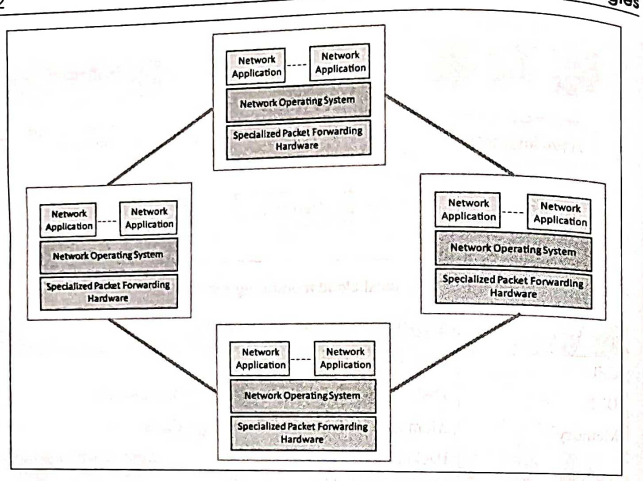

Figure 2.11: Conventional network architecture

become increasingly difficult.

- **Management Overhead**: Conventional networks involve significant management overhead. Network managers find it increasingly difficult to manage multiple network devices and interfaces from multiple vendors. Upgradation of network requires configuration changes in multiple devices (switches, routers, firewalls, etc.)
- **Limited Scalability**: The virtualization technologies used in cloud computing environments has increased the number of virtual hosts requiring network access. Multi-tenanted applications hosted in the cloud are distributed across multiple virtual machines that require exchange of traffic. Big data applications run distributed algorithms on a large number of virtual machines that require huge amounts of data exchange between virtual machines. Such computing environments require highly scalable and easy to manage network architectures with minimal manual configurations, which is becoming increasingly difficult with conventional networks.

SDN attempts to create network architectures that are simpler, inexpensive, scalable, agile and easy to manage. Figures 2.12 and 2.13 show the SDN architecture and the SDN layers in which the control and data planes are decoupled and the network controller is centralized. Software-based SDN controllers maintain a unified view of the network and make configuration, management and provisioning simpler. The underlying infrastructure in SDN uses simple packet forwarding hardware as opposed to specialized hardware in conventional networks. The underlying network infrastructure is abstracted from the applications. Network devices become simple with SDN as they do not require implementations of a large number of

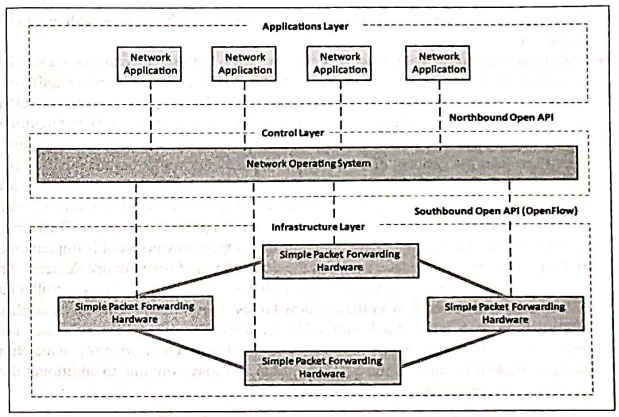

Figure 2.12: SDN architecture

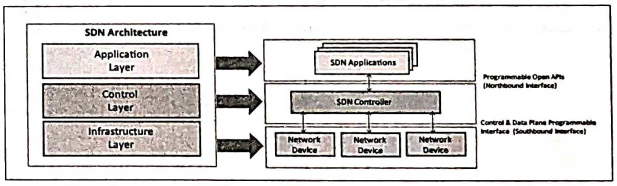

Figure 2.13: SDN layers

protocols. Network devices receive instructions from the SDN controller on how to forward the packets. These devices can be simpler and cost less as they can be built from standard hardware and software components.

Key elements of SDN are as follows:

- **Centralized Network Controller**: With decoupled the control and data planes and centralized network controller, the network administrators can rapidly configure the network. SDN applications can be deployed through programmable open APIs. This speeds up innovation as the network administrators no longer need to wait for the device vendors to embed new features in their proprietary hardware.
- **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. 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. Figure 2.14 shows the components of an 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. OpenFlow protocol is implemented on both sides of the interface between the controller and the network devices. The controller manages the switch via the OpenFlow switch protocol. The controller can add, update, and delete flow entries in flow tables. Figure 2.15 shows an example of an OpenFlow flow table. Each flow table contains a set of flow entries. Each flow entry consists of match fields, counters, and a set of instructions to apply to matching packets. Matching starts at the first flow table and may continue to additional flow tables of the pipeline [12].

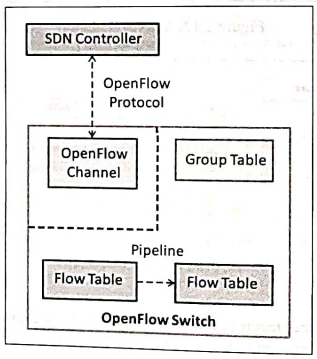

Figure 2.14: OpenFlow switch

2.8 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. 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.

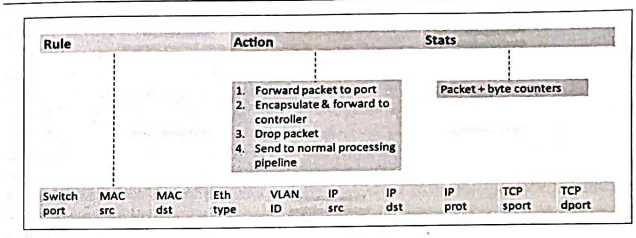

Figure 2.15: OpenFlow flow table

Network functions can be virtualized without SDN, similarly, SDN can run without NFV.

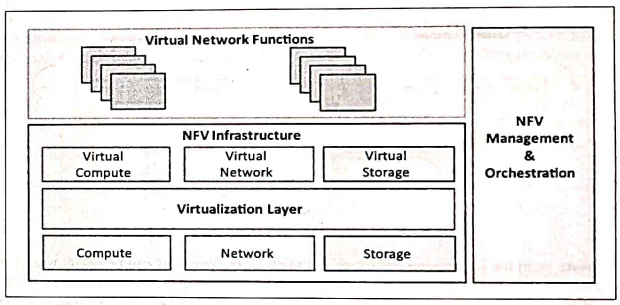

Figure 2.16: NFV architecture

Figure 2.16 shows the NFV architecture, as being standardized by the European Telecommunications Standards Institute (ETSI) [11]. Key elements of the NFV architecture are as follows:

- **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.

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

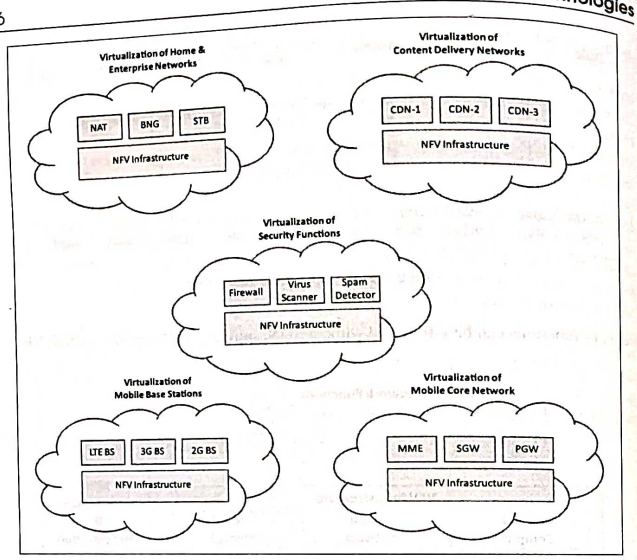

Figure 2.17: NFV use cases

in software from the underlying hardware. Thus network functions can be easily tested and upgraded by installing new software while the hardware remains the same. Virtualizing network functions reduces the equipment costs and also reduces power consumption. The multi-tenanted nature of the cloud allows virtualized network functions to be shared for multiple network services. NFV is applicable only to data plane and control plane functions in fixed and mobile networks. Figure 2.17 shows use cases of NFV for home and enterprise networks, content delivery networks, mobile base stations, mobile core network and security functions.

2.9 MapReduce

MapReduce is a parallel data processing model for processing and analysis of massive scale data [14]. MapReduce model has two phases: Map and Reduce. MapReduce programs are written in a functional programming style to create Map and Reduce functions. The input data to the map and reduce phases is in the form of key-value pairs. Run-time systems for MapReduce are typically large clusters built of commodity hardware. The MapReduce run-time systems take care of tasks such partitioning the data, scheduling of jobs and communication between nodes in the cluster. This makes it easier for programmers

to analyze massive scale data without worrying about tasks such as data partitioning and scheduling. Figure 2.18 shows the workflow of MapReduce. 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. When all the Map tasks are completed, the Reduce phase begins in which the intermediate data with the same key is aggregated. An optional Combine task can be used to perform data aggregation on the intermediate data of the same key for the output of the mapper before transferring the output to the Reduce task. Figure 2.19 shows the flow of data for a MapReduce job. MapReduce programs take a set of input key-value pairs and produce a set of output key-value pairs. MapReduce programs take advantage of locality of data and the data processing takes place on the nodes where the data resides. In traditional approaches for data analysis, data is moved to the compute nodes which results in significant of data transmission between the nodes in a cluster. MapReduce programming model moves the computation to where the data resides thus decreasing the transmission of data and improving efficiency. MapReduce programming model is well suited for parallel processing of massive scale data in which the data analysis tasks can be accomplished by independent map and reduce operations.

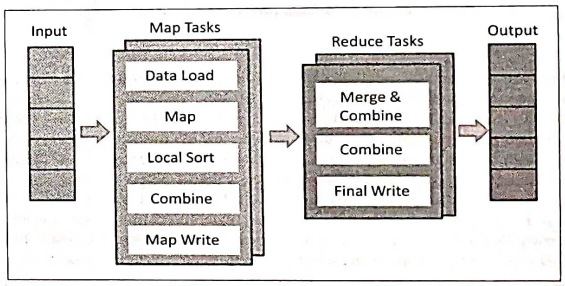

Figure 2.18: MapReduce workflow

2.10 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

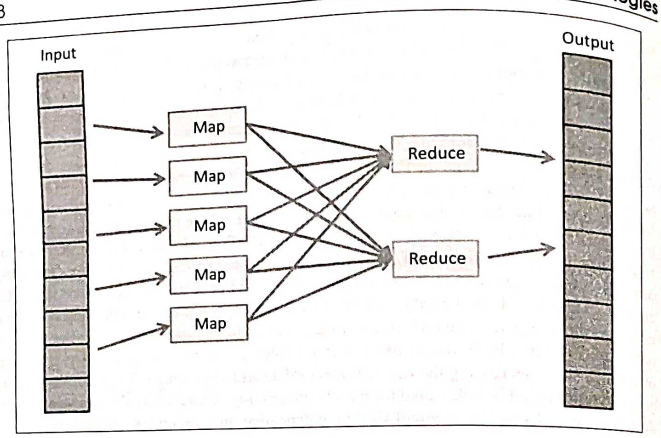

Figure 2.19: Data flow in MapReduce

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. Figure 2.20 shows the examples of OAuth and RBAC. OAuth is an open standard for authorization that allows resource owners to share their private resources stored on one site with an- other 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. RBAC is an approach for restricting access to authorized users. Figure 2.21 shows an example of a typical RBAC framework. A user who wants to access cloud resources is required to send his/her data to the system administrator who assigns permissions and access control policies which are stored in the User Roles and Data Access Policies databases respectively.

2.11 Service Level Agreements

A Service Level Agreement (SLA) for cloud specifies the level of service that is formally defined as a part of the service contract with the cloud service provider. SLAs provide a level of service for each service which is specified in the form of minimum level of service guaranteed and a target level. SLAs contain a number of performance metrics and the corresponding service level objectives. Table 2.5 lists the common criteria cloud SLAs.

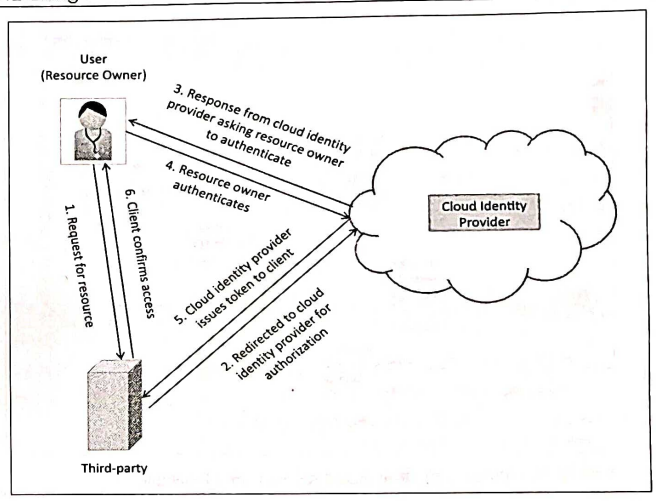

Figure 2.20: OAuth example

2.12 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. Cloud computing provides the benefit of provision resources on-demand. On-demand provisioning and elastic pricing models bring cost savings for customers. Elastic pricing model is suited for customers who consume cloud resources for short durations and who cannot predict the usage beforehand.

Fixed Pricing

In fixed pricing models, customers are charged a fixed amount per month for the cloud resources. For example, fixed amount can be charged per month for running a virtual machine instance, irrespective of the actual usage. Fixed pricing model is suited for customers who want to use cloud resources for longer durations and want more control over the cloud expenses.

Spot Pricing

Spot pricing models offer variable pricing for cloud resources which is driven by market demand. When the demand for cloud resources is high, the prices increase and when the

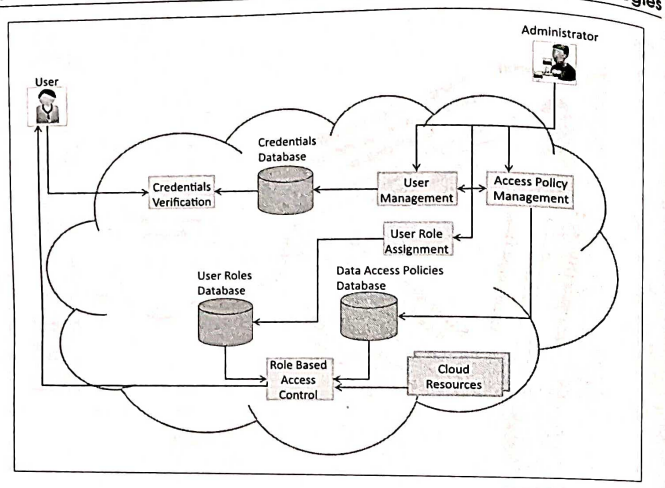

Figure 2.21: Role-based Access Control example

demand is lower, the prices decrease.

Table 2.6 lists the billable resources for cloud including virtual machines, network, storage, data services, security services, support, application services, deployment and management services.

Summary

In this chapter you learned cloud computing concepts and enabling technologies such as virtualization, load balancing, scalability & elasticity, deployment, replication, MapReduce, identity & access management, service level agreements and billing. Virtualization partitions the resources of a physical system (such as computing, storage, network and memory) into multiple virtual resources and enables resource pooling and multi-tenancy.

Review Questions

1. What are the various layers in a virtualization architecture?
2. What is the difference between full and para-virtualization?
3. What are the benefits of load balancing?
4. What are sticky sessions?
5. What are the differences between traditional and on-demand scaling approaches?
6. What are the various stages in the deployment lifecycle?
7. What is the difference between array-based and host-based replication?

Criteria	Details
Availability	Percentage of time the service is guaranteed to be available
Performance	Response time, Throughput
Disaster Recovery	Mean time to recover
Problem resolution	Process to identify problems, support options, resolution expectations
Security and privacy of data	Mechanisms for security of data in storage and transmission

Table 2.5: List of criteria for cloud SLAs

Resource	Details
Virtual machines	CPU, memory, storage, disk I/O, network I/O
Network	Network I/O, load balancers, DNS, firewall, VPN
Storage	Cloud storage, storage volumes, storage gateway
Data services	Data import/export services, data encryption, data compression, data backup, data redundancy, content delivery
Security services	Identity and access management, isolation, compliance
Support	Level of support, SLA, fault tolerance
Application services	Queuing service, notification service, workflow service, payment service
Deployment and management services	Monitoring service, deployment service

Table 2.6: List of billable resources for cloud

8. In MapReduce, what are the functions of map, reduce and combine tasks?
9. Describe three applications that can benefit from the MapReduce programming model?
10. What are the various criteria for service level agreements?