UNIT 1

1)Explain High-performance computing and High-throughput computing. What are the design objectives to achieve High-performance computing and High-throughput computing? What are the applications of High-performance computing and High-throughput computing?

HPC systems emphasize the raw speed performance. The speed of HPC systems has increased from Gflops in the early 1990s to now Pflops in 2010. This improvement was driven mainly by the demands from scientific, engineering, and manufacturing communities.

The development of market-oriented high-end computing systems is undergoing a strategic change from an HPC paradigm to an HTC paradigm. This HTC paradigm pays more attention to high-flux computing. The main application for high-flux computing is in Internet searches and web services by millions or more users simultaneously. The performance goal thus shifts to measure *high throughput* or the number of tasks completed per unit of time. HTC technology needs to not only improve in terms of batch processing speed, but also address the acute problems of cost, energy savings, security, and reliability at many data and enterprise computing centers.

Following are design objectives:

• **Efficiency** measures the utilization rate of resources in an execution model by exploiting massive parallelism in HPC. For HTC, efficiency is more closely related to job throughput, data access, storage, and power efficienc

• **Dependability** measures the reliability and self-management from the chip to the system and application levels. The purpose is to provide high-throughput service with Quality of Service (QoS) assurance, even under failure conditions.

• **Adaptation in the programming model** measures the ability to support billions of job requests over massive data sets and virtualized cloud resources under various workload and service models.

• **Flexibility in application deployment** measures the ability of distributed systems to run well in both HPC (science and engineering) and HTC (business) applications.

Applications of HPC and HTC Systems:

Domain	Specific Applications
Science and engineering	Scientific simulations, genomic analysis, etc. Earthquake prediction, global warming, weather forecasting, etc.
Business, education, services industry, and health care	Telecommunication, content delivery, e-commerce, etc. Banking, stock exchanges, transaction processing, etc. Air traffic control, electric power grids, distance education, etc. Health care, hospital automation, telemedicine, etc.
Internet and web services, and government applications	Internet search, data centers, decision-making systems, etc. Traffic monitoring, worm containment, cyber security, etc. Digital government, online tax return processing, social networking, etc.
Mission-critical applications	Military command and control, intelligent systems, crisis management, etc.

2)What are the three new computing paradigms? Define Centralized computing, Parallel computing, Distributed computing, Cloud Computing, Ubiquitous Computing, Internet Computing and Utility computing.

Advances in virtualization make it possible to see the growth of Internet clouds as a new computing paradigm. The maturity of *radio-frequency identification (RFID), Global Positioning System (GPS),* and sensor technologies has triggered the development of the *Internet of Things (IoT).*

• **Centralized computing** This is a computing paradigm by which all computer resources are centralized in one physical system. All resources (processors, memory, and storage) are fully shared and tightly coupled within one integrated OS. Many data centers and supercomputers are *centralized systems*, but they are used in parallel, distributed, and cloud computing applications [18,26].

• **Parallel computing** In parallel computing, all processors are either tightly coupled with centralized shared memory or loosely coupled with distributed memory. Some authors refer to this discipline as *parallel processing* [15,27]. Interprocessor communication is accomplished through shared memory or via message passing. A computer system capable of parallel computing is commonly known as a *parallel computer* [28]. Programs running in a parallel computer are called *parallel programs*.The process of writing *parallel programs* is often referred to as *parallel programming* [32].

• **Distributed computing** This is a field of computer science/engineering that studies distributed systems. A *distributed system* [8,13,37,46] consists of multiple autonomous computers, each having its own private memory, communicating through a computer

network. Information exchange in a distributed system is accomplished through *message passing*. A computer program that runs in a distributed system is known as a *distributed program*. The process of writing distributed programs is referred to as *distributed programming*.

• **Cloud computing** An *Internet cloud* of resources can be either a centralized or a distributed computing system. The cloud applies parallel or distributed computing, or both. Clouds can be built with physical or virtualized resources over large data centers that are centralized or distributed. Some authors consider cloud computing to be a form **of *utility computing*** or *service computing.*

Ubiquitous computing refers to computing with pervasive devices at any place and time using wired or wireless communication.

Internet computing is even broader and covers all computing paradigms over the Internet.

3)Discuss the different degrees of parallelism.

Fifty years ago, when hardware was bulky and expensive, most computers were designed in a bit-serial fashion. In this scenario, ***bit-level parallelism (BLP)*** converts bit-serial processing to word-level processing gradually.

Over the years, users graduated from 4-bit microprocessors to 8-, 16-, 32-, and 64-bit CPUs. This led us to the next wave of improvement, known as ***instruction-level parallelism (ILP)***, in which the processor executes multiple instructions simultaneously rather than only oneinstruction at a time. For the past 30 years, we have practiced ILP through pipelining, superscalar computing, *VLIW (very long instruction word)* architectures, and multithreading. ILP requires branch prediction, dynamic scheduling, speculation, and compiler support to work efficiently.

Data-level parallelism (DLP) was made popular through *SIMD (single instruction, multiple data)* and vector machines using vector or array types of instructions. DLP requires even more hardware support and compiler assistance to work properly.

Ever since the introduction of multicore processors and *chip multiprocessors (CMPs)*, we have been exploring ***task-level parallelism*** *(TLP).* A modern processor explores all of the aforementioned parallelism types. In fact, BLP, ILP, and DLP are well supported by advances in hardware and compilers. However, TLP is far from being very successful due to difficulty in programming and compilation of code for efficient execution on multicore CMPs.

As we move from parallel processing to distributed processing, we will see an increase in computing granularity to ***job-level parallelism (JLP).*** It is fair to say that coarsegrain parallelism is built on top of fine-grain parallelism.

4)What is Internet of Things? What is Cyber physical system? Explain.

The **IoT** refers to the networked interconnection of everyday objects, tools, devices, or computers. One can view the IoT as a wireless network of sensors that interconnect all things in our daily life. These things can be large or small and they vary with respect to time and place. The idea is to tag every object using RFID or a related sensor or electronic technology such as GPS.

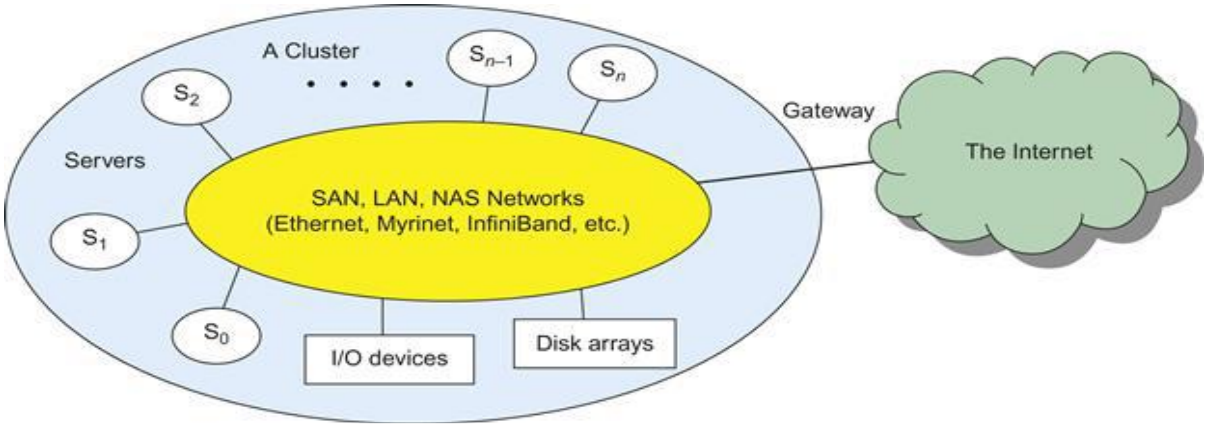
In the IoT era, all objects and devices are instrumented, interconnected, and interacted with each other intelligently. This communication can be made between people and things or among the things themselves. Three communication patterns co-exist: namely H2H (human-to-human), H2T (human-to-thing), and T2T (thing-to-thing). Here things include machines such as PCs and mobile phones. The idea here is to connect things (including human and machine objects) at any time and any place intelligently with low cost. Any place connections include at the PC, indoor (away from PC), outdoors, and on the move. Any time connections include daytime, night, outdoors and indoors, and on the move as well. The dynamic connections will grow exponentially into a new dynamic network of networks, called the *Internet of Things* (IoT). The IoT is still in its infancy stage of development. Many prototype IoTs with restricted areas of coverage are under experimentation at the time of this writing.

A ***cyber-physical system*** **(CPS)** is the result of interaction between computational processes and the physical world. A CPS integrates "cyber" (heterogeneous, asynchronous) with "physical" (concurrent and information-dense) objects. A CPS merges the "3C" technologies of *computation*, *communication*, and *control* into an intelligent closed feedback system between the physical world and the information world

5)What is a computer cluster? Explain the architecture of cluster. What are the designs issues in a cluster? What are the feasible implementation of various features of computer clusters?

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

architecture of a typical server cluster



Critical Cluster Design Issues and Feasible Implementations

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

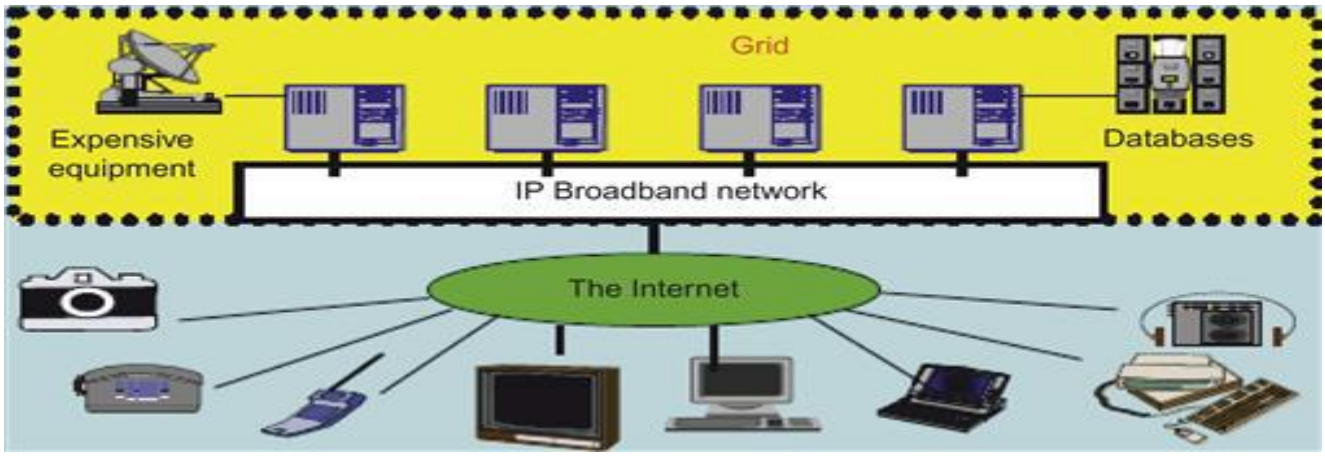
6)What is grid computing? Explain with example.

Grid computing is envisioned to allow close interaction among applications running on distant computers simultaneously.

Computational Grids

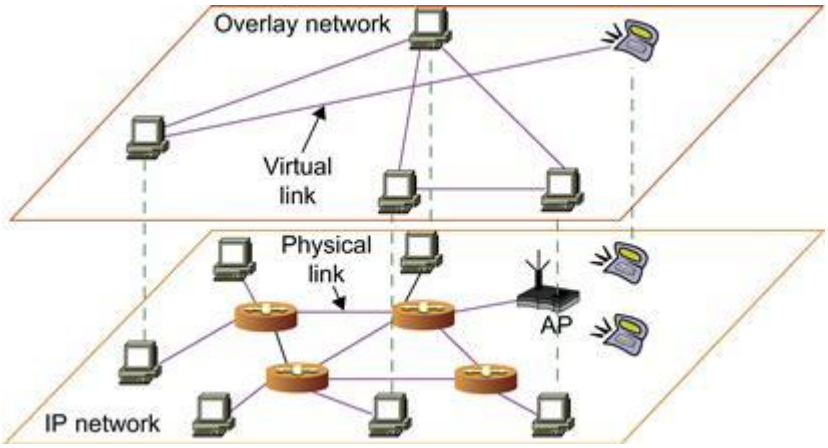
Like an electric utility power grid, a *computing grid* offers an infrastructure that couples computers, software/middleware, special instruments, and people and sensors together. The grid is often constructed across LAN, WAN, or Internet backbone networks at a regional, national, or global scale. Enterprises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

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



7)What are peer-to-peer systems and overlay networks? Explain.

peer-to-peer systems : In a P2P system, every node acts as both a client and a server, providing part of the system resources. Peer machines are simply client computers connected to the Internet. All client machines act autonomously to join or leave the system freely. Thisimpliesthatnomaster-slaverelationshipexistsamongthepeers.Nocentralcoordinationorcentraldatabaseisneeded. In other words, no peer machine has a global view of the entire P2P system. The system is self-organizing with distributed control.



overlay networks:

Data items or files are distributed in the participating peers. Based on communication or file-sharing needs, the peer IDs form an overlay network at the logical level. This overlay is a virtual network formed by mapping each physical machine with its ID, logically, through a virtual mapping.

There are two types of overlay networks: unstructured and structured.

An unstructured overlay network is characterized by a random graph. There is no fixed route to send messages or files among the nodes. Often, flooding is applied to send a query to all nodes in an unstructured overlay, thus resulting in heavy network traffic and nondeterministic search results.

Structured overlay networks follow certain connectivity topology and rules for inserting and removing nodes (peer IDs) from the overlay graph. Routing mechanisms are developed to take advantage of the structured overlays.

8)What is a cloud? What are internet clouds? Explain the three service models of cloud.

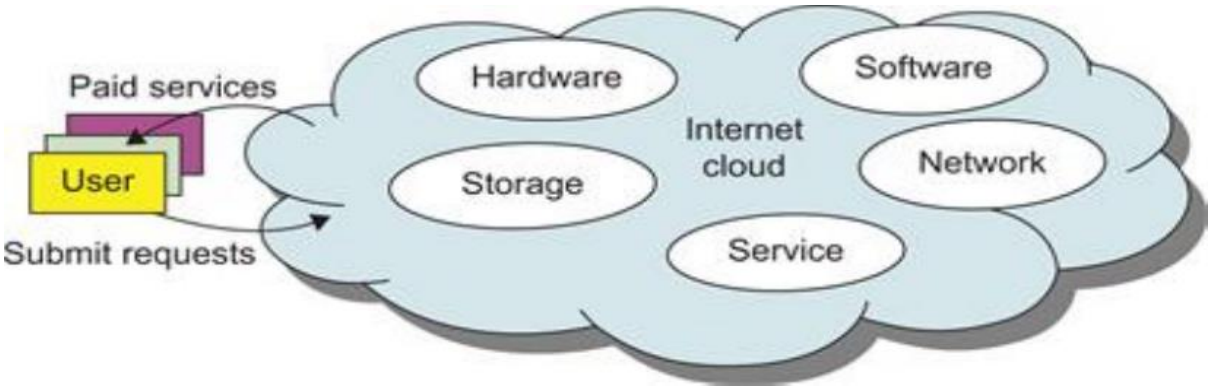
cloud : A cloud is a pool of virtualized computer resources. A cloud can host a variety of different workloads, including batch-style backend jobs and interactive and user-facing applications.

internet clouds :Cloud computing applies a virtualized platform with elastic resources on demand by provisioning hardware, software, and data sets dynamically.

The idea is to move desktop computing to a service-oriented platform using server clusters and huge databases at data centers. Cloud computing leverages its low cost and simplicity to benefit both users and providers. Machine virtualization has enabled such cost-effectiveness.

Cloud computing intends to satisfy many user applications simultaneously.

Virtualized resources from data centers to form an Internet cloud, provisioned with hardware, software, storage,network, and services for paid users to run their applications.



three service models of cloud:

Infrastructure as a Service (IaaS) This model puts together infrastructures demanded by users—namely servers, storage, networks, and the data center fabric. The user can deploy and run on multiple VMs running guest OSes on specific applications. The user does not manage or control the underlying cloud infrastructure, but can specify when to request and release the needed resources.

• *Platform as a Service (PaaS)* This model enables the user to deploy user-built applications onto a virtualized cloud platform. PaaS includes middleware, databases, development tools, and some runtime support such as Web 2.0 and Java. The platform includes both hardware and software integrated with specific programming interfaces. The provider supplies the API and software tools (e.g., Java, Python, Web 2.0, .NET). The user is freed from managing the cloud infrastructure.

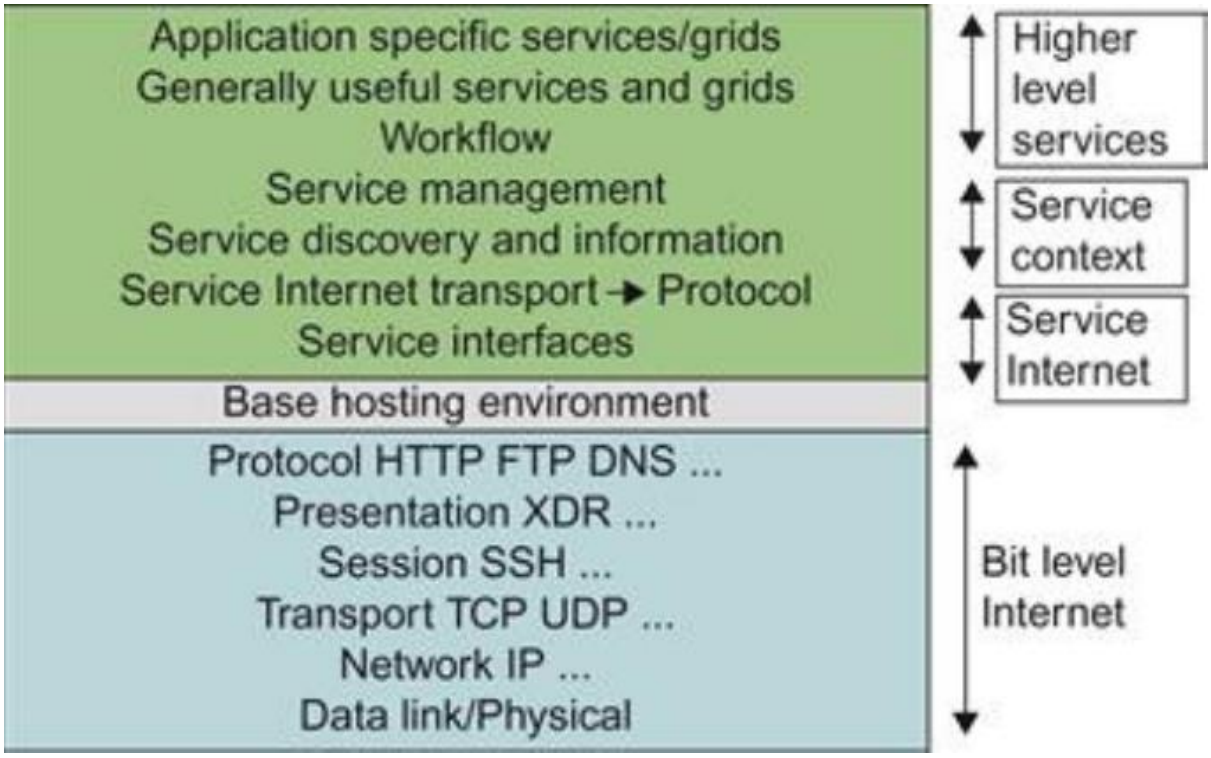
• *Software as a Service (SaaS)* This refers to browser-initiated application software over thousands of paid cloud customers. The SaaS model applies to business processes, industry applications, consumer relationship management (CRM), enterprise resources planning (ERP), human resources (HR), and collaborative applications. On the customer side, there is no upfront investment in servers or software licensing. On the provider side, costs are rather low, compared with conventional hosting of user applications.

9)Explain service oriented architecture. (or Explain the layered architecture for web services and grids.)

In grids/web services, Java, and CORBA, an entity is, respectively, a service, a Java object, and a CORBA distributed object in a variety of languages. These architectures build on the traditional seven Open Systems Interconnection (OSI) layers that provide the base networking abstractions.

On top of this we have a base software environment, which would be .NET or Apache Axis for web services, the Java Virtual Machine for Java, and a broker network for CORBA.

On top of this base environment one would build a higher level environment reflecting the special features of the distributed computing environment.



Layered achitecture for web services and the grids.

10)Discuss the evolution of service oriented architecture.

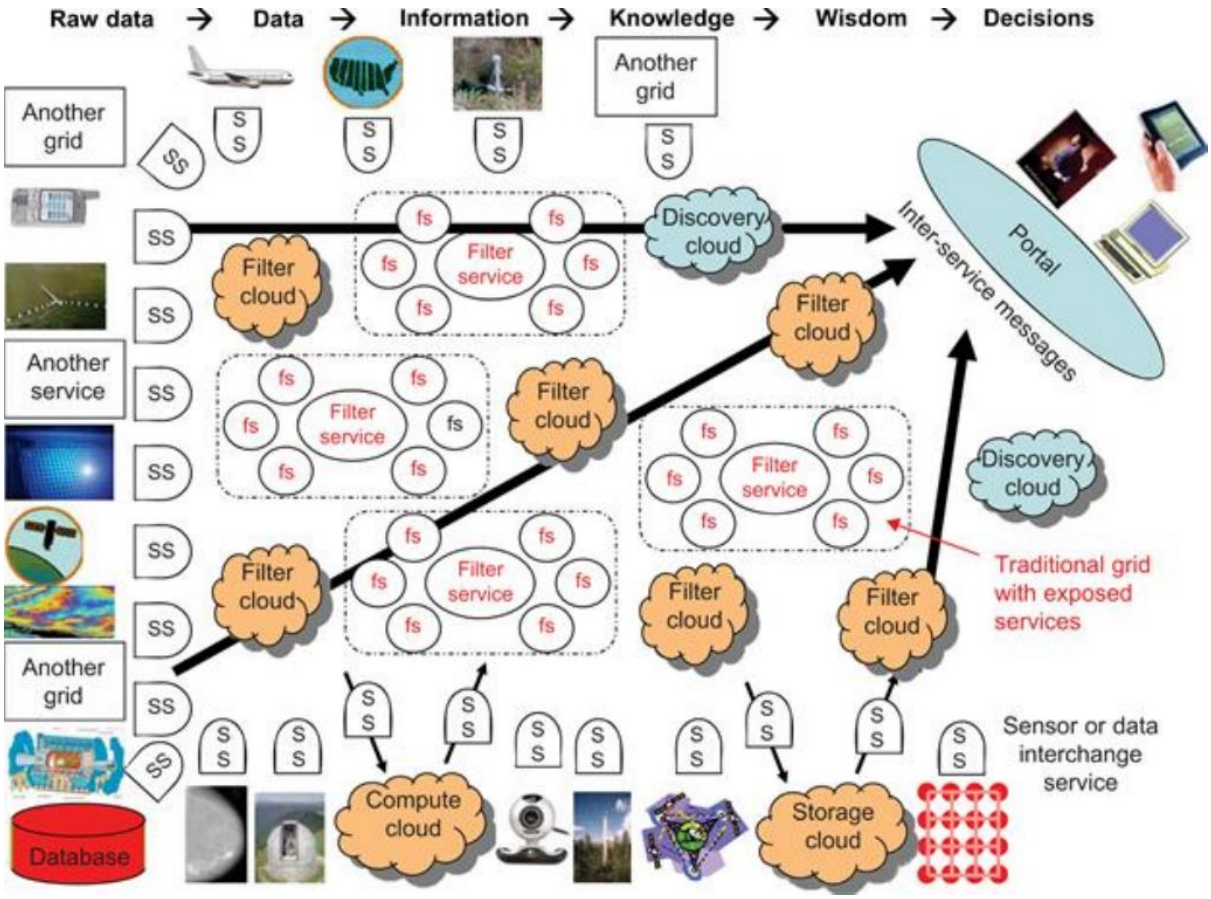
service-oriented architecture (SOA) has evolved over the years. SOA applies to building grids, clouds, grids of clouds, clouds of grids, clouds of clouds (also known as interclouds), and systems of systems in general.

A large number of sensors provide data-collection services, denoted in the figure as SS (sensor service). A sensor can be a ZigBee device, a Bluetooth device, a WiFi access point, a personal computer, a GPA, or a wireless phone, among other things.

Raw data is collected by sensor services.

All the SS devices interact with large or small computers, many forms of grids, databases, the compute cloud, the storage cloud, the filter cloud, the discovery cloud, and so on.

Filter services (fs in the figure) are used to eliminate unwanted raw data, in order to respond to specific requests from the web, the grid, or web services.



The evolution of SOA: grids of clouds and grids, where "SS" refers to a sensor service and "fs" to a filter or transforming service.

11)Compare grids and clouds.

A grid system applies static resources, while a cloud emphasizes elastic resources.

For some researchers, the differences between grids and clouds are limited only in dynamic resource allocation based on virtualization and autonomic computing.

One can build a grid out of multiple clouds. This type of grid can do a better job than a pure cloud, because it can explicitly support negotiated resource allocation.

Thus one may end up building with a system of systems: such as a cloud of clouds, a grid of clouds, or a cloud of grids, or inter-clouds as a basic SOA architecture.

12)Compare the following features of Amoeba, DCE and Mosix: History and current status, Distributed OS architecture, OS Kernel, Middleware and Virtualization support, Communication mechanisms.

Distributed OS Functionality	AMOEBA Developed at Vrije University [46]	DCE as OSF/1 by Open Software Foundation [7]	MOSIX for Linux Clusters at Hebrew University [3]
History and Current System Status	Written in C and tested in the European community; version 5.2 released in 1995	Built as a user extension on top of UNIX, VMS, Windows, OS/2, etc.	Developed since 1977, now called MOSIX2 used in HPC Linux and GPU clusters
Distributed OS Architecture	Microkernel-based and location-transparent, uses many servers to handle files, directory, replication, run, boot, and TCP/IP services	Middleware OS providing a platform for running distributed applications; The system supports RPC, security, and threads	A distributed OS with resource discovery, process migration, runtime support, load balancing, flood control, configuration, etc.
OS Kernel, Middleware, and Virtualization Support	A special microkernel that handles low-level process, memory, I/O, and communication functions	DCE packages handle file,time, directory, security services, RPC, and authentication at middleware or user space	MOSIX2 runs with Linux 2.6; extensions for use in multiple clusters and clouds with provisioned VMs
Communication Mechanisms	Uses a network-layer FLIP protocol and RPC to implement point-to-point and group communication	RPC supports authenticated communication and other security services in user programs	Using PVM, MPI in collective communications, priority process control, and queuing services

13)Explain the concept of transparent computing environments for computing platforms.

The user data, applications, OS, and hardware are separated into four levels.

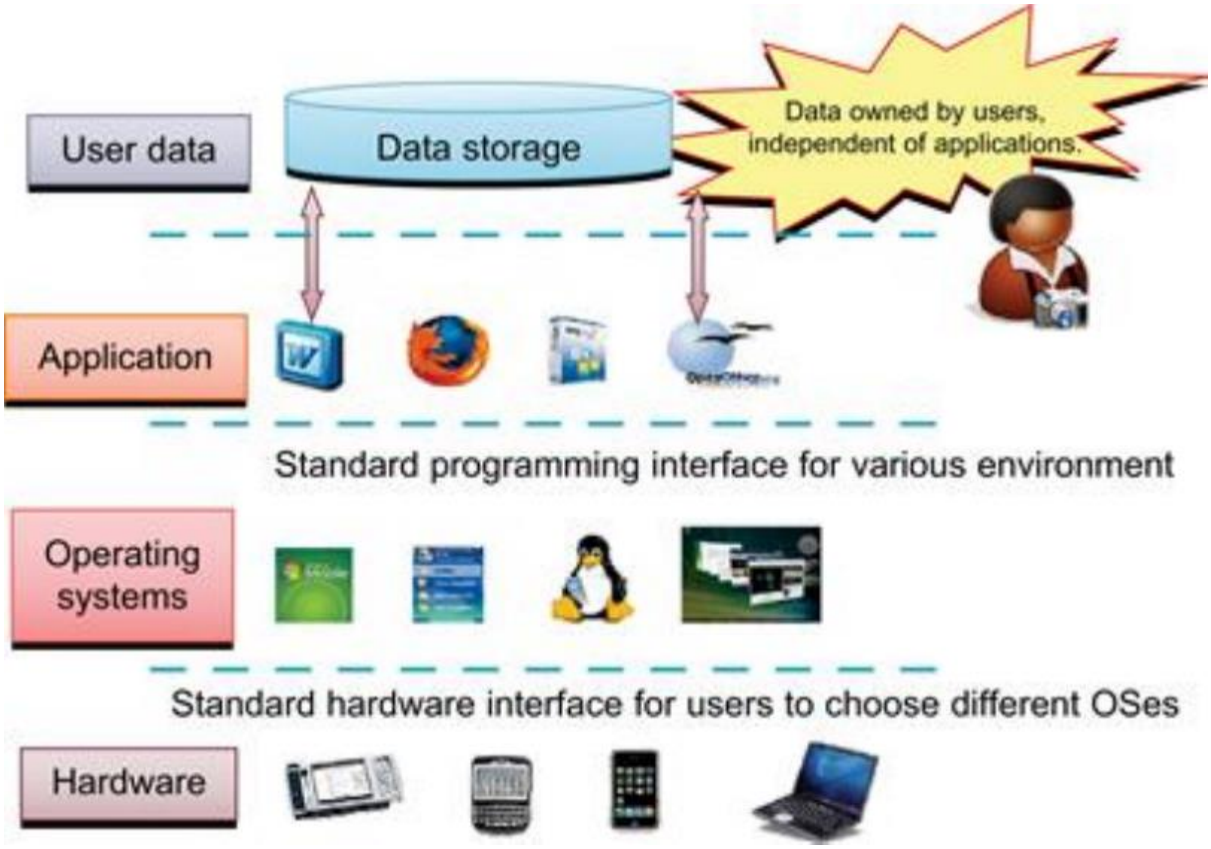
Data is owned by users, independent of the applications.

The OS provides clear interfaces, standard programming interfaces, or system calls to application programmers.

In future cloud infrastructure, the hardware will be separated by standard interfaces from the OS. Thus, users will be able to choose from different OSes on top of the hardware devices they prefer to use.

To separate user data from specific application programs, users can enable cloud applications as SaaS. Thus, users can switch among different services.

The data will not be bound to specific applications.



A transparent computing environment that separates the user data, application, OS, and hardware in time and space – an ideal model for cloud computing.

14)Explain the different programming models for parallel and distributed computing.

MPI is the most popular programming model for message-passing systems. Google's MapReduce and BigTable are for effective use of resources from Internet clouds and data centers. Service clouds demand extending Hadoop, EC2, and S3 to facilitate distributed computing over distributed storage systems.

Parallel and Distributed Programming Models and Tool Sets

Model	Description	Features
MPI	A library of subprograms that can be called from C or FORTRAN to write parallel programs running on distributed computer systems [6,28,42]	Specify synchronous or asynchronous point-to-point and collective communication commands and I/O operations in user programs for message-passing execution
MapReduce	A web programming model for scalable data processing on large clusters over large data sets, or in web search operations [16]	*Map* function generates a set of intermediate key/value pairs; *Reduce* function merges all intermediate values with the same key
Hadoop	A software library to write and run large user applications on vast data sets in business applications (http://hadoop.apache.org/core)	A scalable, economical, efficient, and reliable tool for providing users with easy access of commercial clusters

Message-Passing Interface (MPI): This is the primary programming standard used to develop parallel and concurrent programs to run on a distributed system. MPI is essentially a library of subprograms that can be called from C or FORTRAN to write parallel programs running on a distributed system. The idea is to embody clusters, grid systems, and P2P systems with upgraded web services and utility computing applications. Besides MPI, distributed programming can be also supported with low-level primitives such as the Parallel Virtual Machine (PVM).

MapReduce: This is a web programming model for scalable data processing on large clusters over large data sets. The model is applied mainly in web-scale search and cloud computing applications. The user specifies a Map function to generate a set of intermediate key/value pairs. Then the user applies a Reduce function to merge all intermediate values with the same intermediate key. MapReduce is highly scalable to explore high degrees of parallelism at different job levels. A typical MapReduce computation process can handle terabytes of data on tens of thousands or more client machines. Hundreds of MapReduce programs can be executed simultaneously; in fact, thousands of MapReduce jobs are executed on Google's clusters every day.

Hadoop Library: Hadoop offers a software platform that was originally developed by a Yahoo! group. The package enables users to write and run applications over vast amounts of distributed data. Users can easily scale Hadoop to store and process petabytes of data in the web space. Also, Hadoop is economical in that it comes with an open source version of MapReduce that minimizes overhead in task spawning and massive data communication. It is efficient, as it processes data with a high degree of parallelism across a large number of commodity nodes, and it is reliable in that it automatically keeps multiple data copies to facilitate redeployment of computing tasks upon unexpected system failures.

15)What are the different performance metrics in distributed systems? Enumerate the dimensions of scalability characterized in parallel and distributed systems.

In a distributed system, performance is attributed to a large number of factors. System throughput is often measured in MIPS, Tflops (tera floating-point operations per second), or TPS (transactions per second).

Other measures include job response time and network latency. An interconnection network that has low latency and high bandwidth is preferred. System overhead is often attributed to OS boot time, compile time, I/O data rate, and the runtime support system used.

Other performance related metrics include the QoS for Internet and web services; system availability and dependability; and security resilience for system defense against network attacks.

The following dimensions of scalability are characterized in parallel and distributed systems:

• Size scalability This refers to achieving higher performance or more functionality by increasing the machine size. The word "size" refers to adding processors, cache, memory, storage, or I/O channels. The most obvious way to determine size scalability is to simply count the number of processors installed. Not all parallel computer or distributed architectures are equally size-scalable. For example, the IBM S2 was scaled up to 512 processors in 1997. But in 2008, the IBM BlueGene/Lsystem scaled up to 65,000 processors.

• Software scalability This refers to upgrades in the OS or compilers, adding mathematical and engineering libraries, porting new application software, and installing more user-friendly programming environments. Some software upgrades may not work with large system configurations. Testing and fine-tuning of new software on larger systems is a nontrivial job.

• Application scalability This refers to matching problem size scalability with machine size scalability. Problem size affects the size of the data set or the workload increase. Instead of increasing machine size, users can enlarge the problem size to enhance system efficiency or cost-effectiveness.

• Technology scalability This refers to a system that can adapt to changes in building technologies, such as the component and networking technologies.When scaling a system design with new technology one must consider three aspects: time, space, and heterogeneity.

(1) Time refers to generation scalability. When changing to new-generation processors, one must consider the impact to the motherboard, power supply, packaging and cooling, and so forth. Based on past experience, most systems upgrade their commodity processors every three to five years.

(2) Space is related to packaging and energy concerns. Technology scalability demands harmony and portability among suppliers.

(3) Heterogeneity refers to the use of hardware components or software packages from different vendors. Heterogeneity may limit the scalability.

16)State Amdahl's Law. What is the problem with fixed load? How can it be overcome?

Consider the execution of a given program on a uniprocessor workstation with a total execution time of T minutes. Now, let's say the program has been parallelized or partitioned for parallel execution on a cluster of many processing nodes. Assume that a fraction α of the code must be executed sequentially, called the sequential bottleneck. Therefore, $(1 - \alpha)$ of the code can be compiled for parallel execution by n processors. The total execution time of the program is calculated by $\alpha T + (1 - \alpha)T/n$, where the first term is the sequential execution time on a single processor and the second term is the parallel execution time on n processing nodes. All system or communication overhead is ignored here. The I/O time or exception handling time is also not included.

Amdahl's Law states that the speedup factor of using the n-processor system over the use of a single processor is expressed by:

$$\text{Speedup} = S = T/[\alpha T + (1-\alpha)T/n] = 1/[\alpha + (1-\alpha)/n]$$

The maximum speedup of n is achieved only if the sequential bottleneck α is reduced to zero or the code is fully parallelizable with α = 0. As the cluster becomes sufficiently large, that is, $n \to \infty$, S approaches 1/α, an upper bound on the speedup S. Surprisingly, this upper bound is independent of the cluster size n. The sequential bottleneck is the portion of the code that cannot be parallelized.

Problem with Fixed Workload : In Amdahl's law, we have assumed the same amount of workload for both sequential and parallel execution of the program with a fixed problem size or data set. This was called fixed-workload speedup.

To execute a fixed work-load on n processors, parallel processing may lead to a system efficiency defined as follows:

$$E = S/n = 1/[\alpha n + 1 - \alpha]$$

Very often the system efficiency is rather low, especially when the cluster size is very large. To execute the aforementioned program on a cluster with n = 256 nodes, extremely low efficiency E = 1/[0.25 × 256 + 0.75] = 1.5% is observed. This is because only a few processors (say, 4) are kept busy, while the majority of the nodes are left idling.

To solve scaled problems, users should apply Gustafson's law.

To achieve higher efficiency when using a large cluster, we must consider scaling the problem size to match the cluster capability.This leads to the following speedup law proposed by John Gustafson, referred as scaled-workload speedup. Let W be the workload in a given program. When using an n-processor system, the user scales the workload to $W' = \alpha W + (1-\alpha)nW$. Note that only the parallelizable portion of the workload is scaled n times in the second term. This scaled workload W′ is essentially the sequential execution time on a single processor. The parallel execution time of a scaled workload W′ on n processors is defined by a scaled-workload speedup as follows:

$$S' = W'/W = [\alpha W + (1-\alpha)nW]/W = \alpha + (1-\alpha)n$$

This speedup is known as Gustafson's law. By fixing the parallel execution time at level W, the following efficiency expression is obtained:

$$E' = S'/n = \alpha/n + (1-\alpha)$$

17)Explain system availability and application flexibility design goals in distributed computing system.

HA (high availability) is desired in all clusters, grids, P2P networks, and cloud systems. A system is highly available if it has a long mean time to failure (MTTF) and a short mean time to repair (MTTR). System availability is formally defined as follows:

$$System\ Availability = MTTF/(MTTF + MTTR)$$

System availability is attributed to many factors. All hardware, software, and network components may fail. Any failure that will pull down the operation of the entire system is called a single point of failure.

Adding hardware redundancy, increasing component reliability, and designing for testability will help to enhance system availability and dependability.

In general, as a distributed system increases in size, availability decreases due to a higher chance of failure and a difficulty in isolating the failures.

18)Explain the different threats to systems and networks in cyberspace.

Information leaks lead to a loss of confidentiality. Loss of data integrity may be caused by user alteration, Trojan horses, and service spoofing attacks.

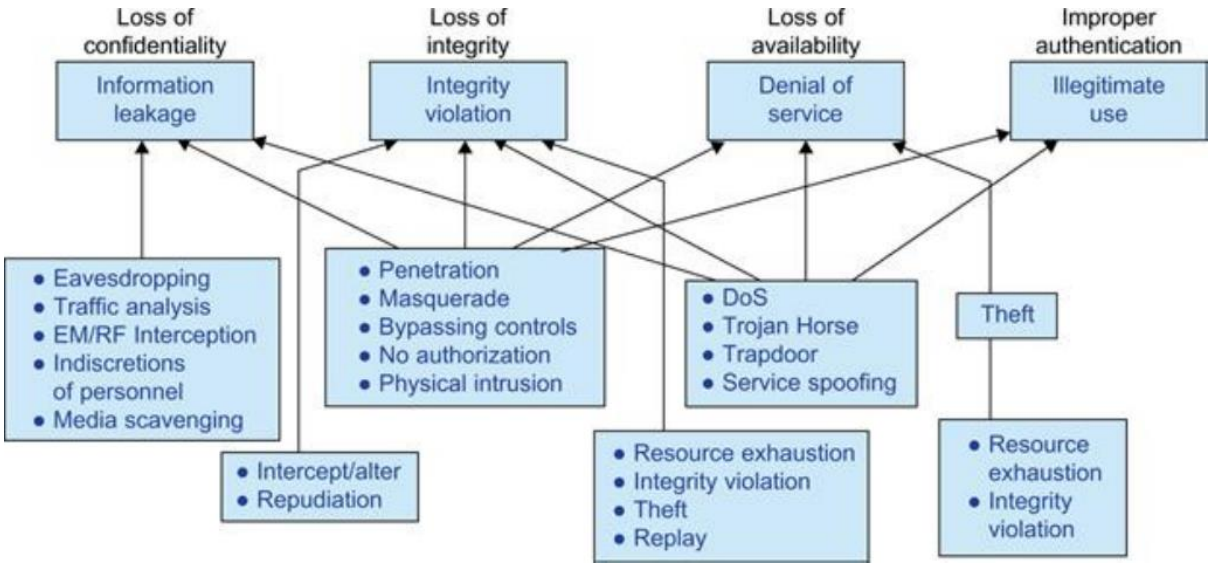
A denial of service (DoS) results in a loss of system operation and Internet connections.

Lack of authentication or authorization leads to attackers' illegitimate use of computing resources. Open resources such as data centers, P2P networks, and grid and cloud infrastructures could become the next targets.

Users need to protect clusters, grids, clouds, and P2P systems. Otherwise, users should not use or trust them for outsourced work.

Malicious intrusions to these systems may destroy valuable hosts, as well as network and storage resources.

Internet anomalies found in routers, gateways, and distributed hosts may hinder the acceptance of these public-resource computing services.



19)How can energy efficiency be achieved in different layers distributed computing? Explain the dynamic power management and dynamic frequency voltage scaling methods incorporated into hardware systems.

Application Layer: Until now, most user applications in science, business, engineering, and financial areas tend to increase a system's speed or quality. By introducing energy-aware applications, the challenge is to design sophisticated multilevel and multi-domain energy management applications without hurting performance. The first step toward this end is to explore a relationship between performance and energy consumption. Indeed, an application's energy consumption depends strongly on the number of instructions needed to execute the application and the number of transactions with the storage unit (or memory). These two factors (compute and storage) are correlated and they affect completion time.

Middleware Layer: The middleware layer acts as a bridge between the application layer and the resource layer. This layer provides resource broker, communication service, task analyzer, task scheduler, security access, reliability control, and information service capabilities. It is also responsible for applying energy-efficient techniques, particularly in task scheduling. Until recently, scheduling was aimed at minimizing makespan, that is, the execution time of a set of tasks. Distributed computing systems necessitate a new cost function covering both makespan and energy consumption.

Resource Layer: The resource layer consists of a wide range of resources including computing nodes and storage units. This layer generally interacts with hardware devices and the operating system; therefore, it is responsible for controlling all distributed resources in distributed computing systems. In the recent past, several mechanisms have been developed for more efficient power management of hardware and operating systems. The majority of them are hardware approaches particularly for processors.

Dynamic power management (DPM) and dynamic voltage-frequency scaling (DVFS) are two popular methods incorporated into recent computer hardware systems.

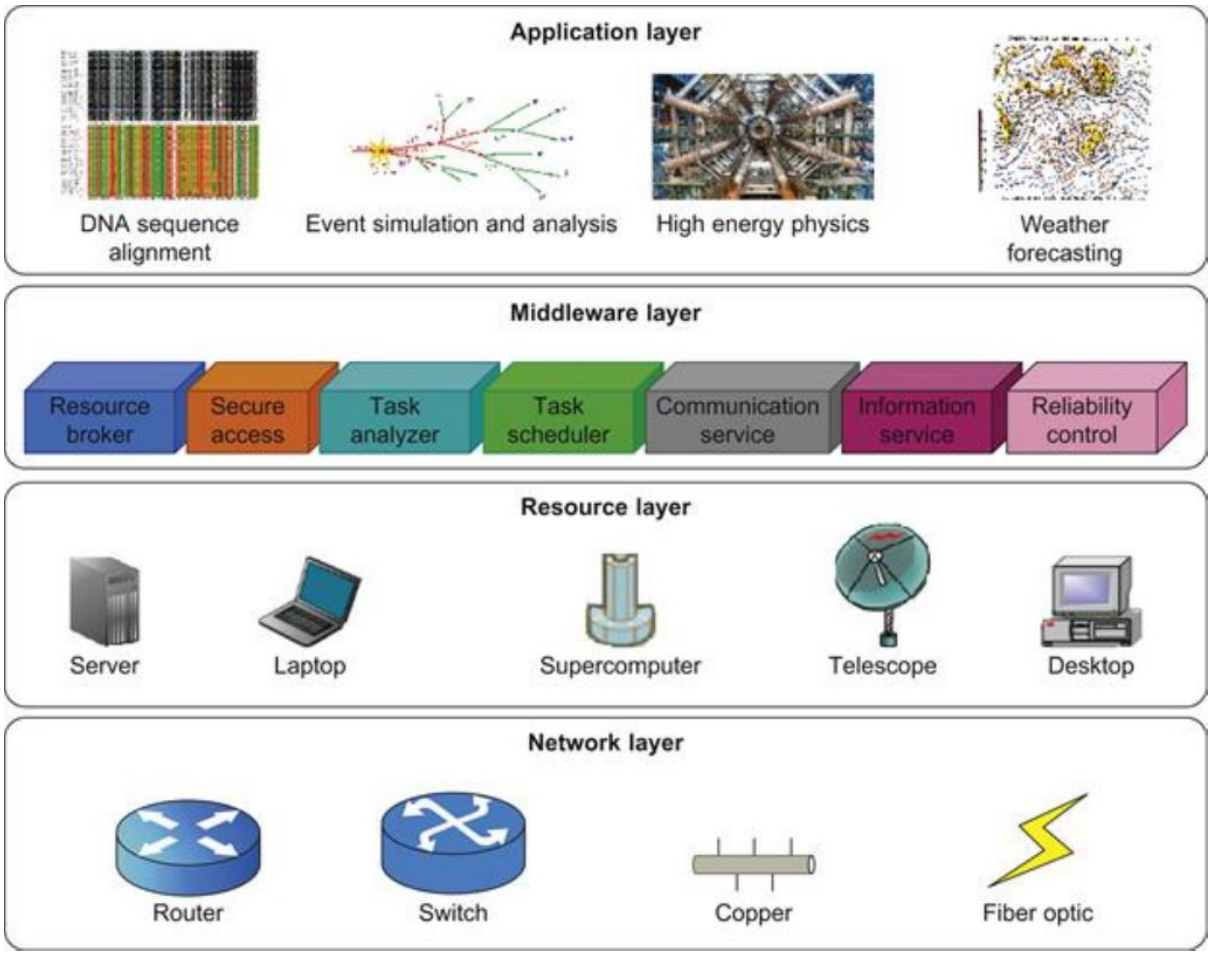
In DPM, hardware devices, such as the CPU, have the capability to switch from idle mode to one or more lower-power modes.

In DVFS, energy savings are achieved based on the fact that the power consumption in CMOS circuits has a direct relationship with frequency and the square of the voltage supply. Execution time and power consumption are controllable by switching among different frequencies and voltages.

Network Layer: Routing and transferring packets and enabling network services to the resource layer are the main responsibility of the network layer in distributed computing systems. The major challenge to build energy-efficient networks is, again, determining how to measure, predict, and create a balance between energy consumption and performance. Two major challenges to designing energy-efficient networks are:

• The models should represent the networks comprehensively as they should give a full understanding of interactions among time, space, and energy.

• New, energy-efficient routing algorithms need to be developed. New, energy-efficient protocols should be developed against network attacks.



20)Consider a multicore processor with four heterogeneous cores labelled A, B, C, and D. Assume cores A and D have the same speed. Core B runs twice as fast as core A, and core C runs three times faster than core A. Assume that all four cores start executing the following application at the same time and no cache misses are encountered in all core operations. Suppose an application needs to compute the square of each element of an array of 256 elements. Assume 1 unit time for core A or D to compute the square of an element. Thus, core B takes ½ unit time and core C takes ⅓ unit time to compute the square of an element. Given the following division of labour in four cores.

Core A 32 elements

Core B 128 elements

Core C 64 elements

Core D 32 elements

a. Compute the total execution time (in time units) for using the four-core processor to compute the squares of 256 elements in parallel. The four cores have different speeds. Some faster cores finish the job and may become idle, while others are still busy computing until all squares are computed.

b. Calculate the processor utilization rate, which is the total amount of time the cores are busy (not idle) divided by the total execution time they are using all cores in the processor to execute the above application.

(a) The completion times for core A, B, C and D are 32/1, 128/2, 64/3, 32/1 respectively. Therefore, the total execution time is 128/2=64 time units

(b) The processor utilization rate is:

(32+64+64/3+32) / (64x4) = 149.33/256 = 58.3%

21)Consider a program for multiplying two large-scale N × N matrices, where N is the matrix size. The sequential multiply time on a single server is T1 = cN3 minutes, where c is a constant determined by the server used. An MPI-code parallel program requires Tn = cN3/n + dN2/n0.5 minutes to complete execution on an n-server cluster system, where d is a constant determined by the MPI version used. Assume the program has a zero sequential bottleneck (α = 0). The second term in Tn accounts for the total message-passing overhead experienced by n servers. Answer the following questions for a given cluster configuration with n = 64 servers, c = 0.8, and d = 0.1. Parts (a, b) have a fixed workload corresponding to the matrix size N = 15,000. Parts (c, d) have a scaled workload associated with an enlarged matrix size N′ = n1/3 N = 641/3 × 15,000 = 4 × 15,000 = 60,000. Assume the same cluster configuration to process both workloads.

Thus, the system parameters n, c, and d stay unchanged. Running the scaled workload, the overhead also increases with the enlarged matrix size N′.

a. Using Amdahl's law, calculate the speedup of the n-server cluster over a single server.

b. What is the efficiency of the cluster system used in Part (a)?

c. Calculate the speedup in executing the scaled workload for an enlarged N′ × N′ matrix on the same cluster configuration using Gustafson's law.

d. Calculate the efficiency of running the scaled workload in Part (c) on the 64-processor cluster.

e. Compare the above speedup and efficiency results and comment on their implications.

a) Speedup, S = 1/[α + (1 - α) / n] Since, α = 0 and n = 64, Speedup, S = 64

b) Efficiency, E = S / n = 1 = 100%

c) Speedup, S' = Time taken by a single server / Time taken by the cluster

= cN'3 / [(cN'3 / n) + (dN'2 / n0.5)]

c = 0.8, d = 0.1, N' = n1/3N, N = 15,000, n = 64

S' = 0.8 x 64 x N3 / [0.2N2(4N + 1)] = 256N / (4N + 1)

Since, 4N >> 1, S' = 256N / 4N = 64

d) Efficiency, E' = α / n + (1 – α), Sinceα = 0, E' = 1 = 100%

e) • For both cases i.e. fixed workload and scaled workload the speedup and efficiency is the same i.e. S = 64 and E = 100%

• The bottleneck for speedup even after using a large number of parallel cores is the part of sequential code α. So even if we have a large cluster we will not have maximum speedup due to α. Also the efficiency is very low when we use a fixed workload.

• We use scaled workload to improve the efficiency. However it is still not 100% due to α.

• In the above 2 cases since α = 0, we are able to get the maximum speedup of 64 and efficiency of 100% for both fixed and scalable workloads.

Unit 2

Sr no	Questions and answers
22	***Explain the six orthogonal attributes on which computer clusters are classified.*** **Ans:** Clusters have been classified in various ways in the literature. We classify clusters using six orthogonal attributes: 1) scalability, 2) packaging, 3) control, 4) homogeneity, 5) programmability, and 6) security. ➢ Scalability Clustering of computers is based on the concept of modular growth. To scale a cluster from hundreds of uniprocessor nodes to a supercluster with 10,000 multicore nodes is a nontrivial task. The scalability could be limited by a number of factors, such as the multicore chip technology, cluster topology, packaging method, power consumption, and cooling scheme applied. The purpose is to achieve scalable performance constrained by the aforementioned factors. We have to also consider other limiting factors such as the memory wall, disk I/O bottlenecks, and latency tolerance, among others. ➢ Packaging Cluster nodes can be packaged in a compact or a slack fashion. In a compact cluster, the nodes are closely packaged in one or more racks sitting in a room, and the nodes are not attached to peripherals (monitors, keyboards, mice, etc.). In a slack cluster, the nodes are attached to their usual peripherals (i.e., they are complete SMPs, workstations, and PCs), and they may be located in different rooms, different buildings, or even remote regions. Packaging directly affects communication wire length, and thus the selection of interconnection technology used. While a compact cluster can utilize a high-bandwidth, low-latency communication network that is often proprietary, nodes of a slack cluster are normally connected through standard LANs or WANs. ➢ Control A cluster can be either controlled or managed in a centralized or decentralized fashion. A compact cluster normally has centralized control, while a slack cluster can be controlled either way. In a centralized cluster, all the nodes are owned, controlled, managed, and administered by a central operator. In a decentralized cluster, the nodes have individual owners. For instance, consider a cluster comprising an interconnected set of desktop workstations in a department, where each workstation is individually owned by an employee. The owner can reconfigure, upgrade, or even shut down the workstation at any time. This lack of a single point of control makes system administration of such a cluster very difficult. It also calls for special techniques for

	process scheduling, workload migration, checkpointing, accounting, and other similar tasks. ➢ Homogeneity A homogeneous cluster uses nodes from the same platform, that is, the same processor architecture and the same operating system; often, the nodes are from the same vendors. A heterogeneous cluster uses nodes of different platforms. Interoperability is an important issue in heterogeneous clusters. For instance, process migration is often needed for load balancing or availability. In a homogeneous cluster, a binary process image can migrate to another node and continue execution. This is not feasible in a heterogeneous cluster, as the binary code will not be executable when the process migrates to a node of a different platform. ➢ Security Intracluster communication can be either exposed or enclosed. In an exposed cluster, the communication paths among the nodes are exposed to the outside world. An outside machine can access the communication paths, and thus individual nodes, using standard protocols (e.g., TCP/IP). Such exposed clusters are easy to implement, but have several disadvantages: • Being exposed, intracluster communication is not secure, unless the communication subsystem performs additional work to ensure privacy and security. • Outside communications may disrupt intracluster communications in an unpredictable fashion. For instance, heavy BBS traffic may disrupt production jobs. • Standard communication protocols tend to have high overhead. In an enclosed cluster, intracluster communication is shielded from the outside world, which alleviates the aforementioned problems. A disadvantage is that there is currently no standard for efficient, enclosed intracluster communication. Consequently, most commercial or academic clusters realize fast communications through one-of-a-kind protocols.
23	***. Discuss the different issues in developing and using a computer cluster*** **Ans:** **Fundamental Cluster Design Issues** we will identify the major design issues of clustered and MPP systems. Both physical and virtual clusters are covered. These systems are often found in computational grids,

national laboratories, business data centers , supercomputer sites, and virtualized cloud platforms. Several issues must be considered in developing and using a cluster.

- **Scalable Performance**

This refers to the fact that scaling of resources (cluster nodes, memory capacity, I/O bandwidth, etc.) leads to a proportional increase in performance. Of course, both scale-up and scale-down capabilities are needed, depending on application demand or cost-effectiveness considerations. Clustering is driven by scalability. One should not ignore this factor in all applications of cluster or MPP computing systems.

- **Single-System Image (SSI)**

A set of workstations connected by an Ethernet network is not necessarily a cluster. A cluster is a single system. For example, suppose a workstation has a 300 Mflops/second processor, 512 MB of memory, and a 4 GB disk and can support 50 active users and 1,000 processes. By clustering 100 such workstations, can we get a single system that is equivalent to one huge workstation, or a megastation, that has a 30 Gflops/second processor, 50 GB of memory, and a 400 GB disk and can

support 5,000 active users and 100,000 processes? This is an appealing goal, but it is very difficult to achieve. SSI techniques are aimed at achieving this goal.

- **Availability Support**

Clusters can provide cost-effective HA capability with lots of redundancy in processors, memory, disks, I/O devices, networks, and operating system images. However, to realize this potential, availability techniques are required.

- **Cluster Job Management**

Clusters try to achieve high system utilization from traditional workstations or PC nodes that are normally not highly utilized. Job management software is required to provide batching, load balancing, parallel processing, and other functionality.

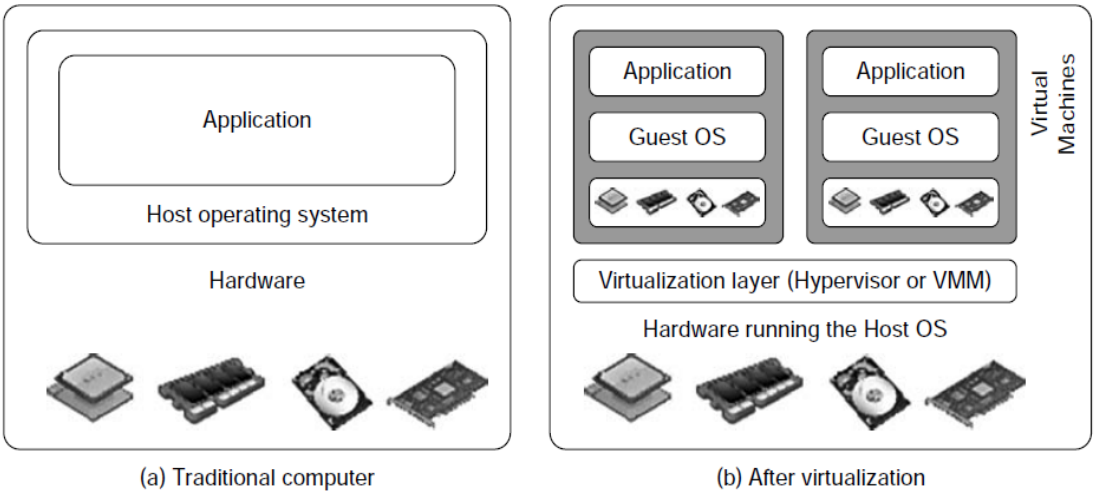
Special software tools are needed to manage multiple jobs simultaneously.

- **Internode Communication**

Because of their higher node complexity, cluster nodes cannot be packaged as compactly as MPP nodes. The internode physical wire lengths are longer in a cluster than in an MPP. This is true even for centralized clusters. A long wire implies greater interconnect network latency. But more importantly, longer wires have more problems in terms of reliability, clock skew, and cross talking. These problems call for reliable

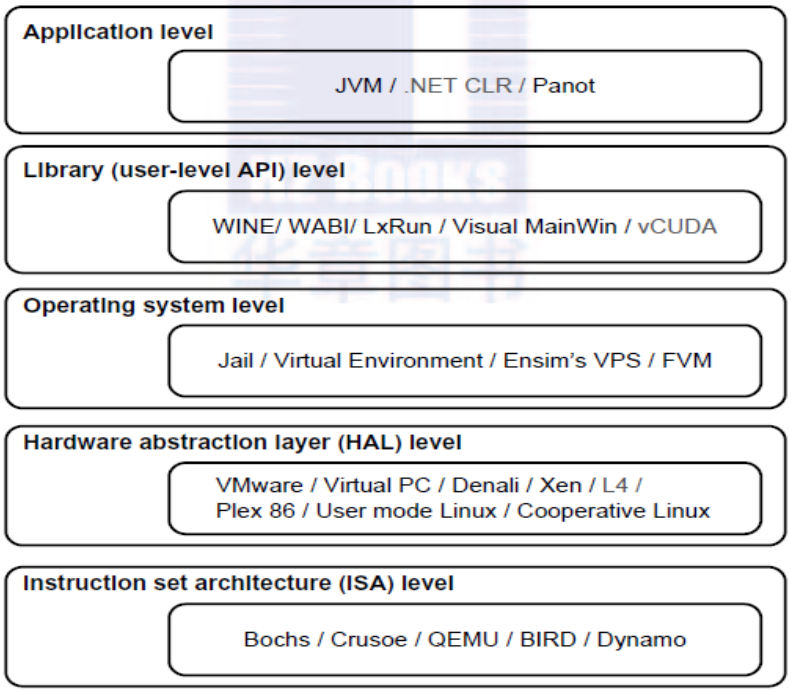
	and secure communication protocols, which increase overhead. Clusters often use commodity networks (e.g., Ethernet) with standard protocols such as TCP/IP. ➢ **Fault Tolerance and Recovery** Clusters of machines can be designed to eliminate all single points of failure. Through redundancy, a cluster can tolerate faulty conditions up to a certain extent. Heartbeat mechanisms can be installed to monitor the running condition of all nodes. In case of a node failure, critical jobs running on the failing nodes can be saved by failing over to the surviving node machines. Rollback recovery schemes restore the computing results through periodic checkpointing.
24	***Explain the different levels of virtualization implementation along with relative merits of each level.*** **Ans:** Virtualization is a computer architecture technology by which multiple virtual machines (VMs) are multiplexed in the same hardware machine. Hardware resources (CPU, memory, I/O devices, etc.) or software resources (operating system and software libraries) can be virtualized in various functional layers. This virtualization technology has been revitalized as the demand for distributed and cloud computing increased sharply in recent years **Levels of Virtualization Implementation** A traditional computer runs with a host operating system specially tailored for its hardware architecture, as shown in Figure.1(a). After virtualization, different user applications managed by their own operating systems (guest OS) can run on the same hardware, independent of the host OS. This is often done by adding additional software, called a virtualization layer as shown in Figure 1(b). This virtualization layer is known as hypervisor or virtual machine monitor (VMM) The VMs are shown in the upper boxes, where applications run with their own guest OS over the virtualized CPU, memory, and I/O resources. The main function of the software layer for virtualization is to virtualize the physical hardware of a host machine into virtual resources to be used by the VMs, exclusively. This can be implemented at various operational levels, as we will discuss shortly. The virtualization software creates the abstraction of VMs by interposing a virtualization layer at various levels of a computer system. Common virtualization layers include the instruction set architecture (ISA) level, hardware level, operating system level, library support level, and application level (see Figure .2).

Figure 1



The architecture of a computer system before and after virtualization, where VMM stands for virtual machine monitor.

Figure 2



Virtualization ranging from hardware to applications in five abstraction levels.

- **Instruction Set Architecture Level**

At the ISA level, virtualization is performed by emulating a given ISA by the ISA of the host machine. For example, MIPS binary code can run on an x86-based host machine with the help of ISA emulation. With this approach, it is possible to run a large amount of legacy binary code written for various processors on any given new hardware host machine. Instruction set emulation leads to virtual ISAs created on any hardware machine. The basic emulation method is through code interpretation. An interpreter program interprets the source instructions to target instructions one by one. One source instruction may require tens or hundreds of native target instructions to perform its function. Obviously, this process is relatively slow. For better performance, dynamic binary translation is desired. This approach translates basic blocks of dynamic source instructions to target instructions. The basic blocks can also be extended

to program traces or super blocks to increase translation efficiency. Instruction set emulation requires binary translation and optimization. A virtual instruction set architecture (V-ISA) thus requires adding a processor-specific software translation layer to the compiler.

- **Hardware Abstraction Level**

Hardware-level virtualization is performed right on top of the bare hardware. On the one hand, this approach generates a virtual hardware environment for a VM. On the other hand, the process manages the underlying hardware through virtualization. The idea is to virtualize a computer's resources, such as its processors, memory, and I/O devices. The intention is to upgrade the hardware utilization rate by multiple users concurrently. The idea was implemented in the IBM VM/370 in the 1960s. More recently, the Xen hypervisor has been applied to virtualize x86-based machines to run Linux or other

guest OS applications.

- **Operating System Level**

This refers to an abstraction layer between traditional OS and user applications. OS-level virtualization creates isolated containers on a single physical server and the OS instances to utilize the hardware and software in data centers. The containers behave like real servers. OS-level virtualization is commonly used in creating virtual hosting environments to allocate hardware resources among a large number of mutually distrusting users. It is also used, to a lesser extent, in consolidating server

hardware by moving services on separate hosts into containers or VMs on one server.

.

- **Library Support Level**

Most applications use APIs exported by user-level libraries rather than using lengthy system calls by the OS. Since most systems provide well-documented APIs, such an interface becomes another candidate for virtualization. Virtualization with library interfaces is possible by controlling the communication link between applications and

the rest of a system through API hooks. The software tool WINE has implemented this approach to support Windows applications on top of UNIX hosts.

Another example is the vCUDA which allows applications executing within VMs to leverage GPU hardware acceleration.

- **User-Application Level**

Virtualization at the application level virtualizes an application as a VM. On a traditional OS, an application often runs as a process. Therefore, application-level virtualization is also known as process-level virtualization. The most popular approach is to deploy high level language (HLL) VMs. In this scenario, the virtualization layer sits as an application program on top of the operating system, and the layer exports an abstraction of a VM that can run programs written and compiled to a particular abstract machine definition. Any program written in the HLL and compiled for this VM will be able to run on it. The Microsoft .NET CLR and Java Virtual Machine (JVM) are two

good examples of this class of VM.

Other forms of application-level virtualization are known as application isolation, application sandboxing, or application streaming. The process involves wrapping the application in a layer that is isolated from the host OS and other applications. The result is an application that is much easier to distribute and remove from user workstations. An example is the LANDesk application virtualization platform which deploys software applications as self-contained, executable files in an isolated environment without requiring installation, system modifications, or elevated security privileges.

- **Relative Merits of Different Approaches**

Following table compares the relative merits of implementing virtualization at various levels. The column headings correspond to four technical merits. “Higher Performance” and “Application Flexibility” are self-explanatory. “Implementation Complexity” implies the cost to implement that particular virtualization level. “Application Isolation” refers to the effort required to isolate resources committed to different VMs. Each row corresponds to a particular level of virtualization.

The number of X’s in the table cells reflects the advantage points of each implementation level. Five X’s implies the best case and one X implies the worst case. Overall, hardware and OS support will yield the highest performance. However, the hardware and application levels are also the most expensive to implement. User isolation is the most difficult to achieve. ISA implementation offers

the best application flexibility.

Relative Merits of Virtualization at Various Levels (More "X"'s Means Higher Merit, with a Maximum of 5 X's)

Level of Implementation	Higher Performance	Application Flexibility	Implementation Complexity	Application Isolation
ISA	X	XXXXX	XXX	XXX
Hardware-level virtualization	XXXXX	XXX	XXXXX	XXXX
OS-level virtualization	XXXXX	XX	XXX	XX
Runtime library support	XXX	XX	XX	XX
User application level	XX	XX	XXXXX	XXXXX

25 ***What are the requirements for design of virtual machine monitor? Explain.***

Ans: **VMM Design Requirements**

As we know hardware-level virtualization inserts a layer between real hardware and traditional operating systems. This layer is commonly called the Virtual Machine Monitor (VMM) and it manages the hardware resources of a computing system. Each time programs access the hardware the VMM captures the process. In this sense, the VMM acts as a traditional OS. One hardware component, such as the CPU, can be virtualized as several virtual copies. Therefore, several traditional operating systems which are the same or different can sit on the same set of hardware simultaneously.

There are three requirements for a VMM.

First, a VMM should provide an environment for programs which is essentially identical to the original machine.

Second, programs run in this environment should show, at worst, only minor decreases in speed.

Third, a VMM should be in complete control of the system resources. Any program run under a VMM should exhibit a function identical to that which it runs on the original machine directly.

Two possible exceptions in terms of differences

are permitted with this requirement: differences caused by the availability of system resources and differences caused by timing dependencies. The former arises when more than one VM is running on the same machine

.

The hardware resource requirements, such as memory, of each VM are reduced, but the sum of them is greater than that of the real machine installed. The latter qualification is required because of the intervening level of software and the effect of any other VMs concurrently existing on the same hardware. Obviously, these two differences pertain to performance, while the function a VMM provides stays the same

as that of a real machine. However, the identical environment requirement excludes the behavior of the usual time-sharing operating system from being classed as a VMM.

A VMM should demonstrate efficiency in using the VMs. Compared with a physical machine, no one prefers a VMM if its efficiency is too low. Traditional emulators and complete software interpreters (simulators) emulate each instruction by means of functions or macros. Such a method provides the most flexible solutions for VMMs. However, emulators or simulators are too slow to be used as real machines. To guarantee the efficiency of a VMM, a statistically dominant subset of the virtual processor's instructions needs to be executed directly by the real processor, with no

software intervention by the VMM. Following table compares four hypervisors and VMMs that are in use today.

Complete control of these resources by a VMM includes the following aspects:

(1) The VMM is responsible for allocating hardware resources for programs;

(2) it is not possible for a program to access any resource not explicitly allocated to it; and (3) it is possible under certain circumstances for a VMM to regain control of resources already allocated. Not all processors satisfy these requirements

for a VMM. A VMM is tightly related to the architectures of processors. It is difficult to implement a VMM for some types of processors, such as the x86. Specific limitations include the inability to trap on some privileged instructions. If a processor is not designed to support virtualization primarily, it is necessary to modify the hardware to satisfy the three requirements for a VMM. This is known as hardware-assisted virtualization.

Comparison of Four VMM and Hypervisor Software Packages

Provider and References	Host CPU	Host OS	Guest OS	Architecture
VMware Workstation [71]	x86, x86-64	Windows, Linux	Windows, Linux, Solaris, FreeBSD, Netware, OS/2, SCO, BeOS, Darwin	Full Virtualization
VMware ESX Server [71]	x86, x86-64	No host OS	The same as VMware Workstation	Para-Virtualization
Xen [7,13,42]	x86, x86-64, IA-64	NetBSD, Linux, Solaris	FreeBSD, NetBSD, Linux, Solaris, Windows XP and 2003 Server	Hypervisor
KVM [31]	x86, x86-64, IA-64, S390, PowerPC	Linux	Linux, Windows, FreeBSD, Solaris	Para-Virtualization

26 ***Explain virtualization at operating system level. What are its advantages and disadvantages?***

Ans: **Virtualization Support at the OS Level**

As we know, it is slow to initialize a hardware-level VM because each VM creates its own image from scratch. In a cloud computing environment, perhaps thousands of VMs need to be initialized simultaneously. Besides slow operation, storing the VM images also becomes an issue. As a matter of fact, there is considerable repeated content among VM images. Moreover, full virtualization at the hardware level also has the disadvantages of slow performance and low density, and the need for para-virtualization to modify the guest OS. To reduce the performance overhead of

hardware-level virtualization, even hardware modification is needed. OS-level virtualization provides a feasible solution for these hardware-level virtualization issues.

Operating system virtualization inserts a virtualization layer inside an operating system to partition a machine's physical resources. It enables multiple isolated VMs within a single operating system kernel. This kind of VM is often called a virtual execution environment (VE), Virtual Private System (VPS), or simply container. From the user's point of view, VEs look like real servers. This means a VE has its own set of processes, file system, user accounts, network interfaces with IP addresses, routing tables, firewall rules, and other personal settings. Although VEs can be customized for different people, they share the same operating system kernel. Therefore, OS-level

virtualization is also called single-OS image virtualization.

Figure below illustrates operating system virtualization from the point of view of a machine stack.

- **Advantages of OS Extensions**

Compared to hardware-level virtualization, the benefits of OS extensions are twofold: (1) VMs at the operating system level have minimal startup/shutdown costs, low resource requirements, and high scalability; and

(2) for an OS-level VM, it is possible for a VM and its host environment to synchronize

state changes when necessary.

These benefits can be achieved via two mechanisms of OS-level virtualization:

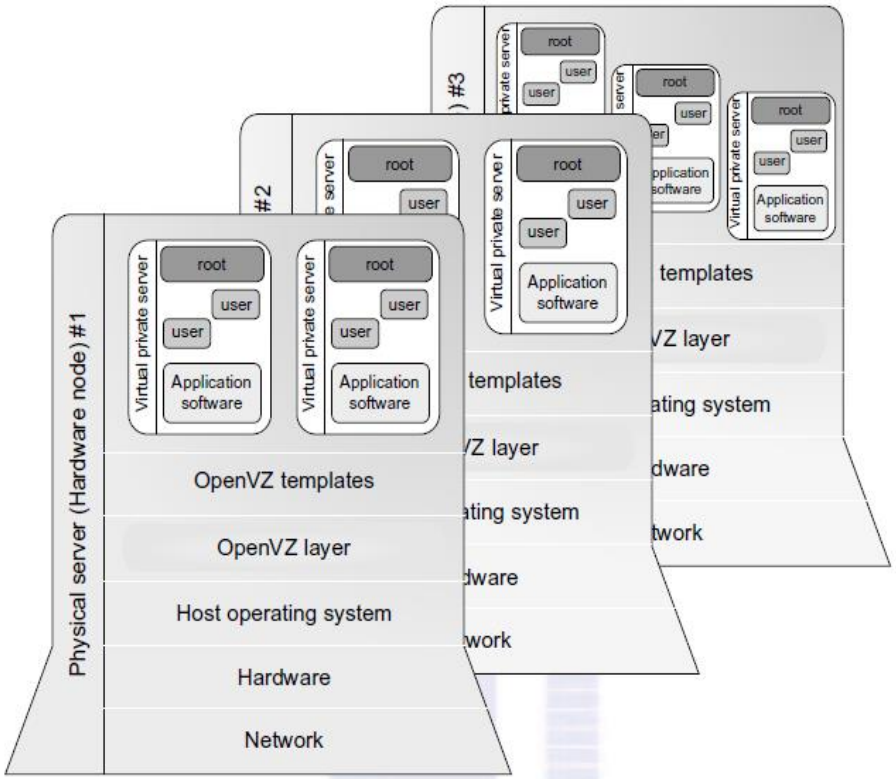
(1) All OS-level VMs on the same physical machine share a single operating system kernel; and

(2) the virtualization layer can be designed in a way that allows processes in VMs to

access as many resources of the host machine as possible, but never to modify them. In cloud computing, the first and second benefits can be used to overcome the defects of slow initialization of VMs at the hardware level, and being unaware of the current application state, respectively

- **Disadvantages of OS Extensions**

The main disadvantage of OS extensions is that all the VMs at operating system level on a single container must have the same kind of guest operating system. That is, although different OS-level VMs may have different operating system distributions, they must pertain to the same operating system family. For example, a Windows distribution such as Windows XP cannot run on a Linux-based container. However, users of cloud computing have various preferences. Some prefer Windows and others prefer Linux or other operating systems. Therefore, there is a challenge for



The OpenVZ virtualization layer inside the host OS, which provides some OS images to create VMs quickly.
(*Courtesy of OpenVZ User's Guide [65]*)

.

OS-level virtualization in such cases.

Figure above illustrates the concept of OS-level virtualization. The virtualization layer is inserted inside the OS to partition the hardware resources for multiple VMs to run their applications in multiple virtual environments. To implement OS-level virtualization, isolated execution environments (VMs) should be created based on a single OS kernel. Furthermore, the access requests from a VM need to be redirected to the VM's local resource partition on the physical machine.

For example, the chroot command in a UNIX system can create several virtual root directories within a host OS. These virtual root directories are the root directories of all VMs created. There are two ways to implement virtual root directories: duplicating common resources to each VM partition; or sharing most resources with the host environment and only creating private resource copies on the VM on demand. The first way incurs significant resource costs and overhead on a physical machine. This issue neutralizes the benefits of OS-level virtualization, compared with

hardware-assisted virtualization. Therefore, OS-level virtualization is often a second choice.

27 ***Explain the middleware support for virtualization.***

Ans: **Middleware Support for Virtualization**

Library-level virtualization is also known as user-level Application Binary Interface (ABI) or API emulation. This type of virtualization can create execution environments for running alien programs on a platform rather than creating a VM to run the entire operating system. API call interception and remapping are the key functions performed. This section provides an overview of several library-level virtualization systems: namely the Windows Application Binary Interface (WABI), lxrun, WINE, Visual MainWin, and vCUDA, which are summarized in following table

Middleware and Library Support for Virtualization

Middleware or Runtime Library and References or Web Link	Brief Introduction and Application Platforms
WABI (http://docs.sun.com/app/docs/doc/802-6306)	Middleware that converts Windows system calls running on x86 PCs to Solaris system calls running on SPARC workstations
Lxrun (Linux Run) (http://www.ugcs.caltech.edu/~steven/lxrun/)	A system call emulator that enables Linux applications written for x86 hosts to run on UNIX systems such as the SCO OpenServer
WINE (http://www.winehq.org/)	A library support system for virtualizing x86 processors to run Windows applications under Linux, FreeBSD, and Solaris
Visual MainWin (http://www.mainsoft.com/)	A compiler support system to develop Windows applications using Visual Studio to run on Solaris, Linux, and AIX hosts
vCUDA (Example 3.2) (IEEE *IPDPS* 2009 [57])	Virtualization support for using general-purpose GPUs to run data-intensive applications under a special guest OS

The WABI offers middleware to convert Windows system calls to Solaris system calls. Lxrun is really a system call emulator that enables Linux applications written for x86

hosts to run on UNIX systems. Similarly, Wine offers library support for virtualizing x86 processors to run Windows applications on UNIX hosts. Visual MainWin offers a compiler support system to develop Windows applications using Visual Studio to run on some UNIX hosts. The vCUDA is explained in following Example with a graphical illustration.

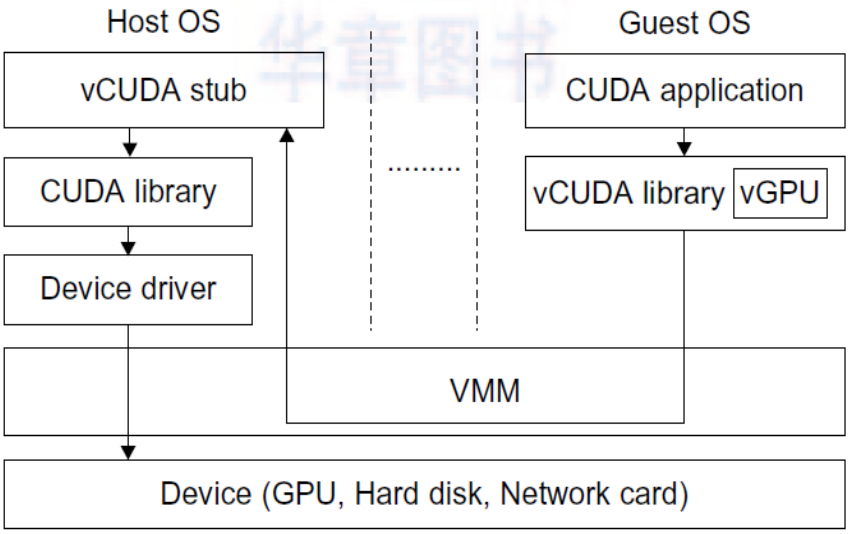
The vCUDA for Virtualization of General-Purpose GPUs

CUDA is a programming model and library for general-purpose GPUs. It leverages the high performance of vGPUs to run compute-intensive applications on host operating systems. However, it is difficult to run CUDA applications on hardware-level VMs directly. vCUDA virtualizes the CUDA library and can be installed on guest OSes. When CUDA applications run on a guest OS and issue a call to the CUDA API, vCUDA intercepts the call and redirects it to the CUDA API running on the host OS.below Figure shows the basic concept of the vCUDA architecture .

The vCUDA employs a client-server model to implement CUDA virtualization. It consists of three user space components: the vCUDA library, a virtual GPU in the guest OS (which acts as a client), and the vCUDA stub in the host OS (which acts as a server). The vCUDA library resides in the guest OS as a substitute for the standard CUDA library. It is responsible for intercepting and redirecting API calls from

the client to the stub. Besides these tasks, vCUDA also creates vGPUs and manages them.

The functionality of a vGPU is threefold: It abstracts the GPU structure and gives applications a uniform view of the underlying hardware; when a CUDA application in the guest OS allocates a device's memory the vGPU can return a local virtual address to the application and notify the remote stub to allocate the real device memory, and the vGPU is responsible for storing the CUDA API flow. The vCUDA stub receives



Basic concept of the vCUDA architecture.

and interprets remote requests and creates a corresponding execution context for the API calls from the guest OS, then returns the results to the guest OS. The vCUDA stub also manages actual physical resource allocation.

28 ***Explain public, private and hybrid clouds.***

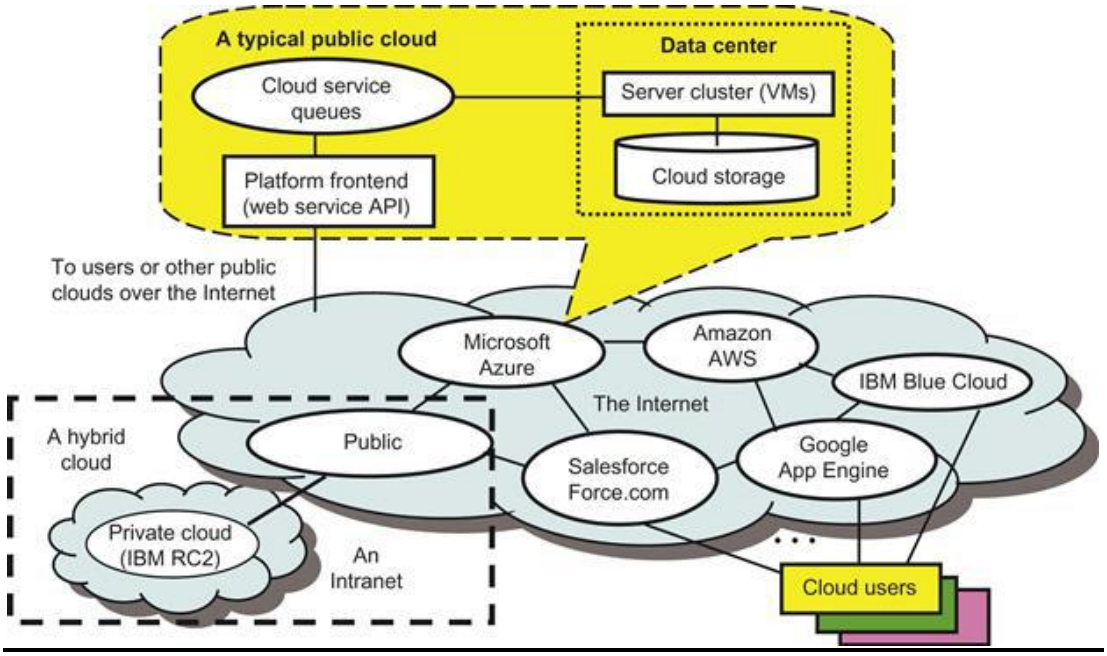
Ans: Public, Private, and Hybrid Clouds

Cloud computing is a high-throughput computing (HTC) paradigm whereby the infrastructure provides the services through a large data center or server farms. The cloud computing model enables users to share access to resources from anywhere at any time through their connected devices.

The cloud offers significant benefit to IT companies by freeing them from the low-level task of setting up the hardware (servers) and managing the system software. Cloud computing applies a virtual platform with elastic resources put together by on-demand provisioning of hardware, software, and data sets, dynamically.

Following shows, both *public clouds* and *private clouds* are developed in the Internet. As many clouds are generated by commmercial providers or by enterprises in a distributed manner, they will be interconnected over the Internet to achieve scalable and efficient computing services. Commercial cloud providers such as Amazon, Google, and Microsoft created their platforms to be distributed geographically. This distribution is partially attributed to fault tolerance, response latency reduction,

and even legal reasons. Intranet-based private clouds are linked to public clouds to get additional resources. Nevertheless, users in Europe may not feel comfortable using clouds in the United States, and vice versa, until extensive *service-level agreements* (SLAs) are developed between the two user communities.



Public, private, and hybrid clouds illustrated by functional architecture and connectivity of representative clouds available by 2011.

- ***Public Clouds***

A *public cloud* is built over the Internet and can be accessed by any user who has paid for the service. Public clouds are owned by service providers and are accessible through a subscription. The callout box in top of above Figure shows the architecture

of a typical public cloud. Many public clouds are available, including Google App Engine (GAE), Amazon Web Services (AWS), Microsoft Azure, IBM Blue Cloud, and Salesforce.com's Force.com. The providers of the aforementioned clouds are

commercial providers that offer a publicly accessible remote interface for creating and managing VM instances within their proprietary infrastructure. A public cloud delivers a selected set of business processes. The application and infrastructure services are offered on a flexible price-per-use basis.

- ***Private Clouds***

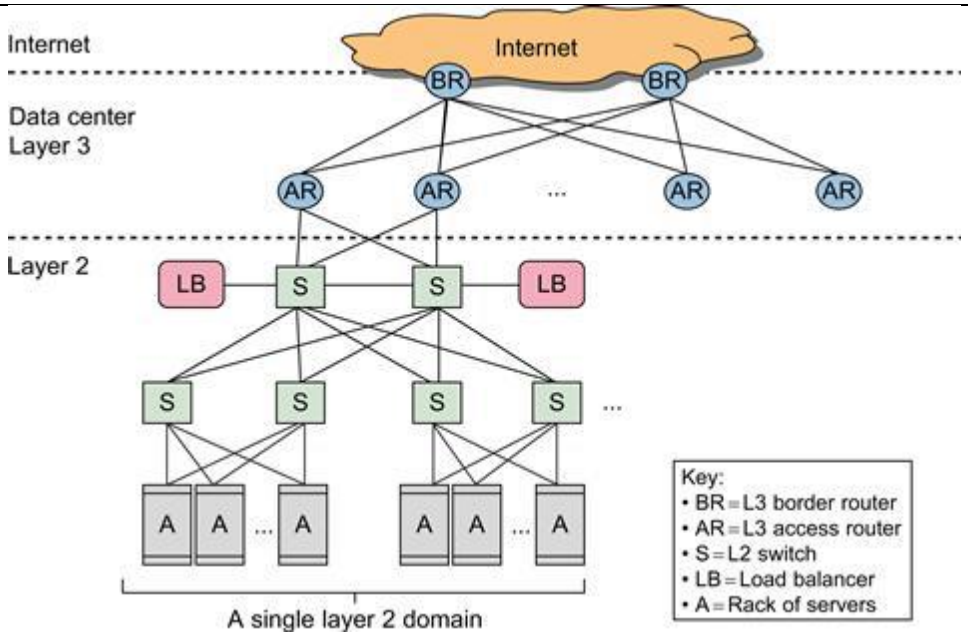
A *private cloud* is built within the domain of an intranet owned by a single organization. Therefore, it is client owned and managed, and its access is limited to the owning clients and their partners. Its deployment was not meant to sell capacity

over the Internet through publicly accessible interfaces. Private clouds give local users a flexible and agile private infrastructure to run service workloads within their administrative domains. A private cloud is supposed to deliver more efficient and

convenient cloud services. It may impact the cloud standardization, while retaining greater customization and organizational control.

- ***Hybrid Clouds***

	A *hybrid cloud* is built with both public and private clouds, as shown at the lower-left corner of above Figure. Private clouds can also support a hybrid cloud model by supplementing local infrastructure with computing capacity from an external public cloud. For example, the *Research Compute Cloud* (RC2) is a private cloud, built by IBM, that interconnects the computing and IT resources at eight IBM Research Centers scattered throughout the United States, Europe, and Asia. A hybrid cloud provides access to clients, the partner network, and third parties. In summary, public clouds promote standardization, preserve capital investment, and offer application flexibility. Private clouds attempt to achieve customization and offer higher efficiency, resiliency, security, and privacy. Hybrid clouds operate in the middle, with many compromises in terms of resource sharing.
29	***With the help of diagram explain the structure of data-centre networking.*** Ans: ***Data-Center Networking Structure*** The core of a cloud is the server cluster (or VM cluster). Cluster nodes are used as compute nodes. A few control nodes are used to manage and monitor cloud activities. The scheduling of user jobs requires that you assign work to virtual clusters created for users. The gateway nodes provide the access points of the service from the outside world. These gateway nodes can be also used for security control of the entire cloud platform. In physical clusters and traditional grids, users expect static demand of resources. Clouds are designed to handle fluctuating workloads, and thus demand variable resources dynamically. Private clouds will satisfy this demand if properly designed and managed. Data centers and supercomputers have some similarities as well as fundamental differences. In the case of data centers, scaling is a fundamental requirement. Data-center server clusters are typically built with large number of servers, ranging from thousands to millions of servers (nodes). For example, Microsoft has a data center in the Chicago area that has 100,000 eight-core servers, housed in 50 containers. In supercomputers, a separate data farm is used, while a data center uses disks on server nodes plus memory cache and databases. Data centers and supercomputers also differ in networking requirements, as illustrated in Figure below. Supercomputers use custom-designed high-bandwidth networks such as fat trees or 3D torus networks . Datacenter networks are mostly IP-based commodity networks, such as the 10 Gbps Ethernet network, which is optimized for Internet access. Figure below shows a multilayer structure for accessing the Internet. The server racks are at the bottom Layer 2, and they are connected through fast switches (S) as the hardware core. The data center is connected to the Internet at Layer 3 with many *access routers* (ARs) and *border routers* (BRs).



Standard data-center networking for the cloud to access the Internet

An example of a private cloud is the one the U.S. National Aeronautics and Space Administration (NASA) is building to enable researchers to run climate models on remote systems it provides. This can save users the capital expense of HPC

machines at local sites. Furthermore, NASA can build the complex weather models around its data centers, which is more cost-effective. Another good example is the cloud built by the European Council for Nuclear Research (CERN). This is a very

big private cloud designed to distribute data, applications, and computing resources to thousands of scientists around the world.

These cloud models demand different levels of performance, data protection, and security enforcement. In this case, different

SLAs may be applied to satisfy both providers and paid users. Cloud computing exploits many existing technologies. For example, grid computing is the backbone of cloud computing in that the grid has the same goals of resource sharing with

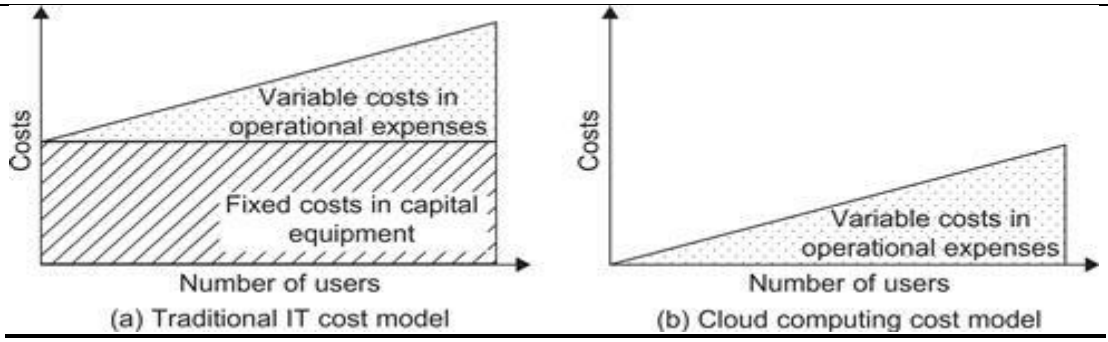
better utilization of research facilities. Grids are more focused on delivering storage and computing resources while cloud computing aims to achieve economies of scale with abstracted services and resources.

30 ***Enumerate the design objectives for cloud computing.***

Ans: ***Cloud Design Objectives***

Despite the controversy surrounding the replacement of desktop or deskside computing by centralized computing and storage services at data centers or big IT

	companies, the cloud computing community has reached some consensus on what has to be done to make cloud computing universally acceptable. The following list highlights six design objectives for cloud computing: • **Shifting computing from desktops to data centers** Computer processing, storage, and software delivery is shifted away from desktops and local servers and toward data centers over the Internet. • **Service provisioning and cloud economics** Providers supply cloud services by signing SLAs with consumers and end users. The services must be efficient in terms of computing, storage, and power consumption. Pricing is based on a payas- you-go policy. • **Scalability in performance** The cloud platforms and software and infrastructure services must be able to scale in performance as the number of users increases. • **Data privacy protection** Can you trust data centers to handle your private data and records? This concern must be addressed to make clouds successful as trusted services. • **High quality of cloud services** The QoS of cloud computing must be standardized to make clouds interoperable among multiple providers. • **New standards and interfaces** This refers to solving the data lock-in problem associated with data centers or cloud providers. Universally accepted APIs and access protocols are needed to provide high portability and flexibility of virtualized applications.
31	***Compare and explain the traditional IT and cloud computing cost models.*** **Ans:** ***Cost Model*** In traditional IT computing, users must acquire their own computer and peripheral equipment as capital expenses. In addition, they have to face operational expenditures in operating and maintaining the computer systems, including personnel and service costs. Figure (a) shows the addition of variable operational costs on top of fixed capital investments in traditional IT. Note that the fixed cost is the main cost, and that it could be reduced slightly as the number of users increases. However, the operational costs may increase sharply with a larger number of users. Therefore, the total cost escalates quickly with massive numbers of users. On the other hand, cloud computing applies a pay-per-use business model, in which user jobs are outsourced to data centers. To use the cloud, one has no up-front cost in hardware acquisitions. Only variable costs are experienced by cloud users, as demonstrated in Figure(b).



Computing economics between traditional IT users and cloud users.

Overall, cloud computing will reduce computing costs significantly for both small users and large enterprises. Computing economics does show a big gap between traditional IT users and cloud users. The savings in acquiring expensive computers

up front releases a lot of burden for startup companies. The fact that cloud users only pay for operational expenses and do not have to invest in permanent equipment is especially attractive to massive numbers of small users. This is a major driving

force for cloud computing to become appealing to most enterprises and heavy computer users. In fact, any IT users whose capital expenses are under more pressure than their operational expenses should consider sending their overflow work to

utility computing or cloud service providers.

32 ***What are cloud ecosystems? Explain the cloud ecosystem for building private clouds.***

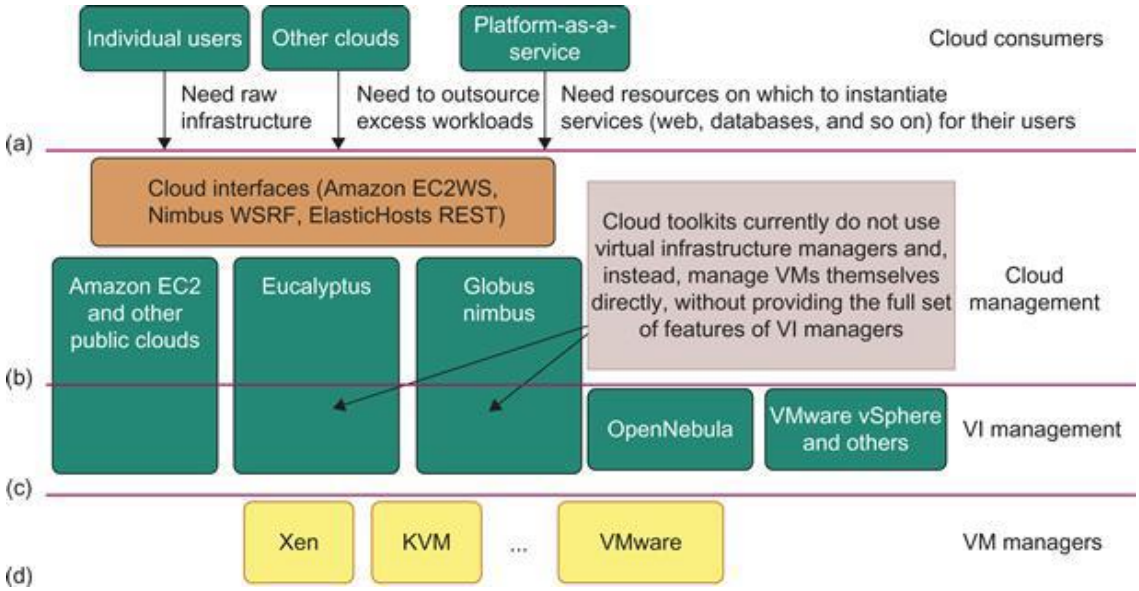
Ans: ***Cloud Ecosystems***

With the emergence of various Internet clouds, an ecosystem of providers, users, and technologies has appeared. This ecosystem has evolved around public clouds. Strong interest is growing in open source cloud computing tools that let organizations

build their own IaaS clouds using their internal infrastructures. Private and hybrid clouds are not exclusive, since public clouds are involved in both cloud types. A private/hybrid cloud allows remote access to its resources over the Internet

using remote web service interfaces such as that used in Amazon EC2.

An ecosystem was suggested by Sotomayor, et al (Figure below) for building private clouds. They suggested four levels of ecosystem development in a private cloud. At the user end, consumers demand a flexible platform. At the cloud management level, the cloud manager provides virtualized resources over an IaaS platform. At the virtual infrastructure (VI) management level, the manager allocates VMs over multiple server clusters. Finally, at the VM management level, the VM managers handle VMs installed on individual host machines. An ecosystem of cloud tools attempts to span both cloud management and VI management. Integrating these two layers is complicated by the lack of open and standard interfaces between them.



Cloud ecosystem for building private clouds:

(a) Consumers demand a flexible platform;

(b) Cloud manager provides virtualized resources over an IaaS platform;

(c) VI manager allocates VMs;

(d) VM managers handle VMs installed on servers

An increasing number of startup companies are now basing their IT strategies on cloud resources, spending little or no capital to manage their own IT infrastructures. We desire a flexible and open architecture that enables organizations to build

private/hybrid clouds. VI management is aimed at this goal. Example VI tools include oVirt (https://fedorahosted.org/ovirt/), vSphere/4 (www.vmware.com/products/vsphere/) from VMWare, and VM Orchestrator (www.platform.com/Products/platform-vm-orchestrator) from Platfom Computing

.

These tools support dynamic placement and VM management on a pool of physical resources, automatic load balancing, server consolidation, and dynamic infrastructure resizing and partitioning. In addition to public clouds such as Amazon

EC2, Eucalyptus and Globus Nimbus are open source tools for virtualization of cloud infrastructure. To access these cloud management tools, one can use the Amazon EC2WS, Nimbus WSRF, and ElasticHost REST cloud interfaces. For VI management,

	OpenNebula and VMware vSphere can be used to manage all VM generation including Xen, KVM, and VMware tools.
33	***Explain Infrastructure as a service with example.*** **Ans: Infrastructure-as-a-Service (IaaS)** Cloud computing delivers infrastructure, platform, and software (application) as services, which are made available as subscription-based services in a pay-as-you-go model to consumers. The services provided over the cloud can be generally categorized into three different service models: namely IaaS, Platform as a Service (PaaS), and Software as a Service (SaaS). ***Infrastructure as a Service*** This model allows users to use virtualized IT resources for computing, storage, and networking. In short, the service is performed by rented cloud infrastructure. The user can deploy and run his applications over his chosen OS environment. The user does not manage or control the underlying cloud infrastructure, but has control over the OS, storage, deployed applications, and possibly select networking components. This IaaS model encompasses *storage as a service*, compute *instances as a service*, and *communication as a service*. The *Virtual Private Cloud (VPC)* in Example below shows how to provide Amazon EC2 clusters and S3 storage to multiple users. Many startup cloud providers have appeared in recent years. GoGrid, FlexiScale, and Aneka are good examples. following Table summarizes the IaaS offerings by five public cloud providers. **Public Cloud Offerings of IaaS Example**

Cloud Name	VM Instance Capacity	API and Access Tools	Hypervisor, Guest OS
Amazon EC2	Each instance has 1–20 EC2 processors, 1.7–15 GB of memory, and 160–1.69 TB of storage.	CLI or web Service (WS) portal	Xen, Linux, Windows
GoGrid	Each instance has 1–6 CPUs, 0.5–8 GB of memory, and 30–480 GB of storage.	REST, Java, PHP, Python, Ruby	Xen, Linux, Windows
Rackspace Cloud	Each instance has a four-core CPU, 0.25–16 GB of memory, and 10–620 GB of storage.	REST, Python, PHP, Java, C#, .NET	Xen, Linux
FlexiScale in the UK	Each instance has 1–4 CPUs, 0.5–16 GB of memory, and 20–270 GB of storage.	web console	Xen, Linux, Windows
Joyent Cloud	Each instance has up to eight CPUs, 0.25–32 GB of memory, and 30–480 GB of storage.	No specific API, SSH, Virtual/Min	OS-level virtualization, OpenSolaris

Example Iaas

Amazon VPC for Multiple Tenants

A user can use a private facility for basic computations. When he must meet a specific workload requirement, he can use the Amazon VPC to provide additional EC2 instances or more storage (S3) to handle urgent applications. Following Figure shows VPC which is essentially a private cloud designed to address the privacy

concerns of public clouds that hamper their application when sensitive data and software are involved.

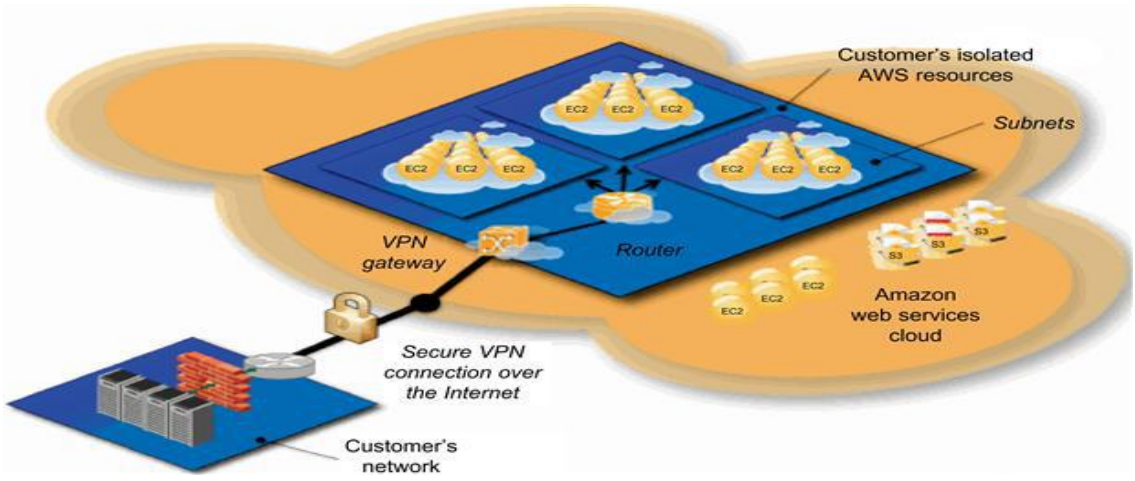


Figure Amazon VPC (virtual private cloud)

Amazon EC2 provides the following services: resources from multiple data centers globally distributed, CL1, web services (SOAP and Query), web-based console user interfaces, access to VM instances via SSH and Windows, 99.5 percent available agreements, per-hour pricing, Linux and Windows OSes, and automatic

scaling and load balancing.

VPC allows the user to isolate provisioned AWS processors, memory, and storage from interference by other users. Both *auto-scaling* and *elastic load balancing* services can support related demands. Auto-scaling enables users to automatically scale their VM instance capacity up or down. With auto-scaling, one can ensure that a sufficient

number of Amazon EC2 instances are provisioned to meet desired performance. Or one can scale down the VM instance capacity to reduce costs, when the workload is reduced.

34 ***Explain Platform as a service with example***.

Ans: ***Platform as a Service (PaaS)***

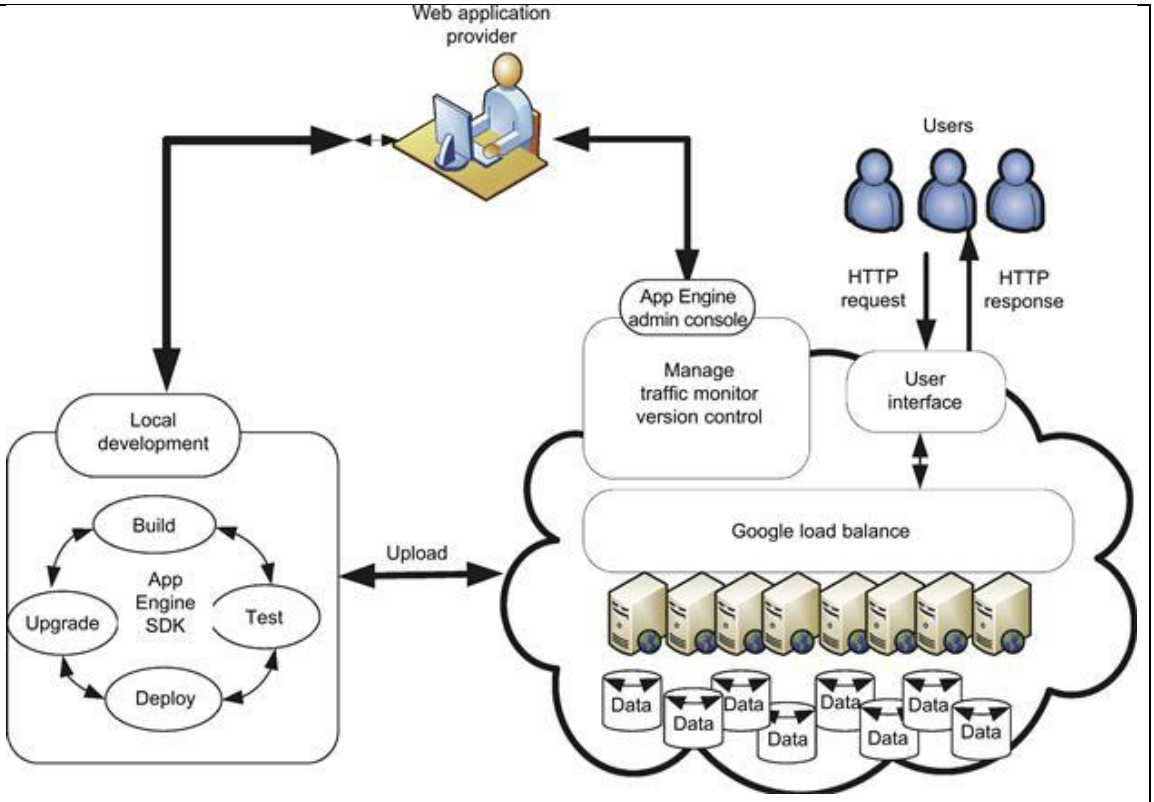
To be able to develop, deploy, and manage the execution of applications using provisioned resources demands a cloud platformwith the proper software environment. Such a platform includes operating system and runtime library support. This

has triggered the creation of the PaaS model to enable users to develop and deploy their user applications. Following Table highlights cloud platform services offered by five PaaS services.

Five Public Cloud Offerings of PaaS

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

	The platform cloud is an integrated computer system consisting of both hardware and software infrastructure. The user application can be developed on this virtualized cloud platform using some programming languages and software tools supported by the provider (e.g., Java, Python, .NET). The user does not manage the underlying cloud infrastructure. The cloud provider supports user application development and testing on a well-defined service platform. This PaaS model enables a collaborated software development platform for users from different parts of the world. This model also encourages third parties to provide software management, integration, and service monitoring solutions. **Example** **Google AppEngine for PaaS Applications** As web applications are running on Google's server clusters, they share the same capability with many other users. The applications have features such as automatic scaling and load balancing which are very convenient while building web applications. The distributed scheduler mechanism can also schedule tasks for triggering events at specified times and regular intervals. Following Figure shows the operational model for GAE. To develop applications using GAE, a development environment must be provided.



Google App Engine platform for PaaS operations

Google provides a fully featured local development environment that simulates GAE on the developer's computer. All the functions and application logic can be implemented locally which is quite similar to traditional software development. The coding and debugging stages can be performed locally as well. After these steps are finished, the SDK provided provides a tool for uploading the user's application to Google's infrastructure where the applications are actually deployed. Many additional third-party capabilities, including software management, integration, and service monitoring solutions, are also provided.

Here are some useful links when logging on to the GAE system:

• Google App Engine home page: http://code.google.com/appengine/

• Sign up for an account or use your Gmail account name: https://appengine.google.com/

• Download GAE SDK: http://code.google.com/appengine/downloads.html

• Python Getting Started Guide: http://code.google.com/appengine/docs/python/gettingstarted/

• Java Getting Started Guide: http://code.google.com/appengine/docs/java/gettingstarted/

	• Quota page for free service: http://code.google.com/appengine/docs/quotas.html#Resources • Billing page if you go over the quota: http://code.google.com/appengine/docs/ billing.html#Billable_Quota_Unit_Cost
35	***Explain Software as a service with example.*** **Ans:** ***Software as a Service (SaaS)*** This refers to browser-initiated application software over thousands of cloud customers. Services and tools offered by PaaS are utilized in construction of applications and management of their deployment on resources offered by IaaS providers. The SaaS model provides software applications as a service. As a result, on the customer side, there is no upfront investment in servers or software licensing. On the provider side, costs are kept rather low, compared with conventional hosting of user applications. Customer data is stored in the cloud that is either vendor proprietary or publicly hosted to support PaaS and IaaS. The best examples of SaaS services include Google Gmail and docs, Microsoft SharePoint, and the CRM software from Salesforce.com. They are all very successful in promoting their own business or are used by thousands of small businesses in their day-to-day operations. Providers such as Google and Microsoft offer integrated IaaS and PaaS services, whereas others such as Amazon and GoGrid offer pure IaaS services and expect third-party PaaS providers such as Manjrasoft to offer application development and deployment services on top of their infrastructure services. To identify important cloud applications in enterprises, the success stories of three real-life cloud applications are presented in following Example for HTC, news media, and business transactions. The benefits of using cloud services are evident in these SaaS applications. **Example** **Three Success Stories on SaaS Applications** 1. To discover new drugs through DNA sequence analysis, Eli Lily Company has used Amazon's AWS platform with provisioned server and storage clusters to conduct high-performance biological sequence analysis without using an expensive supercomputer. The benefit of this IaaS application is reduced drug deployment time with much lower costs. 2. The *New York Times* has applied Amazon's EC2 and S3 services to retrieve useful pictorial information quickly from millions of archival articles and newspapers. The *New York Times* has significantly reduced the time and cost in getting the job done.

	3. Pitney Bowes, an e-commerce company, offers clients the opportunity to perform B2B transactions using the Microsoft Azure platform, along with .NET and SQL services. These offerings have significantly increased the company's client base.
36	***Enumerate the enabling technologies in Hardware, Software and Networking for clouds.*** **Ans:** ***Enabling Technologies for Clouds*** The key driving forces behind cloud computing are the ubiquity of broadband and wireless networking, falling storage costs, and progressive improvements in Internet computing software. Cloud users are able to demand more capacity at peak demand, reduce costs, experiment with new services, and remove unneeded capacity, whereas service providers can increase system utilization via multiplexing, virtualization, and dynamic resource provisioning. Clouds are enabled by the progress in hardware, software, and networking technologies summarized in following Table. **Cloud-Enabling Technologies in Hardware, Software, and Networking** **Technology** — **Requirements and Benefits** Fast platform deployment — Fast, efficient, and flexible deployment of cloud resources to provide dynamic computing environment to users Virtual clusters on demand — Virtualized cluster of VMs provisioned to satisfy user demand and virtual cluster reconfigured as workload changes Multitenant techniques — SaaS for distributing software to a large number of users for their simultaneous use and resource sharing if so desired Massive data processing — Internet search and web services which often require massive data processing, especially to support personalized services Web-scale communication — Support for e-commerce, distance education, telemedicine, social networking, digital government, and digital entertainment applications Distributed storage — Large-scale storage of personal records and public archive information which demands distributed storage over the clouds Licensing and billing services — License management and billing services which greatly benefit all types of cloud services in utility computing These technologies play instrumental roles in making cloud computing a reality. Most of these technologies are mature today to meet increasing demand. In the hardware area, the rapid progress in multicore CPUs, memory chips, and disk arrays has made it possible to build faster data centers with huge amounts of storage space. Resource virtualization enables rapid cloud deployment and disaster recovery. *Service-oriented architecture* (SOA) also plays a vital role. Progress in providing SaaS, Web 2.0 standards, and Internet performance have all contributed to the emergence of cloud services. Today's clouds are designed to serve a large number of tenants over massive volumes of data. The availability of large-scale, distributed storage systems is the foundation of today's data centers. Of course, cloud computing is greatly benefitted by the progress made in license management and automatic billing techniques in recent years.
37	***With the help of a diagram explain the generic cloud architecture.***

Ans: ***A Generic Cloud Architecture***

Figure below shows a security-aware cloud architecture. The Internet cloud is envisioned as a massive cluster of servers. These servers are provisioned on demand to perform collective web services or distributed applications using data-center

resources. The cloud platform is formed dynamically by provisioning or deprovisioning servers, software, and database resources. Servers in the cloud can be physical machines or VMs. User interfaces are applied to request services. The provisioning

tool carves out the cloud system to deliver the requested service.

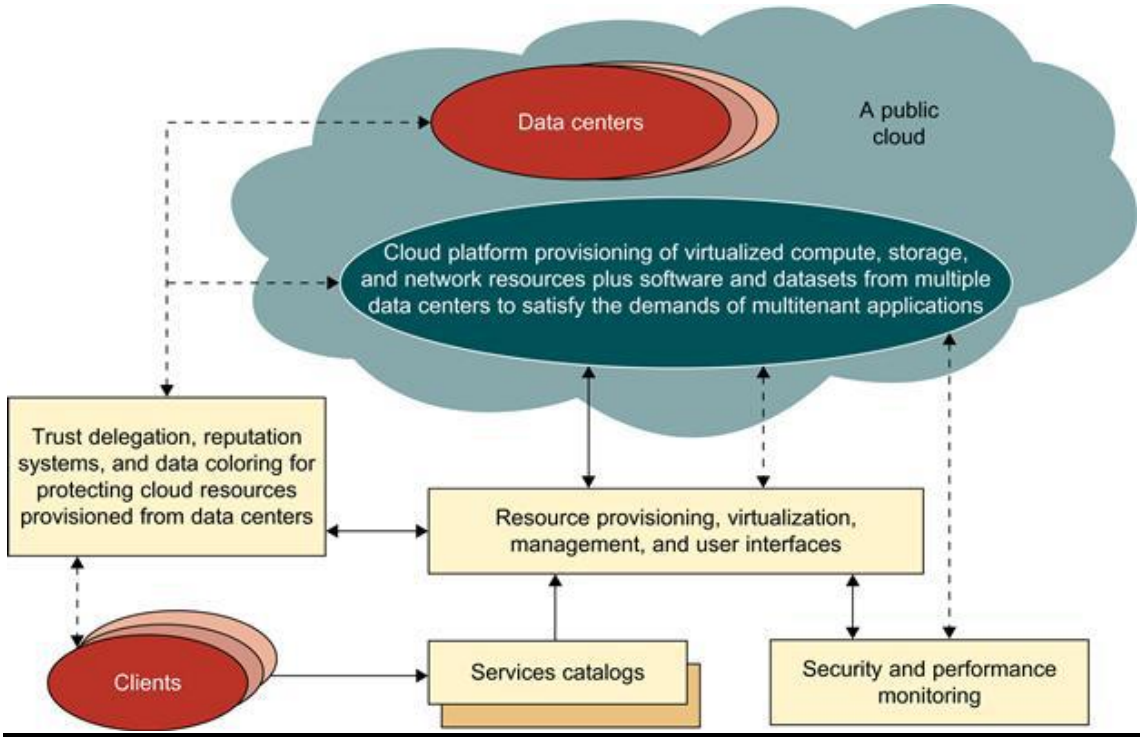


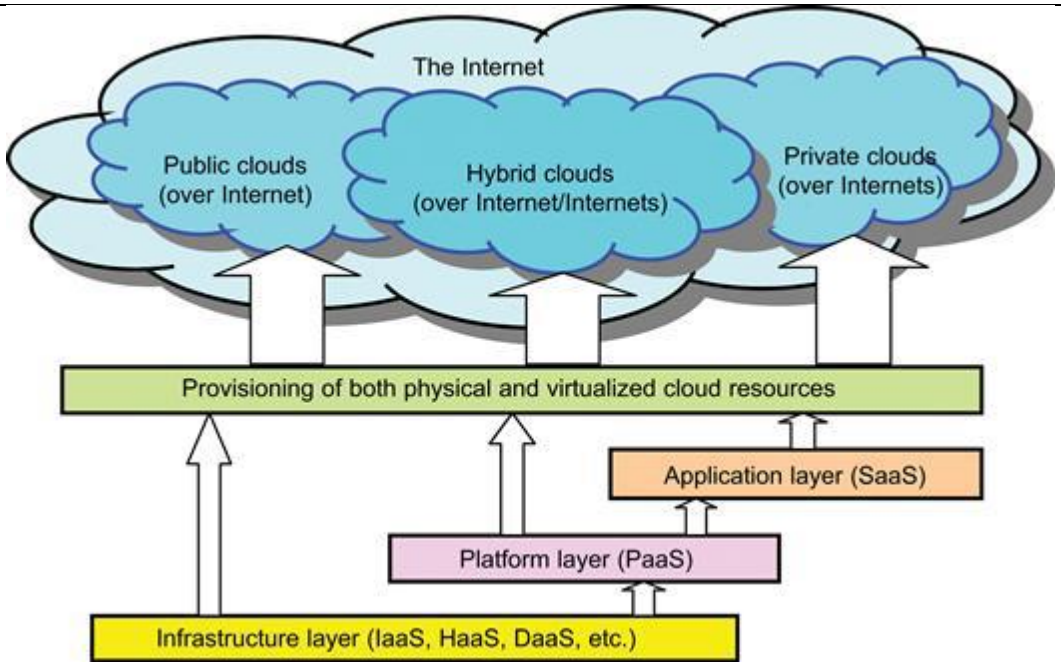
FIGURE A security-aware cloud platform built with a virtual cluster of VMs, storage, and networking resources over the data-center servers operated by providers

In addition to building the server cluster, the cloud platform demands distributed storage and accompanying services. The cloud computing resources are built into the data centers, which are typically owned and operated by a third-party provider.

Consumers do not need to know the underlying technologies. In a cloud, software becomes a service. The cloud demands a high degree of trust of massive amounts of data retrieved from large data centers. We need to build a framework to process

large-scale data stored in the storage system. This demands a distributed file system over the database system. Other cloud resources are added into a cloud platform, including storage area networks (SANs), database systems, firewalls, and security

	devices. Web service providers offer special APIs that enable developers to exploit Internet clouds. Monitoring and metering units are used to track the usage and performance of provisioned resources. The software infrastructure of a cloud platform must handle all resource management and do most of the maintenance automatically. Software must detect the status of each node server joining and leaving, and perform relevant tasks accordingly. Cloud computing providers, such as Google and Microsoft, have built a large number of data centers all over the world. Each data center may have thousands of servers. The location of the data center is chosen to reduce power and cooling costs. Thus, the data centers are often built around hydroelectric power. The cloud physical platform builder is more concerned about the performance/price ratio and reliability issues than shear speed performance. In general, private clouds are easier to manage, and public clouds are easier to access. The trends in cloud development are that more and more clouds will be hybrid. This is because many cloud applications must go beyond the boundary of an intranet. One must learn how to create a private cloud and how to interact with public clouds in the open Internet. Security becomes a critical issue in safeguarding the operation of all cloud types.
38	***With the help of a diagram explain the layered architectural development of cloud platform.*** **Ans:** **Layered Cloud Architectural Development** The architecture of a cloud is developed at three layers: infrastructure, platform, and application, as demonstrated in following Figure. These three development layers are implemented with virtualization and standardization of hardware and software resources provisioned in the cloud. The services to public, private, and hybrid clouds are conveyed to users through networking support over the Internet and intranets involved. It is clear that the infrastructure layer is deployed first to support IaaS services. This infrastructure layer serves as the foundation for building the platform layer of the cloud for supporting PaaS services. In turn, the platform layer is a foundation for implementing the application layer for SaaS applications. Different types of cloud services demand application of these resources separately.



Layered architectural development of the cloud platform for IaaS, PaaS, and SaaS applications over the Internet.

The infrastructure layer is built with virtualized compute, storage, and network resources. The abstraction of these hardware resources is meant to provide the flexibility demanded by users. Internally, virtualization realizes automated provisioning

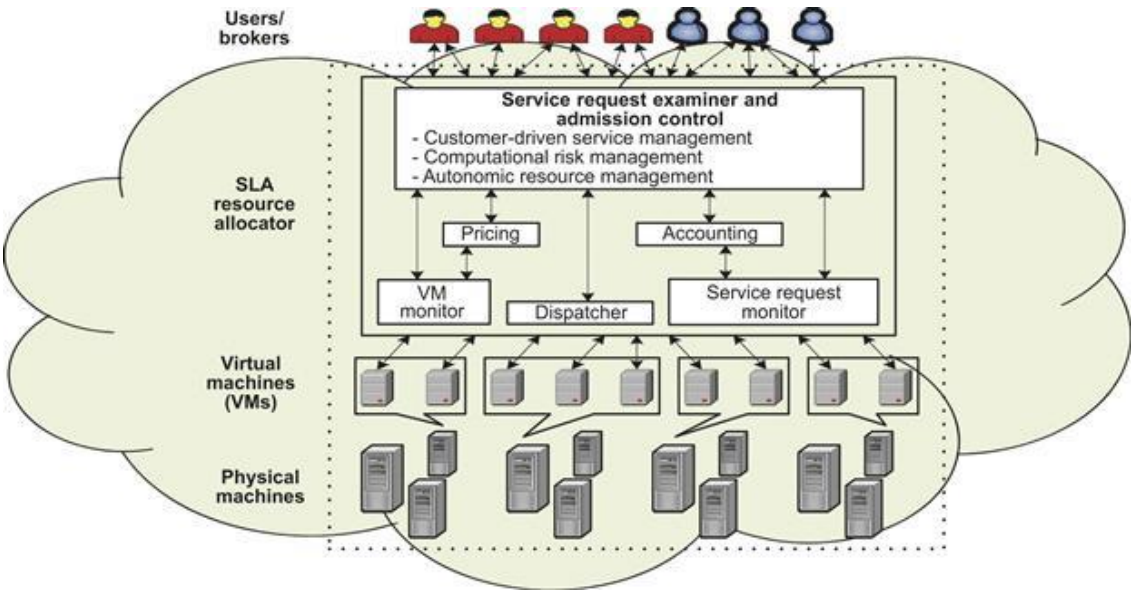
of resources and optimizes the infrastructure management process. The platform layer is for general-purpose and repeated usage of the collection of software resources. This layer provides users with an environment to develop their applications,

to test operation flows, and to monitor execution results and performance. The platform should be able to assure users that they have scalability, dependability, and security protection. In a way, the virtualized cloud platform serves as a “system middleware” between the infrastructure and application layers of the cloud.

The application layer is formed with a collection of all needed software modules for SaaS applications. Service applications in this layer include daily office management work, such as information retrieval, document processing, and calendar and authentication services. The application layer is also heavily used by enterprises in business marketing and sales, consumer relationship management (CRM), financial transactions, and supply chain management. It should be noted that not all cloud

services are restricted to a single layer. Many applications may apply resources at mixed layers. After all, the three layers are built from the bottom up with a dependence relationship.

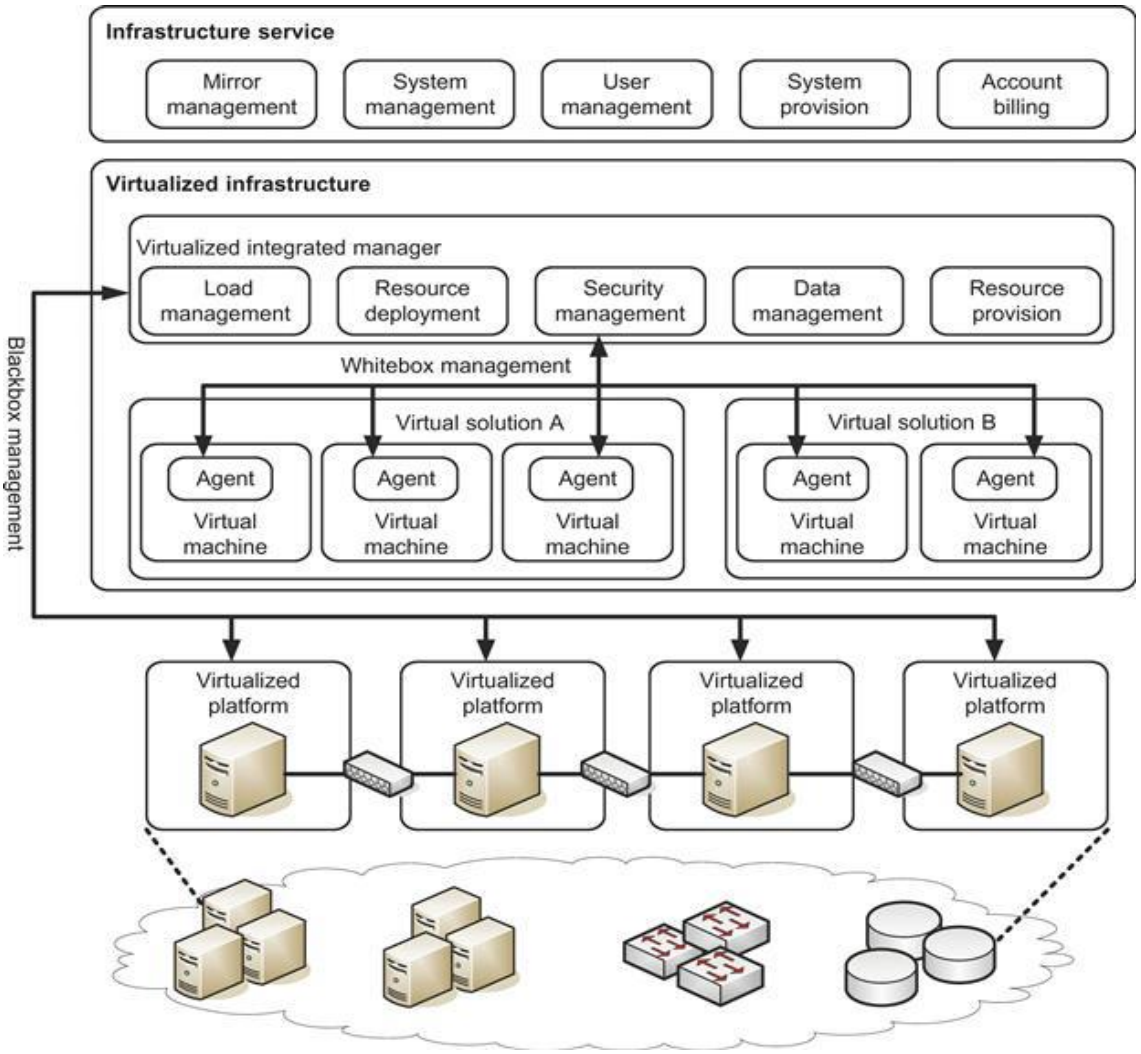
From the provider’s perspective, the services at various layers demand different amounts of functionality support and resource management by providers. In general, SaaS demands the most work from the provider, PaaS is in the middle, and

	IaaS demands the least. For example, Amazon EC2 provides not only virtualized CPU resources to users, but also management of these provisioned resources. Services at the application layer demand more work from providers. The best example of this is the Salesforce.com CRM service, in which the provider supplies not only the hardware at the bottom layer and the software at the top layer, but also the platform and software tools for user application development and monitoring.
39	***With the help of a diagram explain market oriented cloud architecture.*** **Ans**: ***Market-Oriented Cloud Architecture*** As consumers rely on cloud providers to meet more of their computing needs, they will require a specific level of QoS to be maintained by their providers, in order to meet their objectives and sustain their operations. Cloud providers consider and meet the different QoS parameters of each individual consumer as negotiated in specific SLAs. To achieve this, the providers cannot deploy traditional system-centric resource management architecture. Instead, market-oriented resource management is necessary to regulate the supply and demand of cloud resources to achieve market equilibrium between supply and demand. The designer needs to provide feedback on economic incentives for both consumers and providers. The purpose is to promote QoS-based resource allocation mechanisms. In addition, clients can benefit from the potential cost reduction of providers, which could lead to a more competitive market, and thus lower prices. Figure below shows the high-level architecture for supporting market-oriented resource allocation in a cloud computing environment. This cloud is basically built with the following entities:  Market-oriented cloud architecture to expand/shrink leasing of resources with variation in QoS/demand

	Users or brokers acting on user's behalf submit service requests from anywhere in the world to the data center and cloud to be processed. The SLA resource allocator acts as the interface between the data center/cloud service provider and external users/brokers. It requires the interaction of the following mechanisms to support SLA-oriented resource management. When a service request is first submitted the service request examiner interprets the submitted request for QoS requirements before determining whether to accept or reject the request. The request examiner ensures that there is no overloading of resources whereby many service requests cannot be fulfilled successfully due to limited resources. It also needs the latest status information regarding resource availability (from the VM Monitor mechanism) and workload processing (from the Service Request Monitor mechanism) in order to make resource allocation decisions effectively. Then it assigns requests to VMs and determines resource entitlements for allocated VMs. The Pricing mechanism decides how service requests are charged. For instance, requests can be charged based on submission time (peak/off-peak), pricing rates (fixed/changing), or availability of resources (supply/demand). Pricing serves as a basis for managing the supply and demand of computing resources within the data center and facilitates in prioritizing resource allocations effectively. The Accounting mechanism maintains the actual usage of resources by requests so that the final cost can be computed and charged to users. In addition, the maintained historical usage information can be utilized by the Service Request Examiner and Admission Control mechanism to improve resource allocation decisions. The VM Monitor mechanism keeps track of the availability of VMs and their resource entitlements. The Dispatcher mechanism starts the execution of accepted service requests on allocated VMs. The Service Request Monitor mechanism keeps track of the execution progress of service requests. Multiple VMs can be started and stopped on demand on a single physical machine to meet accepted service requests, hence providing maximum flexibility to configure various partitions of resources on the same physical machine to different specific requirements of service requests. In addition, multiple VMs can concurrently run applications based on different operating system environments on a single physical machine since the VMs are isolated from one another on the same physical machine.
40	***Discuss the Quality of Service factors for clouds.***

	Ans: ***Quality of Service Factors*** The data center comprises multiple computing servers that provide resources to meet service demands. In the case of a cloud as a commercial offering to enable crucial business operations of companies, there are critical QoS parameters to consider in a service request, such as time, cost, reliability, and trust/security. In particular, QoS requirements cannot be static and may change over time due to continuing changes in business operations and operating environments. In short, there should be greater importance on customers since they pay to access services in clouds. In addition, the state of the art in cloud computing has no or limited support for dynamic negotiation of SLAs between participants and mechanisms for automatic allocation of resources to multiple competing requests. Negotiation mechanisms are needed to respond to alternate offers protocol for establishing SLAs .. Commercial cloud offerings must be able to support customer-driven service management based on customer profiles and requested service requirements. Commercial clouds define computational risk management tactics to identify, assess, and manage risks involved in the execution of applications with regard to service requirements and customer needs. The cloud also derives appropriate market-based resource management strategies that encompass both customer-driven service management and computational risk management to sustain SLA-oriented resource allocation. The system incorporates autonomic resource management models that effectively self-manage changes in service requirements to satisfy both new service demands and existing service obligations, and leverage VM technology to dynamically assign resource shares according to service requirements.
41	***With the help of a diagram explain the infrastructure needed to virtualize the servers in a data centre for implementing specific cloud applications.*** **Ans:** One very distinguishing feature of cloud computing infrastructure is the use of system virtualization and the modification to provisioning tools. Virtualization of servers on a shared cluster can consolidate web services. As the VMs are the containers of cloud services, the provisioning tools will first find the corresponding physical machines and deploy the VMs to those nodes before scheduling the service to run on the virtual nodes. In addition, in cloud computing, virtualization also means the resources and fundamental infrastructure are virtualized. The user will not care about the computing resources that are used for providing the services. Cloud users do not need to know and have no way to discover physical resources that are involved while processing a service request. Also, application developers do not care about some infrastructure issues such as scalability and fault tolerance (i.e., they are virtualized).

Application developers focus on service logic. Figure below shows the infrastructure needed to virtualize the servers in a data center for implementing specific cloud applications.



Virtualized servers, storage, and network for cloud platform construction

42 ***What is hardware virtualization? Discuss the virtualization support in Windows Azure, Amazon Web Service and Google App Engine.***

Ans: ***Hardware Virtualization***

In many cloud computing systems, virtualization software is used to virtualize the hardware. System virtualization software is a special kind of software which simulates the execution of hardware and runs even unmodified operating systems. Cloud computing systems use virtualization software as the running environment for legacy software such as old operating systems and unusual applications. Virtualization software is also used as the platform for developing new cloud applications

that enable developers to use any operating systems and programming environments they like. The development environment and deployment environment can now be the same, which eliminates some runtime problems.

Some cloud computing providers have used virtualization technology to provide this service for developers. system virtualization software is considered the hardware analog mechanism to run an unmodified operating system, usually on bare hardware directly, on top of software. Table below lists some of the system virtualization software in wide use at the time of this writing. Currently, the VMs installed on a cloud computing platform are mainly used for hosting third-party programs. VMs provide flexible runtime services to free users from worrying about the system environment.

Using VMs in a cloud computing platform ensures extreme flexibility for users. As the computing resources are shared by many users, a method is required to maximize the users' privileges and still keep them separated safely. Traditional sharing

of cluster resources depends on the user and group mechanism on a system. Such sharing is not flexible. Users cannot customize the system for their special purposes. Operating systems cannot be changed. The separation is not complete. An

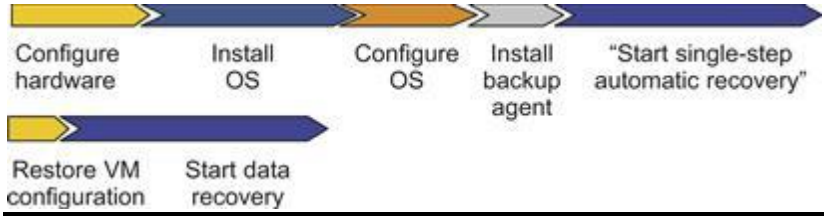
environment that meets one user's requirements often cannot satisfy another user. Virtualization allows users to have full privileges while keeping them separate.

Users have full access to their own VMs, which are completely separate from other users' VMs. Multiple VMs can be mounted on the same physical server. Different VMs may run with different OSes. We also need to establish the virtual disk

storage and virtual networks needed by the VMs. The virtualized resources form a resource pool. The virtualization is carried out by special servers dedicated to generating the virtualized resource pool. The virtualized infrastructure (black box in

the middle) is built with many *virtualizing integration managers*. These managers handle loads, resources, security, data, and provisioning functions. Figure below shows two VM platforms. Each platform carries out a virtual solution to a user job. All

cloud services are managed in the boxes at the top.



Recovery overhead of a conventional disaster recovery scheme, compared with that required to recover from live migration of VMs.

Table below three public clouds in the context of virtualization support: AWS, Microsoft

Azure, and GAE. AWS provides extreme flexibility (VMs) for users to execute their own applications. GAE provides limited application-level virtualization for users to build applications only based on the services that are created by Google. Microsoft

provides programming-level virtualization (.NET virtualization) for users to build their applications.

Provider	AWS	Microsoft Azure	GAE
Compute cloud with virtual cluster of servers	x86 instruction set, Xen VMs, resource elasticity allows scalability through virtual cluster, or a third party such as RightScale must provide the cluster	Common language runtime VMs provisioned by declarative descriptions	Predefined application framework handlers written in Python, automatic scaling up and down, server failover inconsistent with the web applications
Storage cloud with virtual storage	Models for block store (EBS) and augmented key/blob store (SimpleDB), automatic scaling varies from EBS to fully automatic (SimpleDB, S3)	SQL Data Services (restricted view of SQL Server), Azure storage service	MegaStore/BigTable
Network cloud services	Declarative IP-level topology; placement details hidden, security groups restricting communication, availability zones isolate network failure, elastic IP applied	Automatic with user's declarative descriptions or roles of app. components	Fixed topology to accommodate three-tier web app. structure, scaling up and down is automatic and programmer-invisible

43 ***Explain the following:***
i) Storage virtualization for green data-centres.
ii) Benefits of using virtual machines in clouds.
iii) Virtual machine cloning for disaster recovery.

Ans:

- ***Storage Virtualization for Green Data Centers***

IT power consumption in the United States has more than doubled to 3 percent of the total energy consumed in the country. The large number of data centers in the country has contributed to this energy crisis to a great extent. More than half of the

companies in the Fortune 500 are actively implementing new corporate energy policies. Recent surveys from both IDC and Gartner confirm the fact that virtualization had a great impact on cost reduction from reduced power consumption in physical

computing systems. This alarming situation has made the IT industry become more energy-aware. With little evolution of alternate energy resources, there is an imminent need to conserve power in all computers. Virtualization and server consolidation

have already proven handy in this aspect. Green data centers and benefits of storage virtualization are considered to further strengthen the synergy of green computing.

- ***VM Cloning for Disaster Recovery***

VM technology requires an advanced disaster recovery scheme. One scheme is to recover one physical machine by another

physical machine. The second scheme is to recover one VM by another VM traditional

disaster recovery from one physical machine to another is rather slow, complex, and expensive. Total recovery time is attributed to the hardware configuration, installing and configuring the OS, installing the backup agents, and the long time

to restart the physical machine. To recover a VM platform, the installation and configuration times for the OS and backup agents are eliminated. Therefore, we end up with a much shorter disaster recovery time, about 40 percent of that to recover

the physical machines. Virtualization aids in fast disaster recovery by VM encapsulation The cloning of VMs offers an effective solution. The idea is to make a

clone VM on a remote server for every running VM on a local server. Among all the clone VMs, only one needs to be active. The remote VM should be in a suspended mode. A cloud control center should be able to activate this clone VM in case of

failure of the original VM, taking a snapshot of the VM to enable live migration in a minimal amount of time. The migrated VM can run on a shared Internet connection. Only updated data and modified states are sent to the suspended VM to update

its state. The *Recovery Property Objective (RPO)* and *Recovery Time Objective (RTO*) are affected by the number of snapshots taken. Security of the VMs should be enforced during live migration of VMs.

- **Benefits of using virtual machines in clouds**.

VM technology has increased in ubiquity. This has enabled users to create customized environments atop physical infrastructure for cloud computing. Use of VMs in clouds has the following distinct benefits:

(1) System administrators consolidate workloads of underutilized servers in fewer servers;

(2) VMs have the ability to run legacy code without interfering with other APIs;

(3) VMs can be used to improve security through creation of sandboxes for running applications with questionable reliability;

And (4) virtualized cloud platforms can apply performance isolation, letting providers offer some guarantees and better QoS to customer applications.

44	***Explain the challenges in cloud architecture development.***

Ans: **Architectural Design Challenges**

we will identify six open challenges in cloud architecture development Plausible solutions to meet these challenges are discussed shortly.

- ***Challenge 1—Service Availability and Data Lock-in Problem***

The management of a cloud service by a single company is often the source of single points of failure. To achieve HA, one can consider using multiple cloud providers. Even if a company has multiple data centers located in different geographic regions,

it may have common software infrastructure and accounting systems. Therefore, using multiple cloud providers may provide more protection from failures. Another availability obstacle is distributed *denial of service* (DDoS) attacks. Criminals

threaten to cut off the incomes of SaaS providers by making their services unavailable. Some utility computing services offer SaaS providers the opportunity to defend against DDoS attacks by using quick scale-ups. Software stacks have improved interoperability among different cloud platforms, but the APIs itself are still proprietary. Thus, customers cannot easily extract their data and programs from one site to run on another.

The obvious solution is to standardize the APIs so that a SaaS developer can deploy services and data across multiple cloud providers. This will rescue the loss of all data due to the failure of a single company. In addition to mitigating data lock-in concerns, standardization of

APIs enables a new usage model in which the same software infrastructure can be used in both public and private clouds. Such an option could enable “surge computing,” in which the public cloud is used to capture the extra tasks that cannot be

easily run in the data center of a private cloud.

- ***Challenge 2—Data Privacy and Security Concerns***

Current cloud offerings are essentially public (rather than private) networks, exposing the system to more attacks. Many obstacles can be overcome immediately with well-understood technologies such as encrypted storage, virtual LANs, and network middleboxes (e.g., firewalls, packet filters). For example, you could encrypt your data before placing it in a cloud. Many nations have laws requiring SaaS providers to keep customer data and copyrighted material within national boundaries.

Traditional network attacks include buffer overflows, DoS attacks, spyware, malware, rootkits, Trojan horses, and worms. In a cloud environment, newer attacks may result from hypervisor malware, guest hopping and hijacking, or VM rootkits.

Another type of attack is the man-in-the-middle attack for VM migrations. In general, passive attacks steal sensitive data or passwords. Active attacks may manipulate kernel data structures which will cause major damage to cloud servers.

- ***Challenge 3—Unpredictable Performance and Bottlenecks***

Multiple VMs can share CPUs and main memory in cloud computing, but I/O sharing is problematic. For example, to run 75 EC2 instances with the STREAM benchmark requires a mean bandwidth of 1,355 MB/second. However, for each of the 75

EC2 instances to write 1 GB files to the local disk requires a mean disk write bandwidth of only 55 MB/second. This demonstrates the problem of I/O interference between VMs. One solution is to improve I/O architectures and operating systems to

efficiently virtualize interrupts and I/O channels. Internet applications continue to become more data-intensive. If we assume applications to be "pulled apart" across the

boundaries of clouds, this may complicate data placement and transport. Cloud users and providers have to think about the

implications of placement and traffic at every level of the system, if they want to minimize costs. This kind of reasoning can be seen in Amazon's development of its new CloudFront service. Therefore, data transfer bottlenecks must be removed, bottleneck links must be widened, and weak servers should be removed.

- ***Challenge 4—Distributed Storage and Widespread Software Bugs***

The database is always growing in cloud applications. The opportunity is to create a storage system that will not only meet this growth, but also combine it with the cloud advantage of scaling arbitrarily up and down on demand. This demands the

design of efficient distributed SANs. Data centers must meet programmers' expectations in terms of scalability, data durability, and HA. Data consistence checking in SAN-connected data centers is a major challenge in cloud computing.

Large-scale distributed bugs cannot be reproduced, so the debugging must occur at a scale in the production data centers.

No data center will provide such a convenience. One solution may be a reliance on using VMs in cloud computing. The level of virtualization may make it possible to capture valuable information in ways that are impossible without using VMs. Debugging over simulators is another approach to attacking the problem, if the simulator is well designed.

- ***Challenge 5—Cloud Scalability, Interoperability, and Standardization***

The pay-as-you-go model applies to storage and network bandwidth; both are counted in terms of the number of bytes used. Computation is different depending on virtualization level. GAE automatically scales in response to load increases and decreases; users are charged by the cycles used. AWS charges by the hour for the number of VM instances used, even if the machine is idle. The opportunity here is to scale quickly up and down in response to load variation, in order to save money,

but without violating SLAs.

Open Virtualization Format (OVF) describes an open, secure, portable, efficient, and extensible format for the packaging and distribution of VMs. It also defines a format for distributing software to be deployed in VMs. This VM format does not rely

on the use of a specific host platform, virtualization platform, or guest operating system. The approach is to address virtual platform-agnostic packaging with certification and integrity of packaged software. The package supports virtual appliances to span more than one VM.

OVF also defines a transport mechanism for VM templates, and can apply to different virtualization platforms with different levels of virtualization. In terms of cloud standardization, we suggest the ability for virtual appliances to run on any virtual

platform. We also need to enable VMs to run on heterogeneous hardware platform hypervisors. This requires hypervisoragnostic VMs. We also need to realize cross-platform live migration between x86 Intel and AMD technologies and support

legacy hardware for load balancing. All these issue are wide open for further research.

- ***Challenge 6—Software Licensing and Reputation Sharing***

Many cloud computing providers originally relied on open source software because the licensing model for commercial software is not ideal for utility computing. The primary opportunity is either for open source to remain popular or simply for commercial software companies to change their licensing structure to better fit cloud computing. One can consider using both pay-for-use and bulk-use licensing schemes to widen the business coverage.

One customer's bad behavior can affect the reputation of the entire cloud. For instance, blacklisting of EC2 IP addresses by spam-prevention services may limit smooth VM installation. An opportunity would be to create reputation-guarding services

similar to the "trusted e-mail" services currently offered (for a fee) to services hosted on smaller ISPs. Another legal

issue concerns the transfer of legal liability. Cloud providers want legal liability to remain with the customer, and vice versa.

	This problem must be solved at the SLA level. We will study reputation systems for protecting data centers in the next section.

UNIT 3

45)Explain the service offerings of the following five major cloud platforms: IBM, Google, Microsoft, Amazon, Salesforce.

Cloud services rely on new advances in machine virtualization, SOA, grid infrastructure management, and power efficiency.Consumers purchase such services in the form of IaaS, PaaS, or SaaS as described earlier. Also, many cloud entrepreneurs are selling value-added utility services to massive numbers of users.

The cloud industry leverages the growing demand by many enterprises and business users to outsource their computing and storage jobs to professional providers. The provider service charges are often much lower than the cost for users to replace their obsolete servers frequently. Table summarizes the profiles of five major cloud providers by 2010 standards.

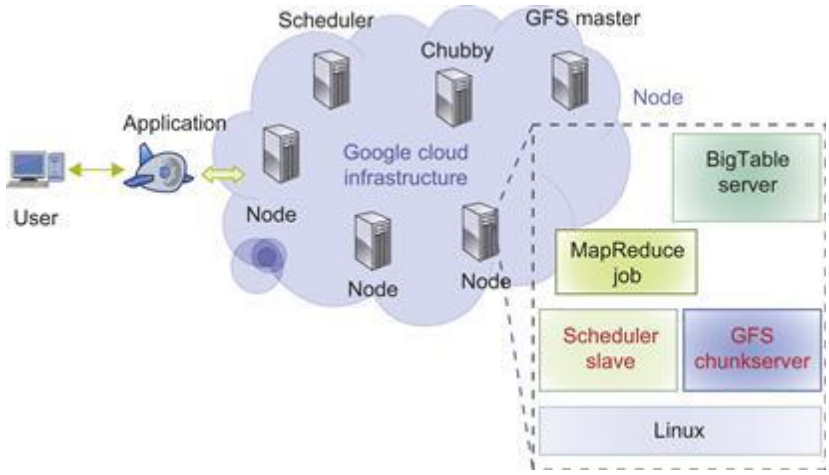
Five Major Cloud Platforms and Their Service Offerings

Model	IBM	Amazon	Google	Microsoft	Salesforce
PaaS	BlueCloud, WCA, RC2		App Engine (GAE)	Windows Azure	Force.com
IaaS	Ensembles	AWS		Windows Azure	
SaaS	Lotus Live		Gmail, Docs	.NET service, Dynamic CRM	Online CRM, Gifttag
Virtualization		OS and Xen	Application Container	OS level/ Hypel-V	
Service Offerings	SOA, B2, TSAM, RAD, Web 2.0	EC2, S3, SQS, SimpleDB	GFS, Chubby, BigTable, MapReduce	Live, SQL Hotmail	Apex, visual force, record security
Security Features	WebSphere2 and PowerVM tuned for protection	PKI, VPN, EBS to recover from failure	Chubby locks for security enforcement	Replicated data, rule-based access control	Admin./record security, uses metadata API
User Interfaces		EC2 command-line tools	Web-based admin. console	Windows Azure portal	
Web API	Yes	Yes	Yes	Yes	Yes
Programming Support	AMI		Python	.NET Framework	

46)Explain the architecture, functional modules and applications of Google App Engine.

Architecture

Figure shows the major building blocks of the Google cloud platform which has been used to deliver the cloud services highlighted earlier. GFS is used for storing large amounts of data. MapReduce is for use in application program development. Chubby is used for distributed application lock services. BigTable offers a storage service for accessing structured data. Users can interact with Google applications via the web interface provided by each application. Third-party application providers can use GAE to build cloud applications for providing services. The applications all run in data centers under tight management by Google engineers. Inside each data center, there are thousands of servers forming different clusters.



functional modules

The GAE platform comprises the following five major components:

a. The *datastore* offers object-oriented, distributed, structured data storage services based on BigTable techniques. The datastore secures data management operations.

b. The *application runtime environment* offers a platform for scalable web programming and execution. It supports two development languages: Python and Java.

c. The *software development kit* (SDK) is used for local application development. The SDK allows users to execute test runs of local applications and upload application code.

d. The *administration console* is used for easy management of user application development cycles, instead of for physical resource management.

e. The *GAE web service infrastructure* provides special interfaces to guarantee flexible use and management of storage and network resources by GAE.

applications of Google App Engine

Well-known GAE applications include the Google Search Engine, Google Docs, Google Earth, and Gmail. These applications can support large numbers of users simultaneously. Users can interact with Google applications via the web interface provided by each

application. Third-party application providers can use GAE to build cloud applications for providing services.

The applications are all run in the Google data centers. Inside each data center, there might be thousands of server nodes to form different clusters. (See the previous section.) Each cluster can run multipurpose servers. GAE supports many web applications. One is a storage service to store application-specific data in the Google infrastructure. The data can be persistently stored in the backend storage server while still providing the facility for queries, sorting, and even transactions similar to traditional database systems.

GAE also provides Google-specific services, such as the Gmail account service (which is the login service, that is, applications can use the Gmail account directly). This can eliminate the tedious work of building customized user management components in web applications. Thus, web applications built on top of GAE can use the APIs authenticating users and sending e-mail using Google accounts.

47)With the help of the diagram, explain the architecture of Microsoft Windows Azure .

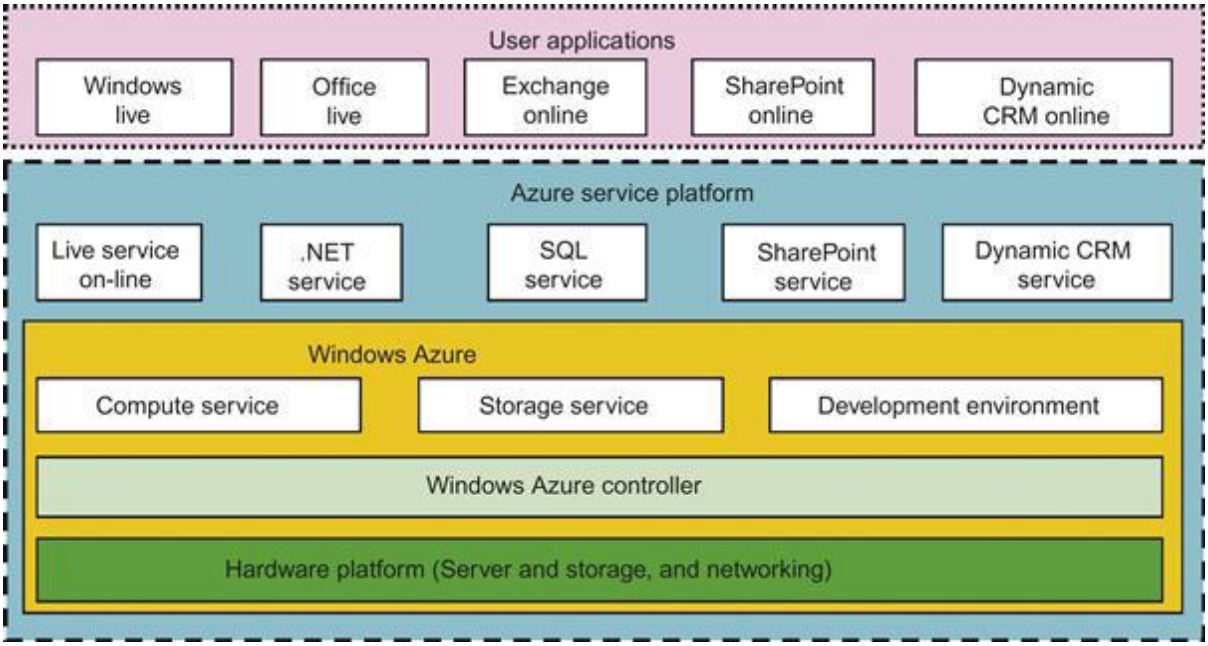
The platform is divided into three major component platforms. Windows Azure offers a cloud platform built on Windows OS and based on Microsoft virtualization technology. Applications are installed on VMs deployed on the data-center servers. Azure manages all servers,

storage, and network resources of the data center. On top of the infrastructure are the various services for building different cloud applications.

Cloud-level services provided by the Azure platform are introduced below:

• **Live service** Users can visit Microsoft Live applications and apply the data involved across multiple machines concurrently.
• **.NET service** This package supports application development on local hosts and execution on cloud machines.
• **SQL Azure** This function makes it easier for users to visit and use the relational database associated with the SQL server
in the cloud.
• **SharePoint service** This provides a scalable and manageable platform for users to develop their special business applications
in upgraded web services.
• **Dynamic CRM service** This provides software developers a business platform in managing CRM applications in financing, marketing, and sales and promotions.

All these cloud services in Azure can interact with traditional Microsoft software applications, such as Windows Live, Office Live, Exchange online, SharePoint online, and dynamic CRM online. The Azure platform applies the standard web communication protocols SOAP and REST.



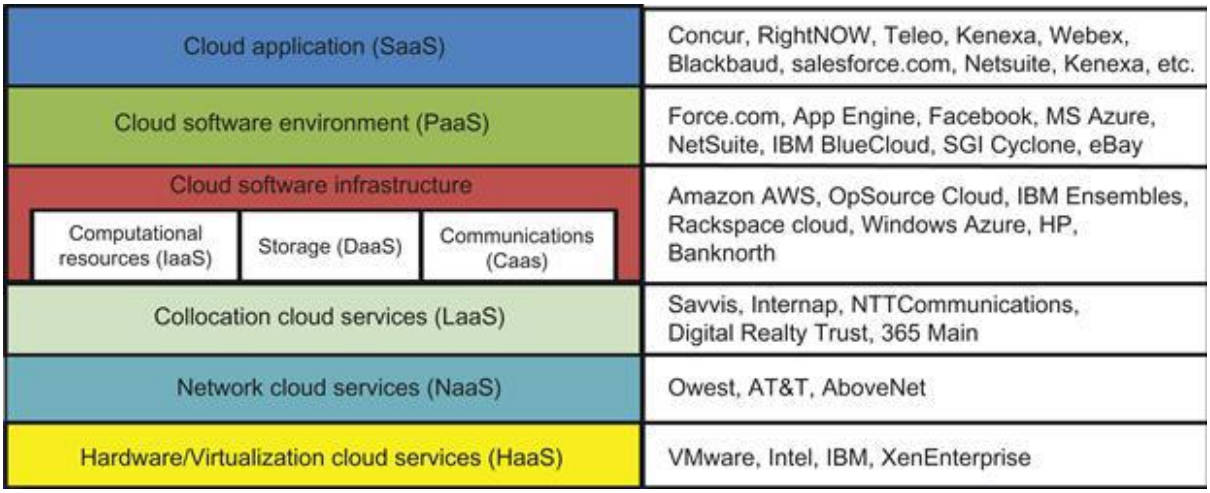
48)Explain the six layers of cloud services.

Figure shows six layers of cloud services, ranging from hardware, network, and collocation to infrastructure, platform, and software applications. We already introduced the top three service layers as SaaS, PaaS, and IaaS, respectively.

The cloud platform provides PaaS, which sits on top of the IaaS infrastructure. The top layer offers SaaS. These must be implemented on the cloud platforms provided. The implication is that one cannot launch SaaS applications with a cloud platform. The cloud platform cannot be built if compute and storage infrastructures are not there. The bottom three layers are more related to physical requirements. The bottommost layer provides *Hardware as a Service (HaaS).*

The next layer is for interconnecting all the hardware components, and is simply called *Network as a Service (NaaS).Virtual LANs* fall within the scope of NaaS. The next layer up offers *Location as a Service (LaaS),* which provides a collocation service to house, power, and secure all the physical hardware and network resources. Some authors say this layer provides *Security as a Service* ("SaaS").

The cloud infrastructure layer can be further subdivided as *Data as a Service (DaaS)* and *Communication as a Service (CaaS)* in addition to compute and storage in IaaS.



49)How is efficient provisioning of resources done? What are different methods of resource provisioning?

Efficient VM provisioning depends on the cloud architecture and management of cloud infrastructures. Resource provisioning schemes also demand fast discovery of services and data in cloud computing infrastructures.

In a virtualized cluster of servers, this demands efficient installation of VMs, live VM migration, and fast recovery from failures. To deploy VMs, users treat them as physical hosts with customized operating systems for specific applications.

For example, Amazon's EC2 uses Xen as the virtual machine monitor (VMM). The same VMM is used in IBM's Blue Cloud. In the EC2 platform, some predefined VM templates are also provided.

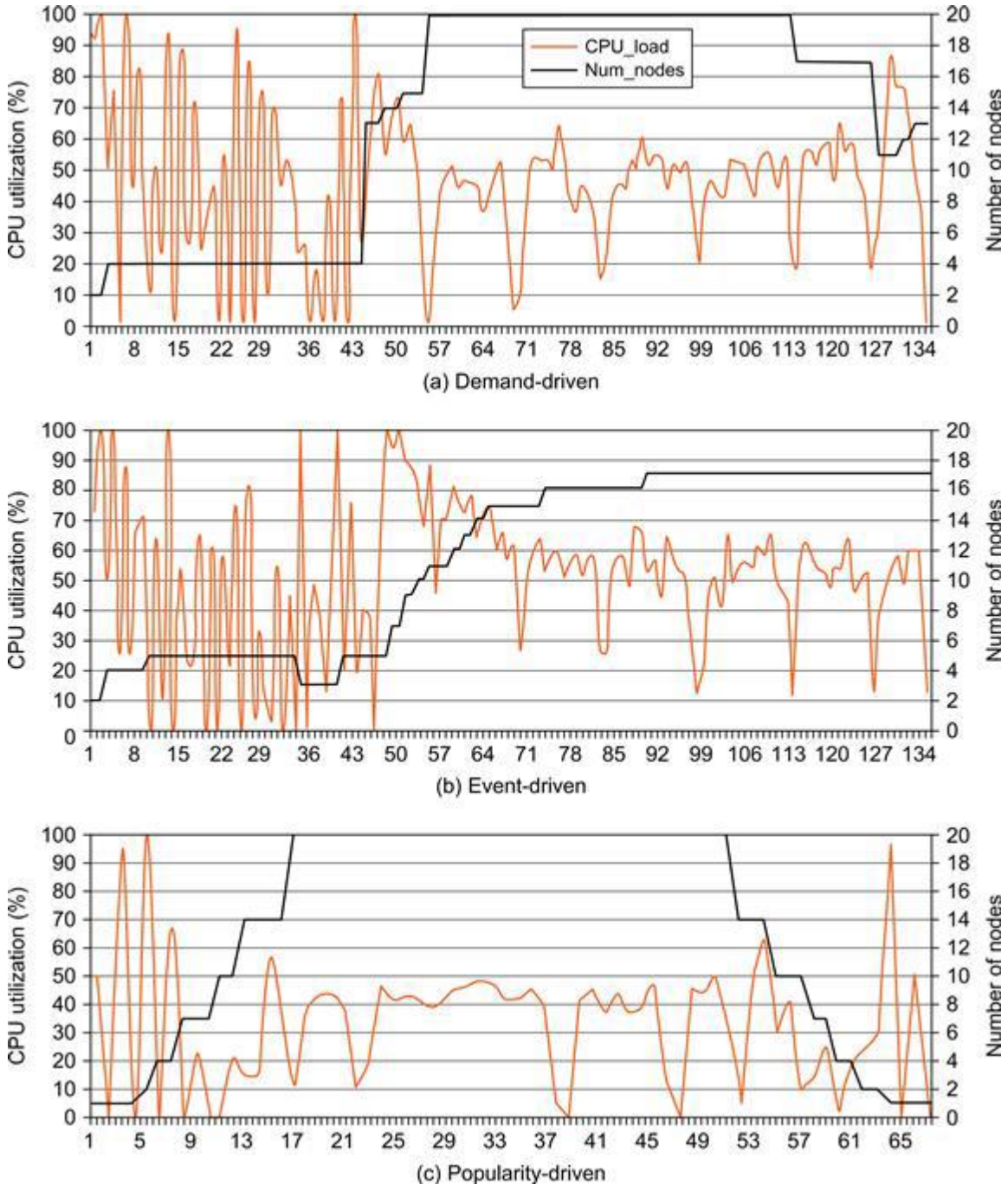
Users can choose different kinds of VMs from the templates. IBM's Blue Cloud does not provide any VM templates. In general, any type of VM can run on top of Xen. Microsoft also applies virtualization in its Azure cloud platform. The provider should offer resource-economic services.

Powerefficient schemes for caching, query processing, and thermal management are mandatory due to increasing energy waste by heat dissipation from data centers. Public or private clouds promise to streamline the on-demand provisioning of software, hardware, and data as a service, achieving economies of scale in IT deployment and operation.

methods of resource provisioning

Three resource-provisioning methods are presented in the following sections. The *demand-driven method* provides static resources and has been used in grid computing for many years. The *event-driven method* is based on predicted workload by time. The *popularity-driven method* is based on Internet traffic monitored.

We characterize these resource provisioning methods as follows (see Figure).



50)Enumerate the steps by which an intergrid gateway (IGG) allocates resources from a local cluster to deploy applications. Explain with the help of diagram.

OR

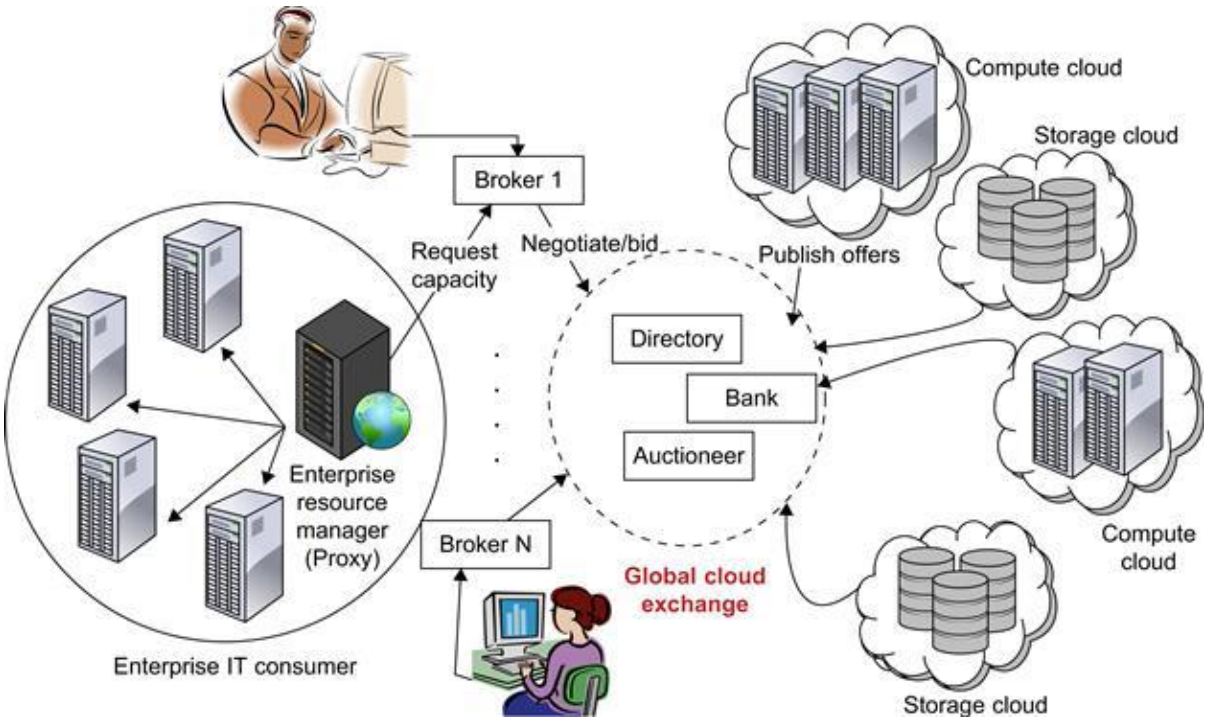
54)With the help of a diagram, explain the interactions between the components of Intergrid.

Figure illustrates the interactions between InterGrid's components. A distributed VM manager makes requests for VMs and queries their status. This manager requests VMs from the gateway on behalf of the user application.

The manager obtains the list of requested VMs from the gateway. This list contains a tuple of public IP/private IP addresses for each VM with Secure Shell (SSH) tunnels. Users must specify which VM template they want to use and the number of VM instances needed, the deadline, the wall time, and the address for an alternative gateway.

The local gateway tries to obtain resources from the underlying VIEs. When this is impossible, the local gateway starts a negotiation with any remote gateways to fulfill the request. When a gateway schedules the VMs, it sends the VM access information to the requester gateway.

Finally, the manager configures the VM, sets up SSH tunnels, and executes the tasks on the VM. Under the peering policy, each gateway's scheduler uses conservative backfilling to schedule requests. When the scheduler can't start a request immediately using local resources, a redirection algorithm will be initiated.



51)How is provisioning of storage resources done in cloud computing systems? Explain with examples.

Provisioning of Storage Resources

The data storage layer is built on top of the physical or virtual servers. As the cloud computing applications often provide service to users, it is unavoidable that the data is stored in the clusters of the cloud provider. The service can be accessed anywhere in the world. One example is e-mail systems.

A typical large e-mail system might have millions of users and ach user can have thousands of e-mails and consume multiple gigabytes of disk space. Another example is a web searching application. In storage technologies, hard disk drives may be augmented with solid-state drives in the future.

This will provide reliable and high-performance data storage. The biggest barriers to adopting flash memory in data centers have been price, capacity, and, to some extent, a lack of sophisticated query-processing techniques.

However, this is about to change as the I/O bandwidth of solid-state drives becomes too impressive to ignore. A distributed file system is very important for storing large-scale data. However, other forms of data storage also exist. Some data does not need the namespace of a tree structure file system, and instead, databases are built with stored data files.

In cloud computing, another form of data storage is (Key, Value) pairs. Amazon S3 service uses SOAP to access the objects stored in the cloud. Following shows three cloud storage services provided by Google, Hadoop, and Amazon.

Storage Services in Three Cloud Computing Systems

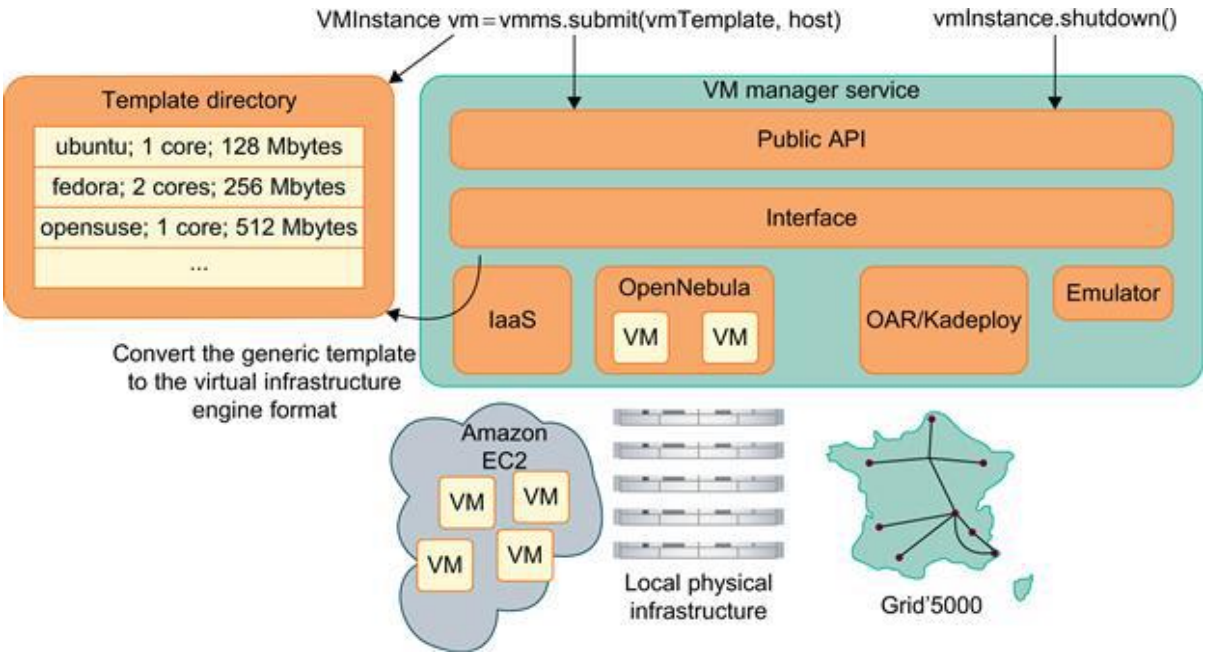
GFS: Google File System - Very large sustainable reading and writing bandwidth, mostly continuous accessing instead of random accessing. The programming interface is similar to that of the POSIX file system accessing interface.

HDFS: Hadoop Distributed File System - The open source clone of GFS Written in Java. The programming interfaces are similar to POSIX but not identical.

Amazon S3 and EBS - S3 is used for retrieving and storing data from/to remote servers. EBS is built on top of S3 for using virtual disks in running EC2 instances.

52)With the help of a diagram, explain the interaction Virtual Machine Managers for cloud creation and management.

Figure shows the interactions among VM managers for cloud creation and management. The managers provide a public API for users to submit and control the VMs.



Independent Service Management

Independent services request facilities to execute many unrelated tasks. Commonly, the APIs provided are some web services that the developer can use conveniently. In Amazon cloud computing infrastructure, SQS is constructed for providing a reliable communication service between different providers. Even the endpoint does not run while another entity has posted a message in SQS.

By using independent service providers, the cloud applications can run different services at the same time. Some other services are used for providing data other than the compute or storage services.

Running Third-Party Applications

Cloud platforms have to provide support for building applications that are constructed by third-party application providers or programmers. As current web applications are often provided by using Web 2.0 forms (interactive applications with Ajax), the programming interfaces are different from the traditional programming interfaces such as functions in runtime libraries.

The APIs are often in the form of services. Web service application engines are often used by programmers for building applications. The web browsers are the user interface for end users.

Virtual Machine Manager

The VM manager is the link between the gateway and resources. The gateway doesn't share physical resources directly, but relies on virtualization technology for abstracting them. Hence, the actual resources it uses are VMs.

The manager manage VMs deployed on a set of physical resources. The VM manager implementation is generic so that it can connect with different VIEs. Typically, VIEs can create and stop VMs on a physical cluster. The Melbourne group has developed managers for OpenNebula, Amazon EC2, and French Grid'5000.

Virtual Machine Templates

A *VM template* is analogous to a computer's configuration and contains a description for a VM with the following static information:

- The number of cores or processors to be assigned to the VM
- The amount of memory the VM requires
- The kernel used to boot the VM's operating system
- The disk image containing the VM's file system
- The price per hour of using a VM

The gateway administrator provides the VM template information when the infrastructure is set up. The administrator can update, add, and delete templates at any time. In addition, each gateway in the InterGrid network must agree on the templates to provide the same configuration on each site. To deploy an instance of a given VM, the VMM generates a descriptor from the template. This descriptor contains the same fields as the template and additional information related to a specific VM instance.

Distributed VM Management

A distributed VM manager makes requests for VMs and queries their status. This manager requests VMs from the gateway on behalf of the user application. The manager obtains the list of requested VMs from the gateway. This list contains a tuple of public IP/private IP addresses for each VM with Secure Shell (SSH) tunnels. Users must specify which VM template they want to use and the number of VM instances needed, the deadline, the wall time, and the address for an alternative gateway. The local gateway tries to obtain resources from the underlying VIEs.

When this is impossible, the local gateway starts a negotiation with any remote gateways to fulfill the request. When a gateway schedules the VMs, it sends the VM access information to the requester gateway. Finally, the manager configures the VM, sets up SSH tunnels, and executes the tasks on the VM. Under the peering policy, each gateway's scheduler uses conservative backfilling to schedule requests. When the scheduler can't start a request immediately using local resources, a redirection algorithm will be initiated.

53)What are virtual machine templates? What do they contain?

A *VM template* is analogous to a computer's configuration and contains a description for a VM with the following static information:

- The number of cores or processors to be assigned to the VM
- The amount of memory the VM requires
- The kernel used to boot the VM's operating system
- The disk image containing the VM's file system
- The price per hour of using a VM

The gateway administrator provides the VM template information when the infrastructure is set up. The administrator can update, add, and delete templates at any time. In addition, each gateway in the InterGrid network must agree on the templates to provide the same configuration on each site. To deploy an instance of a given VM, the VMM generates a descriptor from the template. This descriptor contains the same fields as the template and additional information related to a specific VM instance.

Typically the additional information includes:

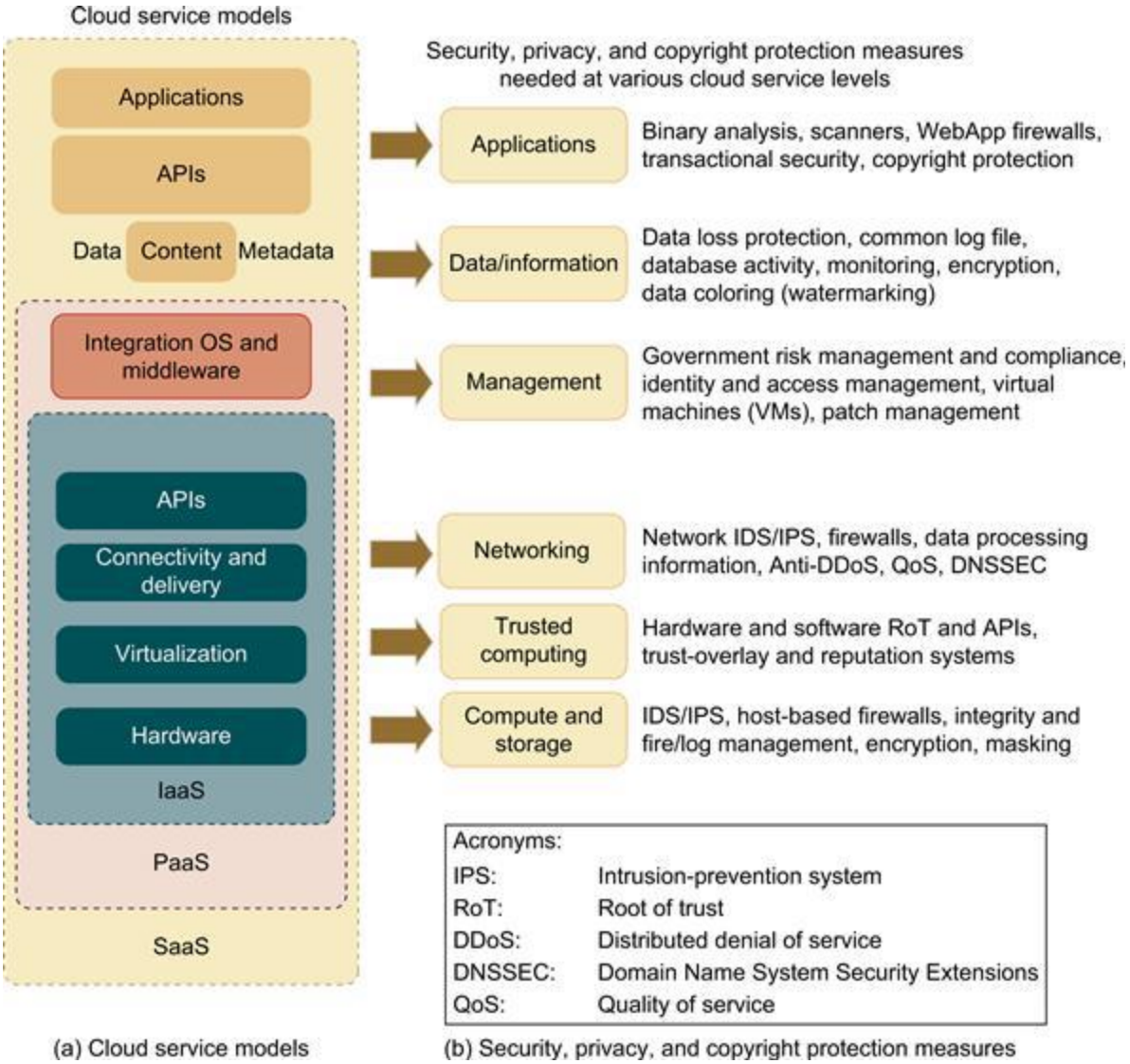
- The disk image that contains the VM's file system
- The address of the physical machine hosting the VM
- The VM's network configuration
- The required information for deployment on an IaaS provider

Before starting an instance, the scheduler gives the network configuration and the host's address; it then allocates MAC and IP addresses for that instance. The template specifies the disk image field. To deploy several instances of the same VM template in parallel, each instance uses a temporary copy of the disk image. Hence, the descriptor contains the path to the copied disk image. The descriptor's fields are different for deploying a VM on an IaaS provider. Network information is not needed, because Amazon EC2 automatically assigns a public IP to the instances. The IGG works with a repository of VM templates, called the *VM template directory*.

55)Explain the security measures corresponding to different cloud service models.

Three basic cloud security enforcements are expected. First, facility security in data centers demands on-site security year round. Biometric readers, CCTV (close-circuit TV), motion detection, and man traps are often deployed. Also, network security demands fault-tolerant external firewalls, intrusion detection systems (IDSes), and third-party vulnerability assessment. Finally, platform security demands SSL and data decryption, strict password policies, and system trust certification.

Figure shows the mapping of cloud models, where special security measures are deployed at various cloud operating levels.



Cloud service models on the left (a) and corresponding security measures on the right (b); the IaaS is at the innermost level, PaaS is at the middle level, and SaaS is at the outermost level, including all hardware, software, datasets, and networking resources.

Servers in the cloud can be physical machines or VMs. User interfaces are applied to request services. The provisioning tool carves out the systems from the cloud to satisfy the requested service.

A security-aware cloud architecture demands security enforcement. Malware-based attacks such as network worms, viruses, and DDoS attacks exploit system vulnerabilities.

These attacks compromise system functionality or provide intruders unauthorized access to critical information. Thus, security defenses are needed to protect all cluster servers and data centers.
Here are some cloud components that demand special security protection:
• Protection of servers from malicious software attacks such as worms, viruses, and malware
• Protection of hypervisors or VM monitors from software-based attacks and vulnerabilities
• Protection of VMs and monitors from service disruption and DoS attacks
• Protection of data and information from theft, corruption, and natural disasters
• Providing authenticated and authorized access to critical data and services

56)Explain the different protection schemes for physical and cyber security at data centres.

Physical and Cyber Security Protection at Cloud/Data Centers

Secure data centers and computer buildings - Choose hazard-free location, enforce building safety. Avoid windows, keep buffer zone around the site, bomb detection, camera surveillance, earthquake-proof, etc.

Use redundant utilities at multiple sites - Multiple power and supplies, alternate network connections, multiple databases at separate sites, data consistency, data watermarking, user authentication, etc.

Trust delegation and negotiation - Cross certificates to delegate trust across PKI domains for various data centers, trust negotiation among certificate authorities (CAs) to resolve policy conflicts

Worm containment and DDoS defense - Internet worm containment and distributed defense against DDoS attacks to secure all data centers and cloud platforms

Reputation system for data centers - Reputation system could be built with P2P technology; one can build a hierarchy of reputation systems from data centers to distributed file systems

Fine-grained file access control - Fine-grained access control at the file or object level; this adds to security protection beyond firewalls and IDSes

Copyright protection and piracy prevention - Piracy prevention achieved with peer collusion prevention, filtering of poisoned content, nondestructive read, alteration detection, etc.

Privacy protection - Uses double authentication, biometric identification, intrusion detection and disaster recovery, privacy enforcement by data watermarking, data classification, etc.

57)How does virtualization help in cloud security? Enumerate security features desired in secure cloud.

Three basic cloud security enforcements are expected. First, facility security in data centers demands on-site security year round. Biometric readers, CCTV (close-circuit TV), motion detection, and man traps are often deployed. Also, network security demands fault-tolerant external firewalls, intrusion detection systems (IDSes), and third-party vulnerability assessment. Finally, platform security demands SSL and data decryption, strict password policies, and system trust certification.

Servers in the cloud can be physical machines or VMs. User interfaces are applied to request services. The provisioning tool carves out the systems from the cloud to satisfy the requested service.

A security-aware cloud architecture demands security enforcement. Malware-based attacks such as network worms, viruses, and DDoS attacks exploit system vulnerabilities.
These attacks compromise system functionality or provide intruders unauthorized access to critical information. Thus, security defenses are needed to protect all cluster servers and data centers.

Here are some cloud components that demand special security protection:
• Protection of servers from malicious software attacks such as worms, viruses, and malware
• Protection of hypervisors or VM monitors from software-based attacks and vulnerabilities
• Protection of VMs and monitors from service disruption and DoS attacks
• Protection of data and information from theft, corruption, and natural disasters
• Providing authenticated and authorized access to critical data and services

Virtualization enhances cloud security. But VMs add an additional layer of software that could become a single point of failure. With virtualization, a single physical machine can be divided or partitioned into multiple VMs (e.g., server consolidation).

This provides each VM with better security isolation and each partition is protected from DoS attacks by other partitions. Security attacks in one VM are isolated and contained from affecting the other VMs.

VM failures do not propagate to other VMs. The hypervisor provides visibility of the guest OS, with complete guest isolation. Fault containment and failure isolation of VMs provide a more secure and robust environment. Malicious intrusions may destroy valuable hosts, networks, and storage resources.

Internet anomalies found in routers, gateways, and distributed hosts may stop cloud services. Trust negotiation is often done at the SLA level. Public Key Infrastructure (PKI) services could be augmented with data-center reputation systems. Worm and DDoS attacks must be contained. It is harder to establish security in the cloud because all data and software are shared by default.

The VM is decoupled from the physical hardware. The entire VM can be represented as a software component and can be regarded as binary or digital data. The VM can be saved, cloned, encrypted, moved, or restored with ease. VMs enable HA and faster disaster recovery.

Live migration of VMs was suggested by many researchers for building *distributed intrusion detection systems (DIDSes)*. Multiple IDS VMs can be deployed at various resource sites including data centers. DIDS design demands trust negation among PKI domains. Security policy conflicts must be resolved at design time and updated periodically.

58)How is distributed defense provided against Distributed Denial of service flooding attacks?

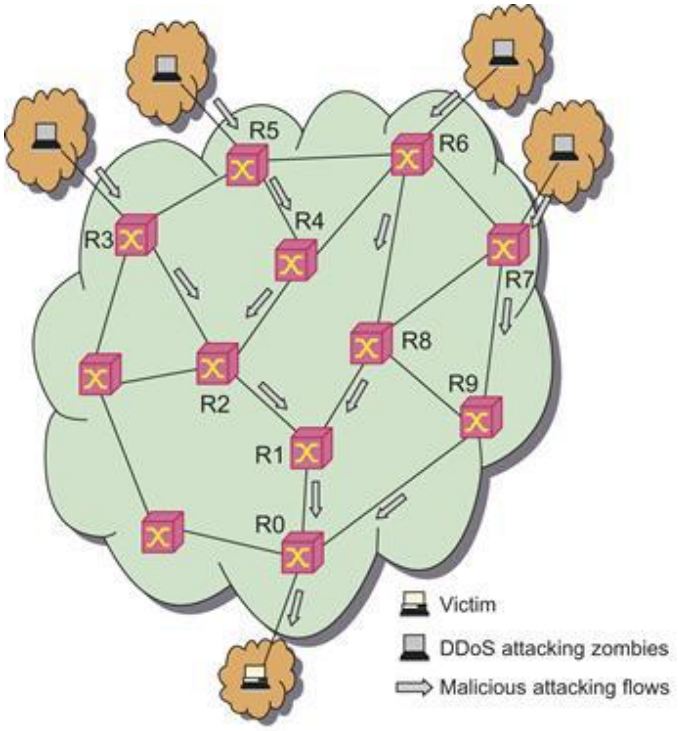
A DDoS defense system must be designed to cover multiple network domains spanned by a given cloud platform. These network domains cover the edge networks where cloud resources are connected.

DDoS attacks come with widespread worms. The flooding traffic is large enough to crash the victim server by buffer overflow, disk exhaustion, or connection saturation. Figure shows a flooding attack pattern. Here, the hidden attacker launched the attack from many zombies toward a victim server at the bottom router R0.

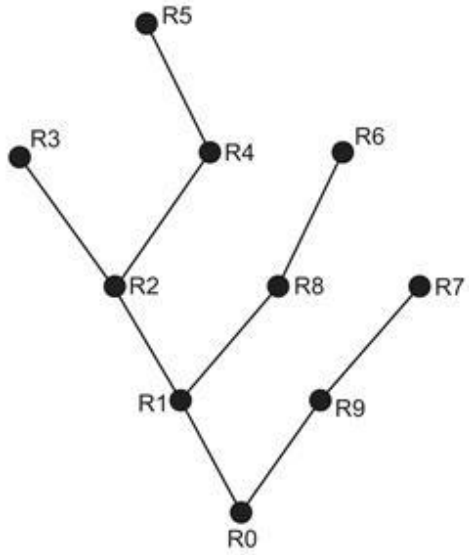
The flooding traffic flows essentially with a tree pattern shown in Figure . Successive attack-transit routers along the tree reveal the abnormal surge in traffic. This DDoS defense system is based on change-point detection by all routers.

Based on the anomaly pattern detected in covered network domains, the scheme detects a DDoS attack before the victim is overwhelmed. The detection scheme is suitable for protecting cloud core networks. The provider-level cooperation eliminates the need for intervention by edge networks.

DDoS attacks and defense by change-point detection at all routers on the flooding tree



(a) Traffic flow pattern of a DDoS attack



(b) The attack traffic flow tree over 10 routers

59)Explain the different data and software protection techniques in cloud environment.

Data and Software Protection Techniques

Data Integrity and Privacy Protection

Users desire a software environment that provides many useful tools to build cloud applications over large data sets.

In addition to application software for MapReduce, BigTable, EC2, 3S, Hadoop, AWS, GAE, and WebSphere2, users need some security and privacy protection software for using the cloud.

Such software should offer the following features:

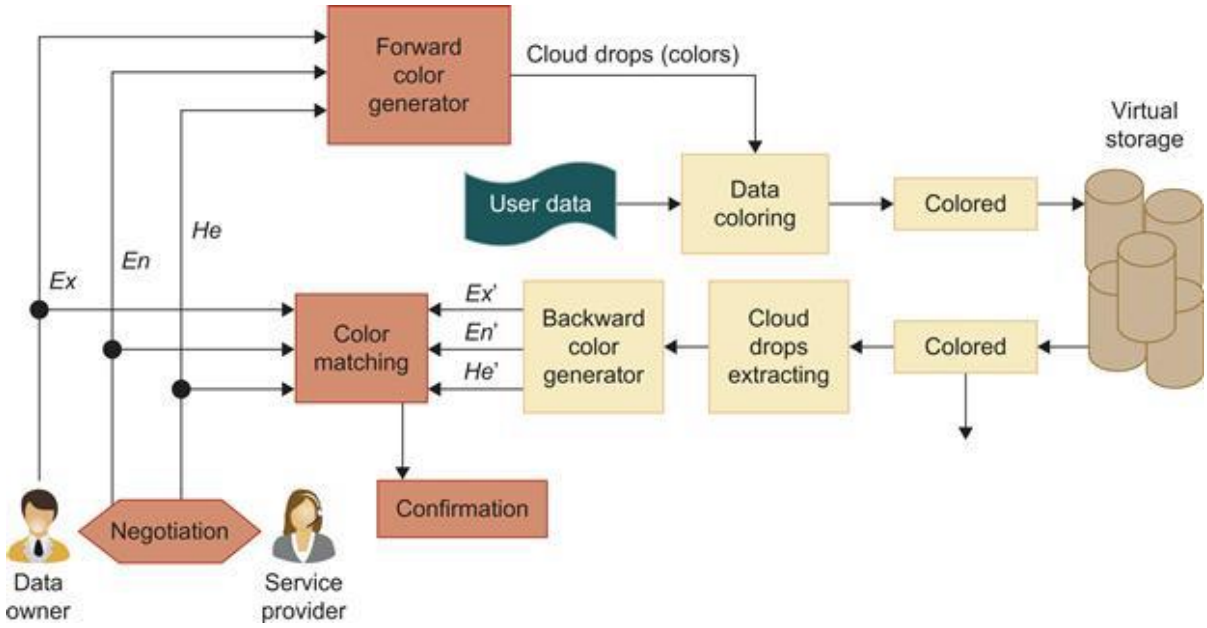
- Special APIs for authenticating users and sending e-mail using commercial accounts
- Fine-grained access control to protect data integrity and deter intruders or hackers
- Shared data sets protected from malicious alteration, deletion, or copyright violation
- Ability to secure the ISP or cloud service provider from invading users' privacy
- Personal firewalls at user ends to keep shared data sets from Java, JavaScript, and ActiveX applets
- A privacy policy consistent with the cloud service provider's policy, to protect against identity theft, spyware, and web bugs
- VPN channels between resource sites to secure transmission of critical data objects

Data Coloring and Cloud Watermarking

With shared files and data sets, privacy, security, and copyright information could be compromised in a cloud computing environment. Users desire to work in a trusted software environment that provides useful tools to build cloud applications over protected data sets. In the past, watermarking was mainly used for digital copyright management.

As shown in Figure, the system generates special colors for each data object. Data coloring means labeling each data object by a unique color. Differently colored data objects are thus distinguishable. The user identification is also colored to be matched with the data colors.

This color matching process can be applied to implement different trust management events. Cloud storage provides a process for the generation, embedding, and extraction of the watermarks in colored objects.



Data Lock-in Problem and Proactive Solutions

Cloud computing moves both the computation and the data to the server clusters maintained by cloud service providers. Once the data is moved into the cloud, users cannot easily extract their data and programs from cloud servers to run on another platform. This leads to a data lock-in problem.

This has hindered the use of cloud computing. Data lock-in is attributed to two causes: lack of interoperability, whereby each cloud vendor has its proprietary API that limits users to extract data once submitted; and lack of application compatibility, in that most computing clouds expect users to write new applications from scratch, when they switch cloud platforms.

One possible solution to data lock-in is the use of standardized cloud APIs. This requires building standardized virtual platforms that adhere to OVF, a platform-independent, efficient, extensible, and open format for VMs. This will enable efficient, secure software distribution, facilitating the mobility of VMs. Using OVF one can move data from one application to another.

This will enhance QoS, and thus enable cross-cloud applications, allowing workload migration among data centers to user-specific storage. By deploying applications, users can access and intermix applications across different cloud services.

60)Explain the capability needs of commercial clouds.

Public opinion on the character or standing (such as honest behavior or reliability) of an entity could be the reputation of a person, an agent, a product, or a service. It represents a collective evaluation by a group of people/agents and resource owners.

Many reputation systems have been proposed in the past mainly for P2P, multiagent, or e-commerce systems. To address reputation systems for cloud services, a systematic approach is based on the design criteria and administration of the reputation systems.

Most of them were designed for P2P or social networks. These reputation systems can be converted for protecting cloud computing applications. In general, the reputation systems are classified as *centralized* or *distributed* depending on how they are implemented.

In a centralized system, a single central authority is responsible for managing the reputation system, while the distributed model involves multiple control centers working collectively. Reputation-based trust management and techniques for securing P2P and social networks could be merged to defend data centers and cloud platforms against attacks from the open network.

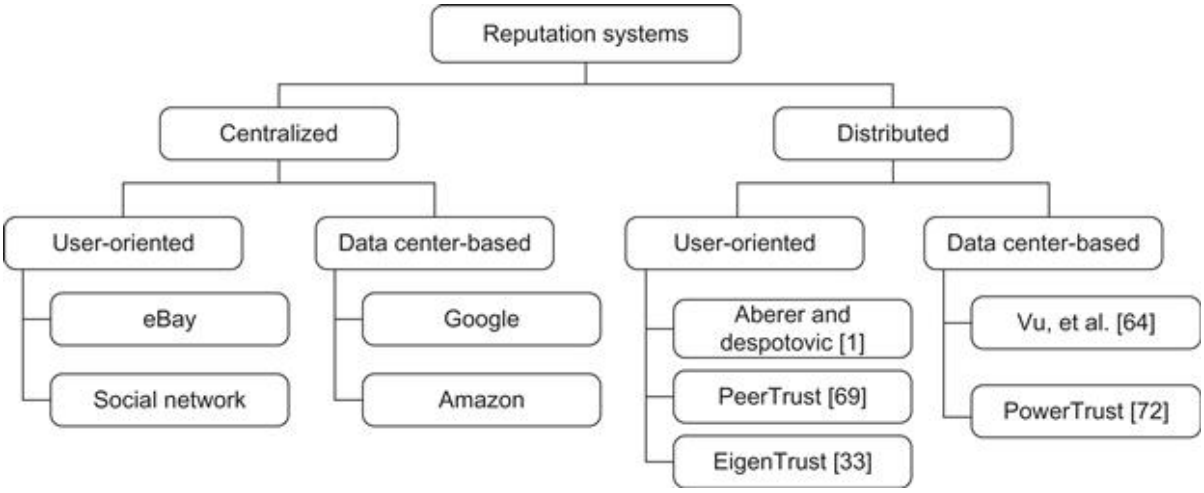
A centralized reputation system is easier to implement, but demands more powerful and reliable server resources; a distributed reputation system is much more complex to build. Distributed systems are more scalable and reliable in terms of handling failures.

At the second tier, reputation systems are further classified by the scope of reputation evaluation. *User-oriented* reputation systems focus on individual users or agents. Most P2P reputation systems belong to this category.

In data centers, reputation is modeled for the resource site as a whole. This reputation applies to products or services offered by the cloud. Commercial reputation systems have been built by eBay, Google, and Amazon in connection with the services they provide.

These are centralized reputation systems. Distributed reputation systems are mostly developed by academic research communities. Aberer and Despotovic have proposed a model to manage trust in P2P systems. The Eigentrust reputation system was developed at Stanford University using a trust matrix approach.

The PeerTrust system was developed at Georgia Institute of Technology for supporting ecommerce applications. The PowerTrust system was developed at the University of Southern California based on Power law characteristics of Internet traffic for P2P applications. Vu, et al. proposed a QoS-based ranking system for P2P transactions.



61)Enumerate the features of Infrastructure cloud.

Infrastructure Cloud Features

Accounting: Includes economies; clearly an active area for commercial clouds

Appliances: Preconfigured virtual machine (VM) image supporting multifaceted tasks such as messagepassing interface (MPI) clusters

Authentication and authorization: Could need single sign-on to multiple systems

Data transport: Transports data between job components both between and within grids and clouds; exploits custom storage patterns as in BitTorrent

Operating systems: Apple, Android, Linux, Windows

Program library: Stores images and other program material

Registry: Information resource for system (system version of metadata management)

Security: Security features other than basic authentication and authorization; includes higher level concepts such as trust

Scheduling: Basic staple of Condor, Platform, Oracle Grid Engine, etc.; clouds have this implicitly as is especially clear with Azure Worker Role

Gang scheduling: Assigns multiple (data-parallel) tasks in a scalable fashion; note that this is provided automatically by MapReduce

Software as a Service (SaaS): Shared between clouds and grids, and can be supported without special attention; Note use of services and corresponding service oriented architectures are very successful and are used in clouds very similarly to previous distributed systems.

Virtualization: Basic feature of clouds supporting "elastic" feature highlighted by Berkeley as characteristic of what defines a (public) cloud; includes virtual networking.

62)What are the traditional features of cluster, grid and parallel computing environments? Explain.

Traditional Features in Cluster, Grid, and Parallel Computing Environments

Cluster management: ROCKS and packages offering a range of tools to make it easy to bring up clusters

Data management: Included metadata support such as RDF triple stores (Semantic web success and can be built on MapReduce as in SHARD); SQL and NOSQL included in

Grid programming environment: Varies from link-together services as in Open Grid Services Architecture (OGSA) to GridRPC (Ninf, GridSolve) and SAGA

OpenMP/threading: Can include parallel compilers such as Cilk; roughly shared memory technologies. Even transactional memory and fine-grained data flow come here

Portals: Can be called (science) gateways and see an interesting change in technology from portlets to HUBzero and now in the cloud: Azure Web Roles and GAE

Scalable parallel computing environments: MPI and associated higher level concepts including ill-fated HP FORTRAN, PGAS (not successful but not disgraced), HPCS languages (X-10, Fortress, Chapel), patterns (including Berkeley dwarves), and functional languages such as F# for distributed memory

Virtual organizations: From specialized grid solutions to popular Web 2.0 capabilities such as Facebook

Workflow: Supports workflows that link job components either within or between grids and clouds; relate to LIMS Laboratory Information Management Systems.

63)What are the platform features supported by clouds? Explain.

Platform Features Supported by Clouds and (Sometimes) Grids

Blob: Basic storage concept typified by Azure Blob and Amazon S3

DPFS: Support of file systems such as Google (MapReduce), HDFS (Hadoop), and Cosmos (Dryad) with compute-data affinity optimized for data processing

Fault tolerance: this was largely ignored in grids, but is a major feature of clouds

MapReduce: Support MapReduce programming model including Hadoop on Linux, Dryad on Windows HPCS, and Twister on Windows and Linux. Include new associated languages such as Sawzall, Pregel, Pig Latin, and LINQ

Monitoring: Many grid solutions such as Inca. Can be based on publish-subscribe

Notification: Basic function of publish-subscribe systems

Programming model: Cloud programming models are built with other platform features and are related to familiar web and grid models

Queues: Queuing system possibly based on publish-subscribe

Scalable synchronization: Apache Zookeeper or Google Chubby. Supports distributed locks and used by BigTable. Not clear if (effectively) used in Azure Table or Amazon SimpleDB

SQL: Relational database

Table: Support of table data structures modeled on Apache Hbase or Amazon SimpleDB/Azure Table. Part of NOSQL movement

Web role: Used in Azure to describe important link to user and can be supported otherwise with a portal framework. This is the main purpose of GAE

Worker role: Implicitly used in both Amazon and grids but was first introduced as a high-level construct by Azure

64)Enlist the techniques related to security, privacy, and availability requirements for developing a healthy and dependable cloud programming environment.

Security, Privacy, and Availability

The following techniques are related to security, privacy, and availability requirements for developing a healthy and dependable cloud programming environment.

We summarize these techniques here:

• Use virtual clustering to achieve dynamic resource provisioning with minimum overhead cost.
• Use stable and persistent data storage with fast queries for information retrieval.
• Use special APIs for authenticating users and sending e-mail using commercial accounts.
• Cloud resources are accessed with security protocols such as HTTPS and SSL.
• Fine-grained access control is desired to protect data integrity and deter intruders or hackers.
• Shared data sets are protected from malicious alteration, deletion, or copyright violations.
• Features are included for availability enhancement and disaster recovery with life migration of VMs.
• Use a reputation system to protect data centers. This system only authorizes trusted clients and stops pirates.

65)Explain the system issues for running a typical parallel program in either a parallel or a distributed manner.

The system issues for running a typical parallel program in either a parallel or a distributed manner would include the following:

• **Partitioning** This is applicable to both computation and data as follows:

• **Computation partitioning** This splits a given job or a program into smaller tasks. Partitioning greatly depends on correctly identifying portions of the job or program that can be performed concurrently. In other words, upon identifying parallelism in the structure of the program, it can be divided into parts to be run on different workers. Different parts may process different data or a copy of the same data.

• **Data partitioning** This splits the input or intermediate data into smaller pieces. Similarly, upon identification of parallelism in the input data, it can also be divided into pieces to be processed on different workers. Data pieces may be processed by different parts of a program or a copy of the same program.

• **Mapping** This assigns the either smaller parts of a program or the smaller pieces of data to underlying resources. This process aims to appropriately assign such parts or pieces to be run simultaneously on different workers and is usually handled by resource allocators in the system.

• **Synchronization** Because different workers may perform different tasks, synchronization and coordination among workers is necessary so that race conditions are prevented and data dependency among different workers is properly managed. Multiple accesses to a shared resource by different workers may raise race conditions, whereas data dependency happens when a worker needs the processed data of other workers.

• **Communication** Because data dependency is one of the main reasons for communication among workers, communication is always triggered when the intermediate data is sent to workers.

• **Scheduling** For a job or program, when the number of computation parts (tasks) or data pieces is more than the number of available workers, a scheduler selects a sequence of tasks or data pieces to be assigned to the workers. It is worth noting that the resource allocator performs the actual mapping of the computation or data pieces to workers, while the scheduler only picks the next part from the queue of unassigned tasks based on a set of rules called the scheduling policy.

For multiple jobs or programs, a scheduler selects a sequence of jobs or programs to be run on the distributed computing system. In this case, scheduling is also necessary when system resources are not sufficient to simultaneously run multiple jobs or programs.

66)Give the formal definition of MapReduce. Explain its logical data-flow. Enumerate the steps for solving problems using MapReduce.

Formal Definition of MapReduce

The MapReduce software framework provides an abstraction layer with the data flow and flow of control to users, and hides the implementation of all data flow steps such as data partitioning, mapping, synchronization, communication, and scheduling. Here, although the data flow in such frameworks is predefined, the abstraction layer provides two well-defined interfaces in the form of two functions: *Map* and *Reduce*. These two main functions can be overridden by the user to achieve specific objectives. Therefore, the user overrides the *Map* and *Reduce* functions first and then invokes the provided *MapReduce* (*Spec, & Results*) function from the library to start the flow of data.

The MapReduce function, *MapReduce* (*Spec, & Results*), takes an important parameter which is a specification object, the *Spec*. This object is first initialized inside the user's program, and then the user writes code to fill it with the names of input and output files, as well as other optional tuning parameters.

This object is also filled with the name of the *Map* and *Reduce* functions to identify these user-defined functions to the MapReduce library. The overall structure of a user's program containing the Map, Reduce, and the Main functions is given below. The Map and Reduce are two major subroutines.

They will be called to implement the desired function performed in the main program.

Map Function (… .)
{
… …
}
Reduce Function (… .)
{
… …
}
Main Function (… .)
{
Initialize **Spec** object
… …
MapReduce (Spec, & Results)
}

MapReduce Logical Data Flow

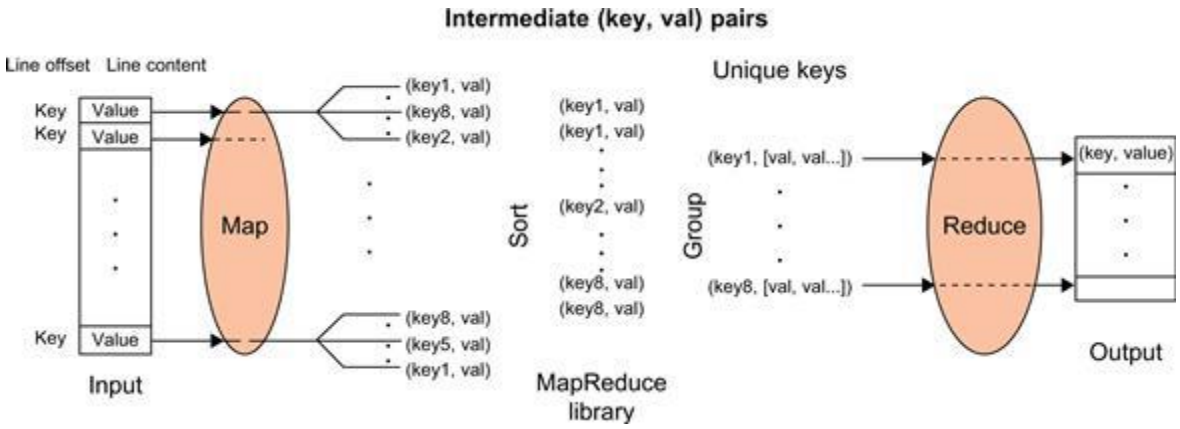
The input data to both the *Map* and the *Reduce* functions has a particular structure. This also pertains for the output data. The input data to the *Map* function is in the form of a (key, value) pair.

For example, the key is the line offset within the input file and the value is the content of the line. The output data from the *Map* function is structured as (key, value) pairs called intermediate (key, value) pairs.

In other words, the user-defined *Map* function processes each input (key, value) pair and produces a number of (zero, one, or more) intermediate (key, value) pairs. Here, the goal is to process all input (key, value) pairs to the *Map* function in parallel .

In turn, the *Reduce* function receives the intermediate (key, value) pairs in the form of a group of intermediate values associated with one intermediate key, *(key, [set of values])*. In fact, the MapReduce framework forms these groups by first sorting the intermediate (key, value) pairs and then grouping values with the same key.

It should be noted that the data is sorted to simplify the grouping process. The *Reduce* function processes each (key, [set of values]) group and produces a set of (key, value) pairs as output.



Strategy to Solve MapReduce Problems

As mentioned earlier, after grouping all the intermediate data, the values of all occurrences of the same key are sorted and grouped together. As a result, after grouping, each key becomes unique in all intermediate data. Therefore, finding unique keys is the starting point to solving a typical MapReduce problem. Then the intermediate (key, value) pairs as the output of the *Map* function will be automatically found.

The following three examples explain how to define keys and values in such problems:

Problem 1: Counting the number of occurrences of each word in a collection of documents
Solution: unique "key": each word, intermediate "value": number of occurrences

Problem 2: Counting the number of occurrences of words having the same size, or the same number of letters, in a collection of documents
Solution: unique "key": each word, intermediate "value": size of the word

Problem 3: Counting the number of occurrences of anagrams in a collection of documents. Anagrams are words with the same set of letters but in a different order (e.g., the words "listen" and "silent").
Solution: unique "key": alphabetically sorted sequence of letters for each word (e.g., "eilnst"), intermediate "value": number of occurrences.

67)Explain the architecture of Twister at runtime. (or Explain iterative MapReduce.)

Twister and Iterative MapReduce

It is important to understand the performance of different runtimes and, in particular, to compare MPI and MapReduce.

The two major sources of parallel overhead are load imbalance and communication. The communication overhead in MapReduce can be quite high, for two reasons:

• MapReduce reads and writes via files, whereas MPI transfers information directly between nodes over the network.

• MPI does not transfer all data from node to node, but just the amount needed to update information. We can call the MPI flow *δ flow* and the MapReduce flow *full data flow*.

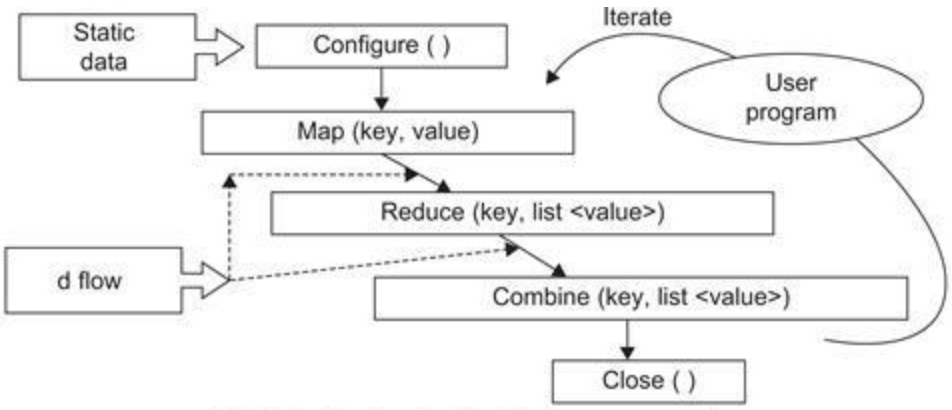
The same phenomenon is seen in all “classic parallel” loosely synchronous applications which typically exhibit an iteration structure over compute phases followed by communication phases.

We can address the performance issues with two important changes:
1. Stream information between steps without writing intermediate steps to disk.
2. Use long-running threads or processors to communicate the δ (between iterations) flow.

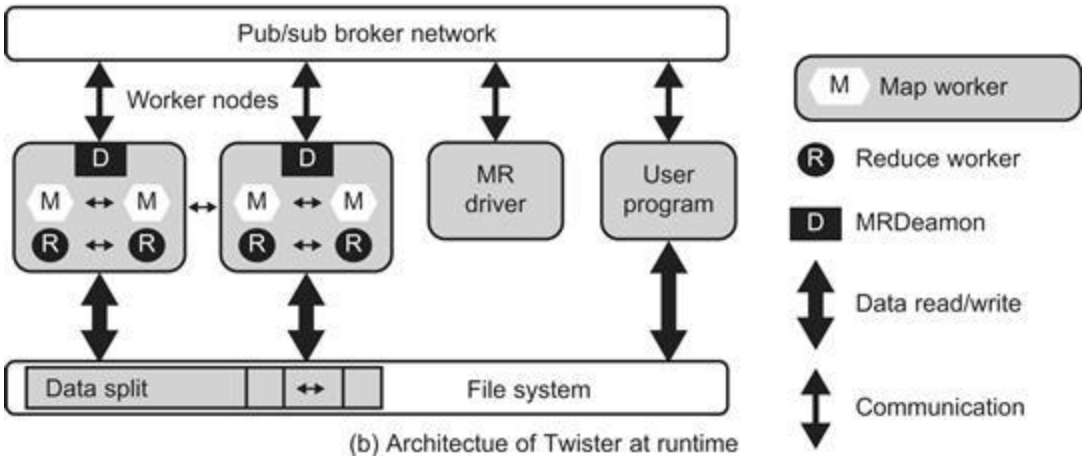
These changes will lead to major performance increases at the cost of poorer fault tolerance and ease to support dynamic changes such as the number of available nodes.

This concept has been investigated in several projects while the direct idea of using MPI for MapReduce applications is investigated . Twister distinguishes the static data which is never reloaded from the dynamic δ flow that is communicated.

The Map-Reduce pair is iteratively executed in long-running threads. the different thread and process structures of 4 parallel programming paradigms: namely Hadoop, Dryad, Twister (also called MapReduce++), and MPI.



(a) Twister for iterative MapReduce programming



(b) Architectue of Twister at runtime

68)What is Hadoop? What are the two fundamental layers in Hadoop? Explain the architecture of MapReduce in Hadoop.

Hadoop is an open source implementation of MapReduce coded and released in Java (rather than C) by Apache. The Hadoop implementation of MapReduce uses the *Hadoop Distributed File System (HDFS)* as its underlying layer rather than GFS. The Hadoop core is divided into two fundamental layers: the MapReduce engine and HDFS.

The MapReduce engine is the computation engine running on top of HDFS as its data storage manager. The control flow of HDFS operations such as write and read can properly highlight roles of the NameNode and DataNodes in the managing operations. In this section, the control flow of the main operations of HDFS on files is further described to manifest the interaction between the user, the NameNode, and the DataNodes in such systems.

The following two sections cover the details of these two fundamental layers.

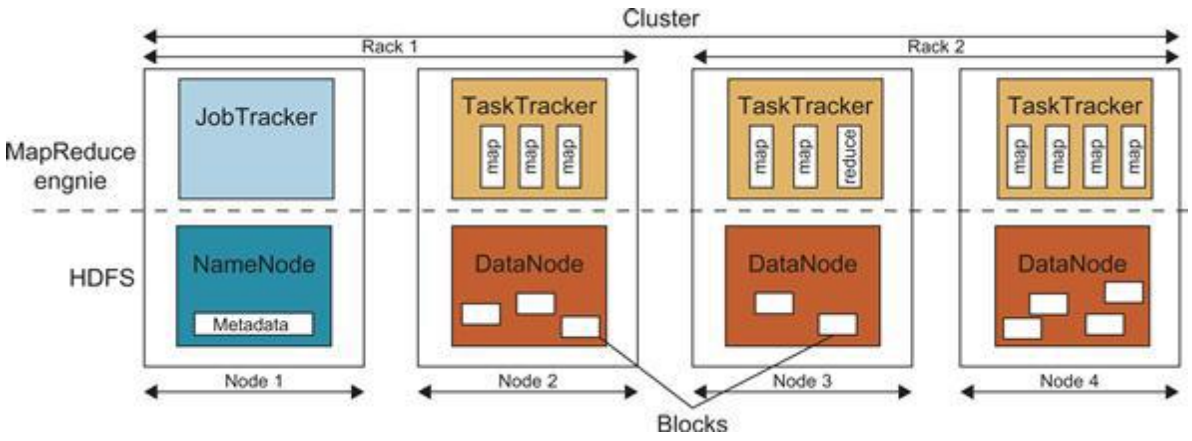
Architecture of MapReduce in Hadoop

The topmost layer of Hadoop is the MapReduce engine that manages the data flow and control flow of MapReduce jobs over distributed computing systems. Similar to HDFS, the MapReduce engine also has a master/slave architecture consisting of a single JobTracker as the master and a number of TaskTrackers as the slaves (workers). The JobTracker manages the MapReduce job over a cluster and is responsible for monitoring jobs and assigning tasks to TaskTrackers.

The TaskTracker manages the execution of the map and/or reduce tasks on a single computation node in the cluster. HDFS and MapReduce architecture in Hadoop where boxes with different shadings refer to different functional nodes applied to different blocks of data.

Each TaskTracker node has a number of simultaneous execution slots, each executing either a map or a reduce task. Slots are defined as the number of simultaneous threads supported by CPUs of the TaskTracker node.

For example, a TaskTracker node with *N* CPUs, each supporting *M* threads, has *M* * *N* simultaneous execution slots .It is worth noting that each data block is processed by one map task running on a single slot. Therefore, there is a one-to-one correspondence between map tasks in a TaskTracker and data blocks in the respective DataNode.



Running a Job in Hadoop

Three components contribute in running a job in this system: a user node, a JobTracker, and several TaskTrackers. The data flow starts by calling the runJob(conf) function inside a user program running on the user node, in which conf is an object containing some tuning parameters for the MapReduce framework and HDFS. The runJob(conf) function and conf are comparable to the MapReduce(Spec, &Results) function and Spec in the first implementation of MapReduce by Google.

• **Job Submission** Each job is submitted from a user node to the JobTracker node.
• **Task assignment** The JobTracker creates one map task for each computed input split by the user node and assigns the map tasks to the execution slots of the TaskTrackers.
• **Task execution** The control flow to execute a task (either map or reduce) starts inside the TaskTracker by copying the job JAR file to its file system.
• **Task running check** A task running check is performed by receiving periodic heartbeat messages to the JobTracker from the TaskTrackers.

69)Explain the steps in running a MapReduce jon in Hadoop

Running a Job in Hadoop

Three components contribute in running a job in this system: a user node, a JobTracker, and several TaskTrackers. The data flow starts by calling the runJob(conf) function inside a user program running on the user node, in which conf is an object containing some tuning parameters for the MapReduce framework and HDFS.

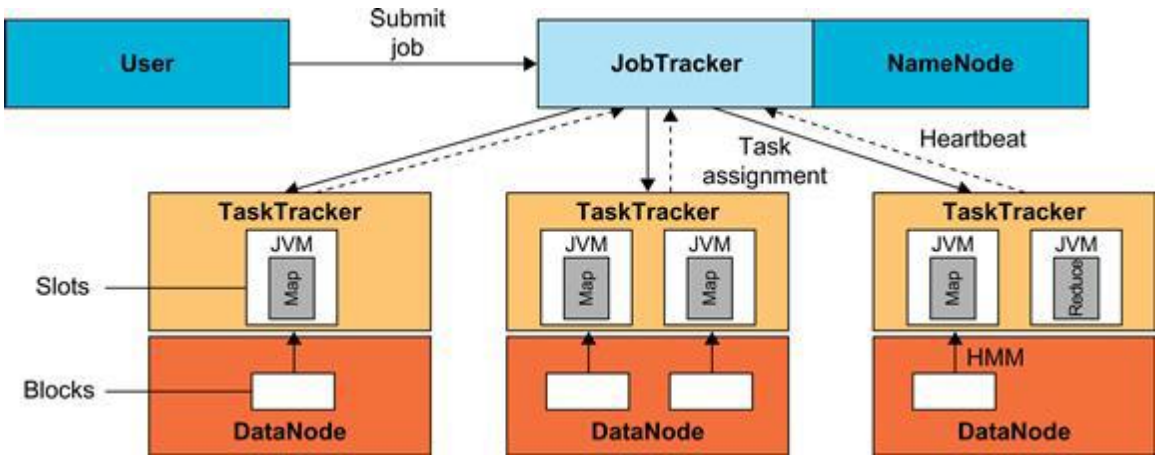
The runJob(conf) function and conf are comparable to the MapReduce(Spec, &Results) function and Spec in the first implementation of MapReduce by Google.

• **Job Submission** Each job is submitted from a user node to the JobTracker node that might be situated in a different node within the cluster through the following procedure:
• A user node asks for a new job ID from the JobTracker and computes input file splits.
• The user node copies some resources, such as the job's JAR file, configuration file, and computed input splits, to the JobTracker's file system.
• The user node submits the job to the JobTracker by calling the submitJob() function.

• **Task assignment** The JobTracker creates one map task for each computed input split by the user node and assigns the map tasks to the execution slots of the TaskTrackers. The JobTracker considers the localization of the data when assigning the map tasks to the TaskTrackers. The JobTracker also creates reduce tasks and assigns them to the TaskTrackers. The number of reduce tasks is predetermined by the user, and there is no locality consideration in assigning them.

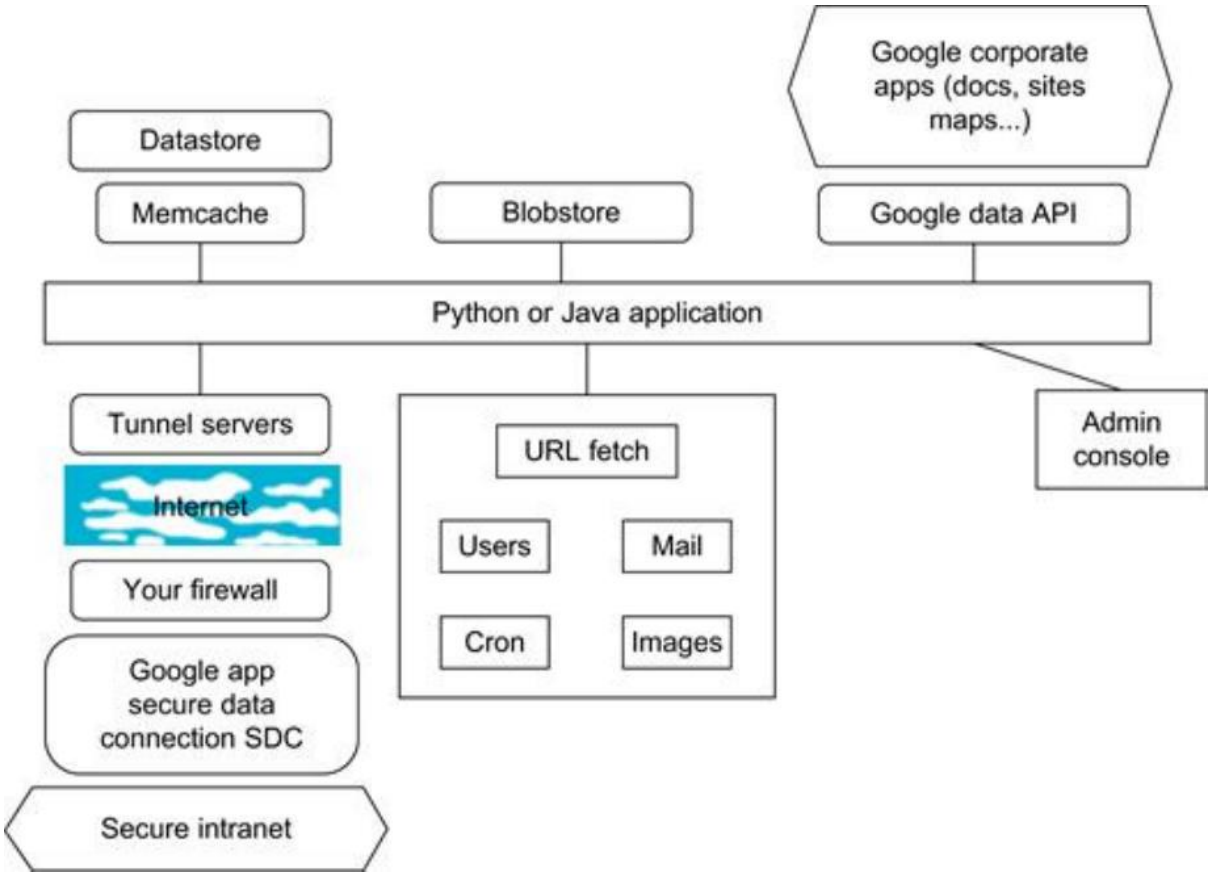
• **Task execution** The control flow to execute a task (either map or reduce) starts inside the TaskTracker by copying the job JAR file to its file system. Instructions inside the job JAR file are executed after launching a Java Virtual Machine (JVM) to run its map or reduce task.

• **Task running check** A task running check is performed by receiving periodic heartbeat messages to the JobTracker from the TaskTrackers. Each heartbeat notifies the JobTracker that the sending TaskTracker is alive, and whether the sending TaskTracker is ready to run a new task.



UNIT 4

70)Explain the programming environment for Google App Engine.



GAE programming model for two supported languages: Java and Python. A client environment that includes an Eclipse plug-in for Java allows you to debug your GAE on your local machine. Also, the GWT Google Web Toolkit is available for Java web application developers.

Developers can use this, or any other language using a JVM-based interpreter or compiler, such as JavaScript or Ruby. Python is often used with frameworks such as Django and CherryPy, but Google also supplies a built in webapp Python environment.

The data store is a NOSQL data management system for entities that can be, at most, 1 MB in size and are labeled by a set of schema-less properties.

The performance of the data store can be enhanced by in-memory caching using the memcache, which can also be used independently of the data store.

Google added the blobstore which is suitable for large files as its size limit is 2 GB.

There are several mechanisms for incorporating external resources. The Google SDC Secure Data Connection can tunnel through the Internet and link your intranet to an external GAE application.

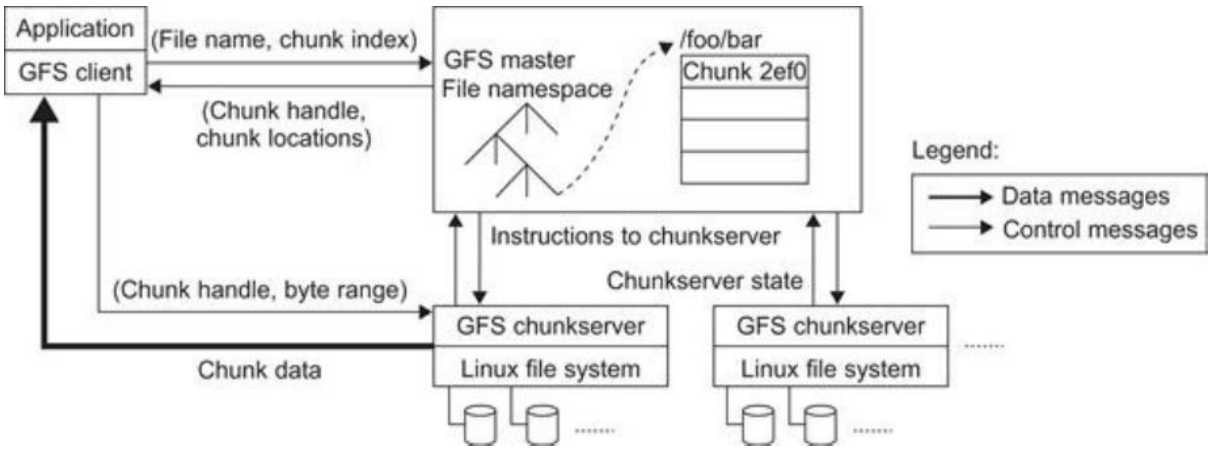
The URL Fetch operation provides the ability for applications to fetch resources and communicate with other hosts over the Internet using HTTP and HTTPS requests.

There is a specialized mail mechanism to send e-mail from your GAE application.

An application can use Google Accounts for user authentication. Google Accounts handles user account creation and sign- in, and a user that already has a Google account (such as a Gmail account) can use that account with your app.

cron jobs are handled by the Cron service.

71)With the help of a neat diagram, explain Google File System.



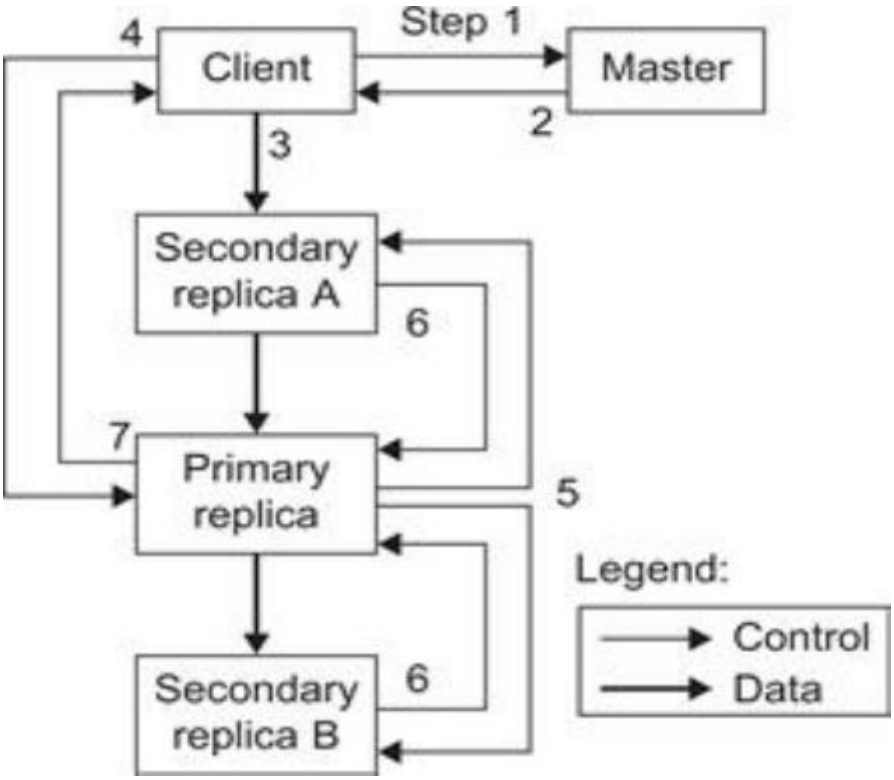
GFS was built primarily as the fundamental storage service for Google's search engine. As the size of the web data that was crawled and saved was quite substantial, Google needed a distributed file system to redundantly store massive amounts of data on cheap and unreliable computers.

None of the traditional distributed file systems can provide such functions and hold such large amounts of data. In addition, GFS was designed for Google applications, and Google applications were built for GFS.

It is quite obvious that there is a single master in the whole cluster. Other nodes act as the chunk servers for storing data, while the single master stores the metadata. The file system namespace and locking facilities are managed by the master. The master periodically communicates with the chunk servers to collect management information as well as give instructions to the chunk servers to do work such as load balancing or fail recovery.

The master has enough information to keep the whole cluster in a healthy state. With a single master, many complicated distributed algorithms can be avoided and the design of the system can be simplified. However, this design does have a potential weakness, as the single GFS master could be the performance bottleneck and the single point of failure. To mitigate this, Google uses a shadow master to replicate all the data on the master, and the design guarantees that all the data operations are performed directly between the client and the chunk server. The control messages are transferred between the master and the clients and they can be cached for future use. With the current quality of commodity servers, the single master can handle a cluster of more than 1,000 nodes.

72)Explain the data mutation sequence in Google File System.



1. The client asks the master which chunk server holds the current lease for the chunk and the locations of the other replicas. If no one has a lease, the master grants one to a replica it chooses (not shown).
2. The master replies with the identity of the primary and the locations of the other (secondary) replicas. The client caches this data for future mutations. It needs to contact the master again only when the primary becomes unreachable or replies that it no longer holds a lease.
3. The client pushes the data to all the replicas. A client can do so in any order. Each chunk server will store the data in an internal LRU buffer cache until the data is used or aged out. By decoupling the data flow from the control flow, we can improve performance by scheduling the expensive data flow based on the network topology regardless of which chunk server is the primary.
4. Once all the replicas have acknowledged receiving the data, the client sends a write request to the primary. The request identifies the data pushed earlier to all the replicas. The primary assigns consecutive serial numbers to all the mutations it receives, possibly from multiple clients, which provides the necessary serialization. It applies the mutation to its own local state in serial order.
5. The primary forwards the write request to all secondary replicas. Each secondary replica applies mutations in the same serial number order assigned by the primary.
6. The secondaries all reply to the primary indicating that they have completed the operation.

7. The primary replies to the client. Any errors encountered at any replicas are reported to the client. In case of errors, the write corrects at the primary and an arbitrary subset of the secondary replicas. The client request is considered to have failed, and the modified region is left in an inconsistent state. Our client code handles such errors by retrying the failed mutation. It will make a few attempts at steps 3 through 7 before falling back to a retry from the beginning of the write.

73)Explain Google's NOSQL system. (or Explain BigData)

BigTable was designed to provide a service for storing and retrieving structured and semistructured data.

BigTable applications include storage of web pages, per-user data, and geographic locations.

Here we use web pages to represent URLs and their associated data, such as contents, crawled metadata, links, anchors, and page rank values.

Per-user data has information for a specific user and includes such data as user preference settings, recent queries/search results, and the user's e-mails.

Geographic locations are used in Google's well-known Google Earth software.

Geographic locations include physical entities (shops, restaurants, etc.), roads, satellite image data, and user annotations.

The scale of such data is incredibly large. There will be billions of URLs, and each URL can have many versions, with an average page size of about 20 KB per version. The user scale is also huge. There are hundreds of millions of users and there will be thousands of queries per second. The same scale occurs in the geographic data, which might consume more than 100 TB of disk space.

The BigTable system is built on top of an existing Google cloud infrastructure. BigTable uses the following building blocks:

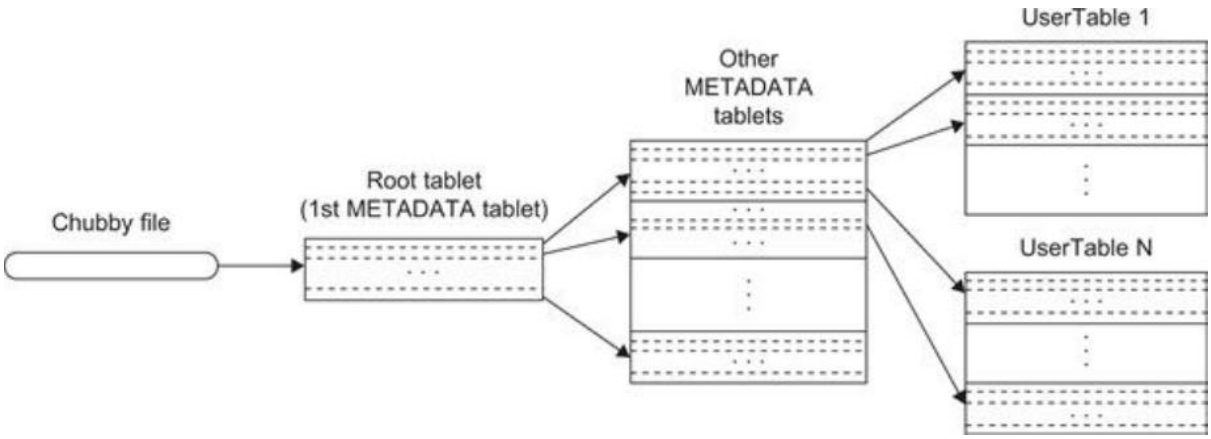
1. GFS: stores persistent state

2. Scheduler: schedules jobs involved in BigTable serving

3. Lock service: master election, location bootstrapping

4. MapReduce: often used to read/write BigTable data

74)Explain the tablet location hierarchy in using the BigTable.



Lets see how to locate the BigTable data starting from the file stored in Chubby.

The first level is a file stored in Chubby that contains the location of the root tablet. The root tablet contains the location of all tablets in a special METADATA table. Each METADATA tablet contains the location of a set of user tablets.

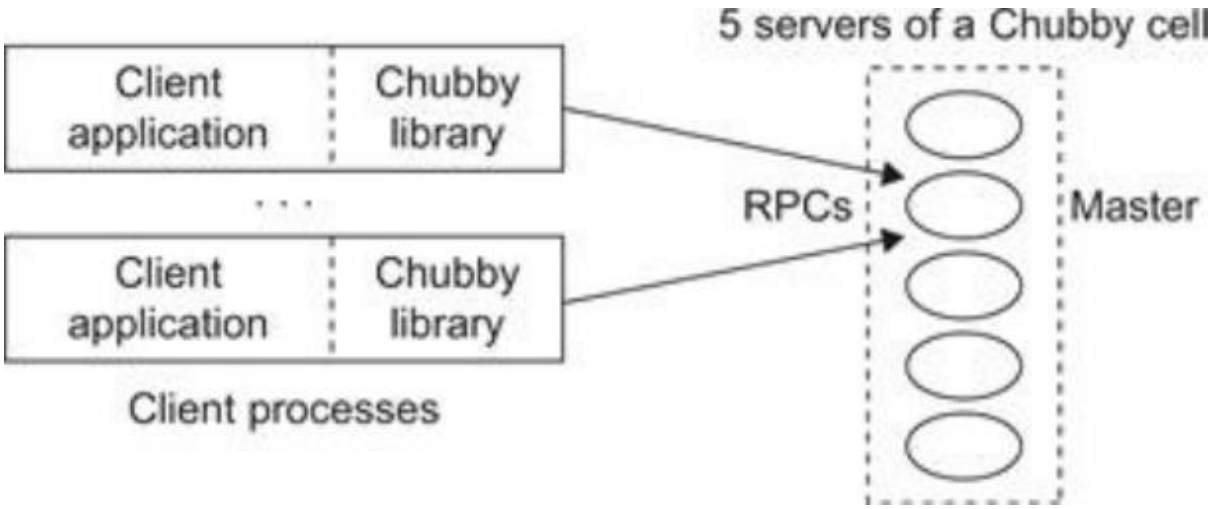
The root tablet is just the first tablet in the METADATA table, but is treated specially; it is never split to ensure that the tablet location hierarchy has no more than three levels.

The METADATA table stores the location of a tablet under a row key that is an encoding of the tablet's table identifier and its end row.

BigTable includes many optimizations and fault-tolerant features. Chubby can guarantee the availability of the file for finding the root tablet. The BigTable master can quickly scan the tablet servers to determine the status of all nodes.

Tablet servers use compaction to store data efficiently. Shared logs are used for logging the operations of multiple tablets so as to reduce the log space as well as keep the system consistent.

75)With the help of a neat diagram explain the structure of Google Chubby for distributed file system.



Chubby is intended to provide a coarse-grained locking service. It can store small files inside Chubby storage which provides a simple namespace as a file system tree.

The files stored in Chubby are quite small compared to the huge files in GFS. Based on the Paxos agreement protocol, the Chubby system can be quite reliable despite the failure of any member node.

Each Chubby cell has five servers inside. Each server in the cell has the same file system namespace. Clients use the Chubby library to talk to the servers in the cell.

Client applications can perform various file operations on any server in the Chubby cell.

Servers run the Paxos protocol to make the whole file system reliable and consistent. Chubby has become Google's primary internal name service. GFS and BigTable use Chubby to elect a primary from redundant replicas.

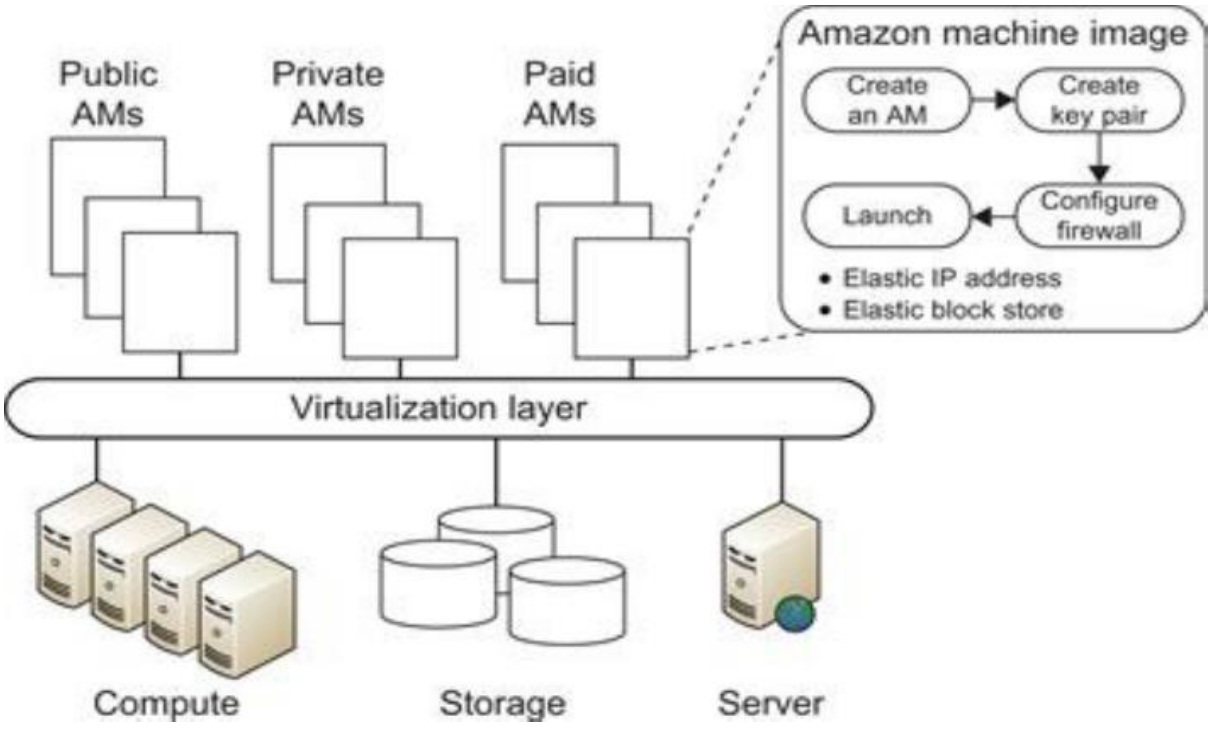
76)Explain the different types of Amazon Machine Images. Explain the execution environment of Amazon Elastic Compute Cloud.

Three Types of AMI

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

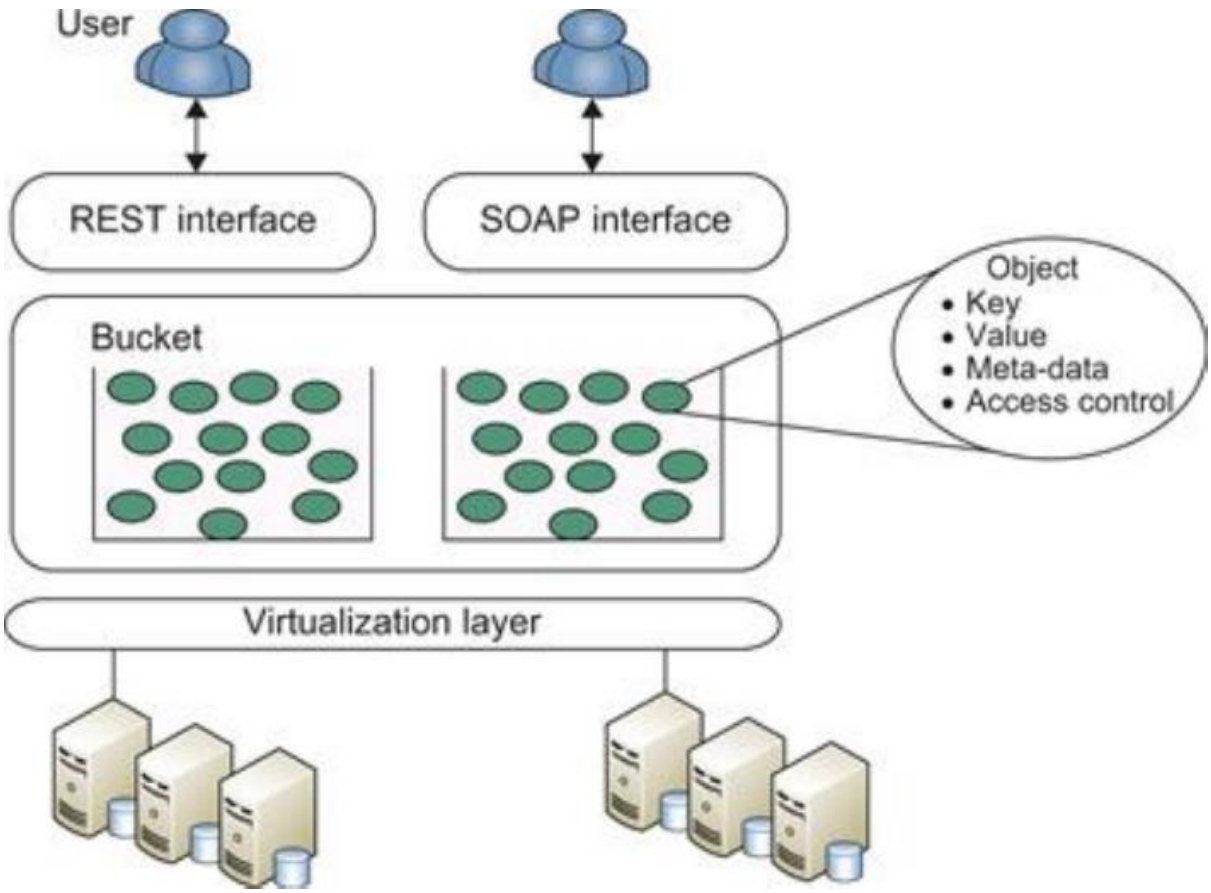
Customers can rent VMs instead of physical machines to run their own applications. By using VMs, customers can load any software of their choice. The elastic feature of such a service is that a customer can create, launch, and terminate server instances as needed, paying by the hour for active servers. Amazon provides several types of preinstalled VMs.

Instances are often called Amazon Machine Images (AMIs) which are preconfigured with operating systems based on Linux or Windows, and additional software.



Create an AMI → Create Key Pair → Configure Firewall → Launch

77)Explain the execution environment of Amazon Simple Storage Service.



Amazon S3 provides a simple web services interface that can be used to store and retrieve any amount of data, at any time, from anywhere on the web.

S3 provides the object-oriented storage service for users. Users can access their objects through Simple Object Access Protocol (SOAP) with either browsers or other client programs which support SOAP.

The fundamental operation unit of S3 is called an object. Each object is stored in a bucket and retrieved via a unique, developer-assigned key. In other words, the bucket is the container of the object. Besides unique key attributes, the object has other attributes such as values, metadata, and access control information.

From the programmer's perspective, the storage provided by S3 can be viewed as a very coarse-grained key-value pair. Through the key-value programming interface, users can write, read, and delete objects containing from 1 byte to 5 gigabytes of data each.

There are two types of web service interface for the user to access the data stored in Amazon clouds. One is a REST (web 2.0) interface, and the other is a SOAP interface.

Here are some key features of S3:

• Redundant through geographic dispersion.

• Designed to provide 99.999999999 percent durability and 99.99 percent availability of objects over a given year with cheaper reduced redundancy storage (RRS).

• Authentication mechanisms to ensure that data is kept secure from unauthorized access. Objects can be made private or public, and rights can be granted to specific users.

• Per-object URLs and ACLs (access control lists).

• Default download protocol of HTTP. A BitTorrent protocol interface is provided to lower costs for high-scale distribution.

• $0.055 (more than 5,000 TB) to 0.15 per GB per month storage (depending on total amount).

• First 1 GB per month input or output free and then $.08 to $0.15 per GB for transfers outside an S3 region.

• There is no data transfer charge for data transferred between Amazon EC2 and Amazon S3 within the same region.

78)Explain the following:

i) **Amazon Elastic Block Store.**
ii) **Amazon Simple DB service.**

The *Elastic Block Store (EBS)* provides the volume block interface for saving and restoring the virtual images of EC2 instances. Traditional EC2 instances will be destroyed after use.

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

Note that S3 is “Storage as a Service” with a messaging interface. EBS is analogous to a distributed file system accessed by traditional OS disk access mechanisms.

EBS allows you to create storage volumes from 1 GB to 1 TB that can be mounted as EC2 instances.

Multiple volumes can be mounted to the same instance. These storage volumes behave like raw, unformatted block devices, with user-supplied device names and a block device interface. You can create a file system on top of Amazon EBS volumes, or use them in any other way you would use a block device (like a hard drive). Snapshots are provided so that the data can be saved incrementally. This can improve performance when saving and restoring data. In terms of pricing, Amazon provides a similar pay-per-use schema as EC2 and S3.

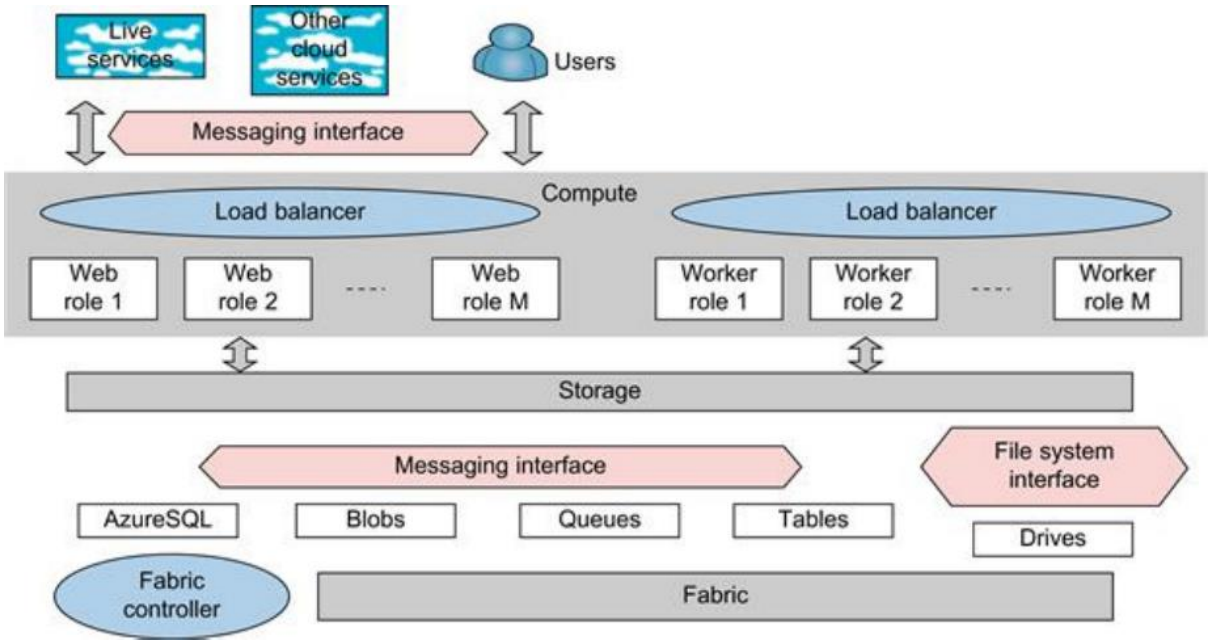
SimpleDB provides a simplified data model based on the relational database data model. Structured data from users must be organized into domains.

Each domain can be considered a table. The items are the rows in the table. A cell in the table is recognized as the value for a specific attribute (column name) of the corresponding row.

This is similar to a table in a relational database. However, it is possible to assign multiple values to a single cell in the table. This is not permitted in a traditional relational database which wants to maintain data consistency.

Many developers simply want to quickly store, access, and query the stored data. SimpleDB removes the requirement to maintain database schemas with strong consistency.

79)With the help of a diagram explain the features of programming Azure cloud platform.



First we have the underlying Azure fabric consisting of virtualized hardware together with a sophisticated control environment implementing dynamic assignment of resources and fault tolerance. This implements domain name system (DNS) and monitoring capabilities. Automated service management allows service models to be defined by an XML template and multiple service copies to be instantiated on request.

When the system is running, services are monitored and one can access event logs, trace/debug data, performance counters, IIS web server logs, crash dumps, and other log files. This information

can be saved in Azure storage. Note that there is no debugging capability for running cloud applications, but debugging is done from a trace. One can divide the basic features into storage and compute capabilities. The Azure application is linked to the Internet through a customized compute VM called a web role supporting basic Microsoft web hosting. Such configured VMs are often called appliances. The other important compute class is the worker role reflecting the importance in cloud computing of a pool of compute resources that are scheduled as needed. The roles support HTTP(S) and TCP. Roles offer the following methods:

• The OnStart() method which is called by the Fabric on startup, and allows you to perform initialization tasks. It reports a Busy status to the load balancer until you return true.

• The OnStop() method which is called when the role is to be shut down and gives a graceful exit.

• The Run() method which contains the main logic.

80)Explain SQL Azure and Azure tables.

The *SQLAzure* service offers SQL Server as a service.

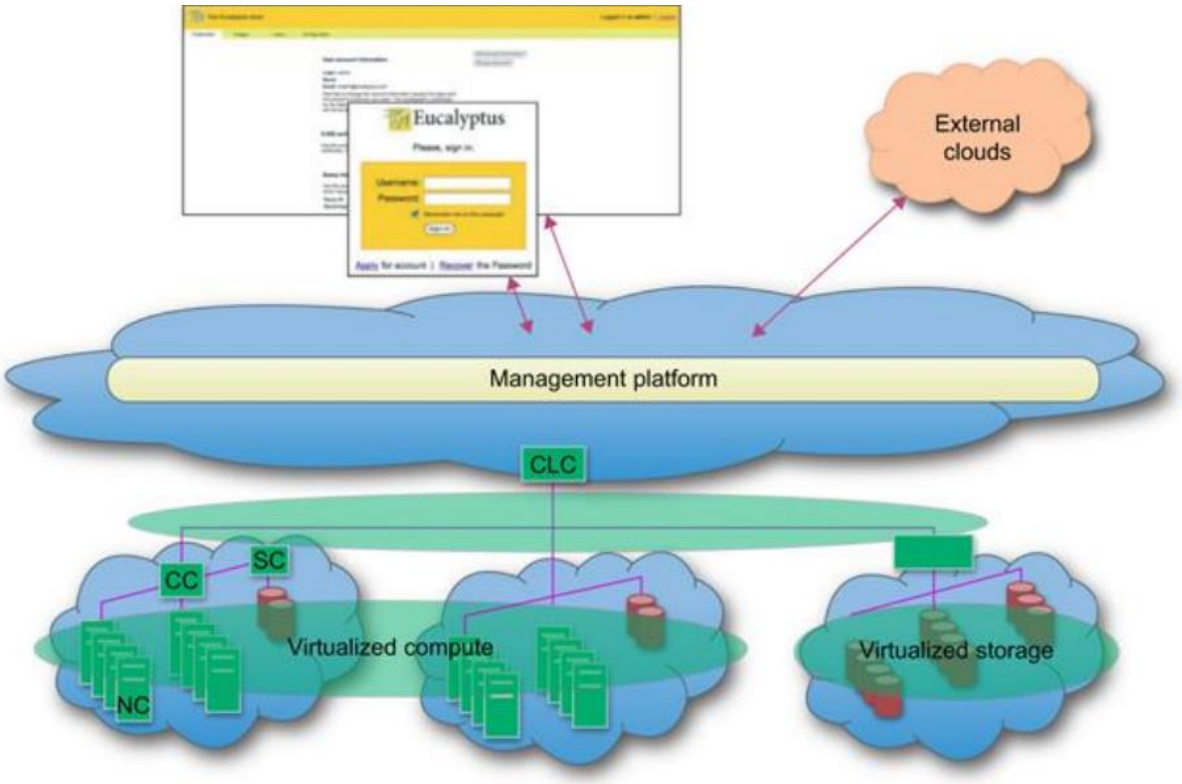
All the storage modalities are accessed with REST interfaces except for the recently introduced Drives that are analogous to Amazon EBS and offer a file system interface as a durable NTFS volume backed by blob storage. The REST interfaces are automatically associated with URLs and all storage is replicated three times for fault tolerance and is guaranteed to be consistent in access.

The basic storage system is built from blobs which are analogous to S3 for Amazon. Blobs are arranged as a three-level hierarchy: Account → Containers → Page or Block Blobs. Containers are analogous to directories in traditional file systems with the account acting as the root. The block blob is used for streaming data and each such blob is made up as a sequence of blocks of up to 4 MB each, while each block has a 64 byte ID. Block blobs can be up to 200 GB in size. Page blobs are for random read/ write access and consist of an array of pages with a maximum blob size of 1 TB. One can associate metadata with blobs as <name, value> pairs with up to 8 KB per blob.

The Azure Table and Queue storage modes are aimed at much smaller data volumes. *Queues* provide reliable message delivery and are naturally used to support work spooling between web and worker roles. Queues consist of an unlimited number of messages which can be retrieved and processed at least once with an 8 KB limit on message size. Azure supports PUT, GET, and DELETE message operations as well as CREATE and DELETE for queues. Each account can have any number of Azure *tables* which consist of rows called *entities* and columns called *properties*.

There is no limit to the number of entities in a table and the technology is designed to scale well to a large number of entities stored on distributed computers. All entities can have up to 255 general properties which are <name, type, value> triples. Two extra properties, *PartitionKey* and *RowKey*, must be defined for each entity, but otherwise, there are no constraints on the names of properties—this table is very flexible! *RowKey* is designed to give each entity a unique label while *PartitionKey* is designed to be shared and entities with the same *PartitionKey* are stored next to each other; a good use of *PartitionKey* can speed up search performance. An entity can have, at most, 1 MB storage; if you need large value sizes, just store a link to a blob store in the *Table* property value. ADO.NET and LINQ support table queries.

81)With the help of a neat diagram explain the architecture of Eucalyptus for virtual machine image management.



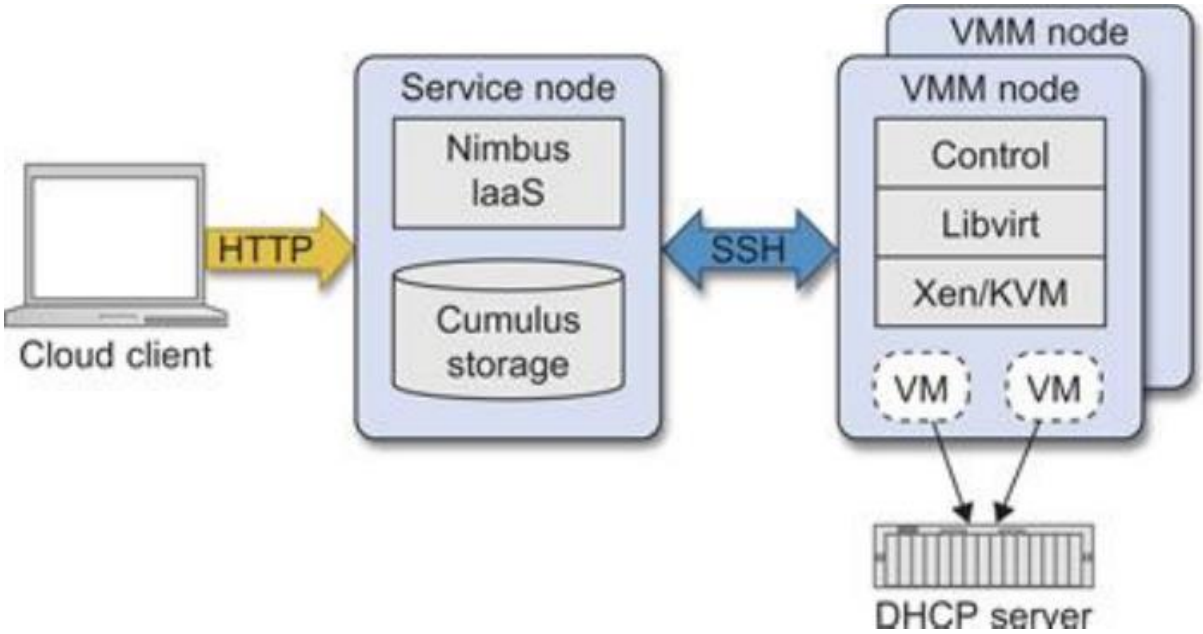
Eucalyptus takes many design queues from Amazon's EC2, and its image management system is no different.

Eucalyptus stores images in Walrus, the block storage system that is analogous to the Amazon S3 service. As such, any user can bundle her own root file system, and upload and then register this image and link it with a particular kernel and ramdisk image.

This image is uploaded into a user-defined bucket within Walrus, and can be retrieved anytime from any availability zone. This allows users to create specialty virtual and deploy them within Eucalyptus with ease.

The Eucalyptus system is available in a commercial proprietary version, as well as the open source version.

82)Explain the Nimbus cloud infrastructure.



Nimbus is a set of open source tools that together provide an IaaS cloud computing solution.

Here a storage cloud implementation called Cumulus has been tightly integrated with the other central services, although it can also be used stand-alone.

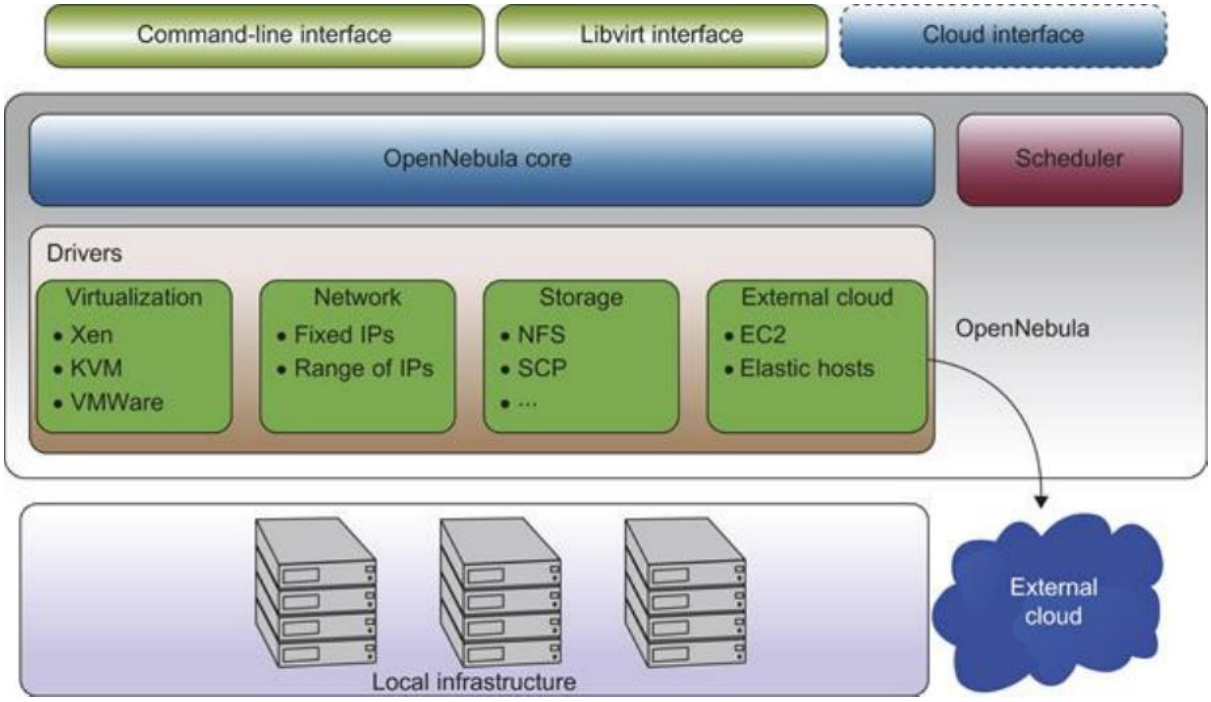
Nimbus supports two resource management strategies.

The first is the default “resource pool” mode. In this mode, the service has direct control of a pool of VM manager nodes and it assumes it can start VMs.

The other supported mode is called “pilot.” Here, the service makes requests to a cluster’s Local Resource Management System (LRMS) to get a VM manager available to deploy VMs.

Nimbus also provides an implementation of Amazon’s EC2 interface that allows users to use clients developed for the real EC2 system against Nimbus-based clouds.

83)What are the main components of OpenNebula architecture? Explain.

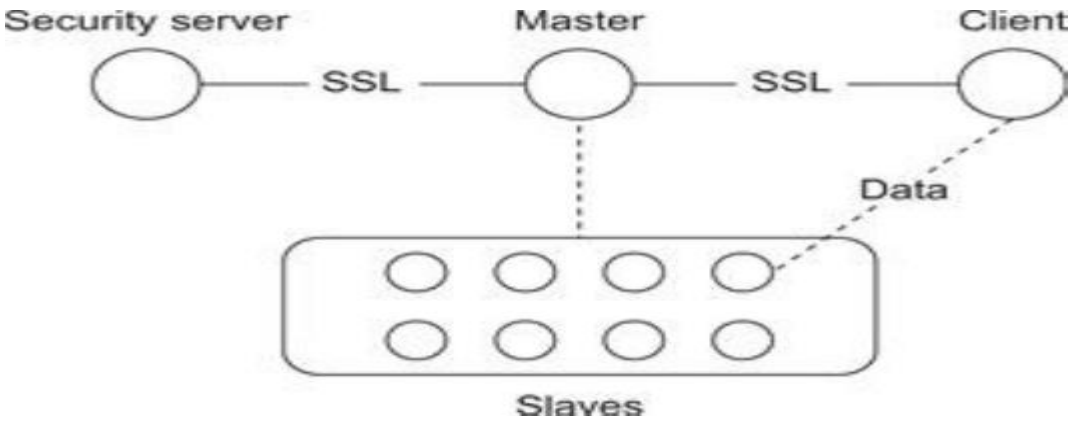


The core is a centralized component that manages the VM full life cycle, including setting up networks dynamically for groups of VMs and managing their storage requirements, such as VM disk image deployment or on-the-fly software environment creation.

Another important component is the capacity manager or scheduler. It governs the functionality provided by the core. The default capacity scheduler is a requirement/rank matchmaker. However, it is also possible to develop more complex scheduling policies, through a lease model and advance reservations.

The last main components are the access drivers. They provide an abstraction of the underlying infrastructure to expose the basic functionality of the monitoring, storage, and virtualization services available in the cluster. Therefore, OpenNebula is not tied to any specific environment and can provide a uniform management layer regardless of the virtualization platform.

84)What is Sector/Sphere? Explain its architecture.



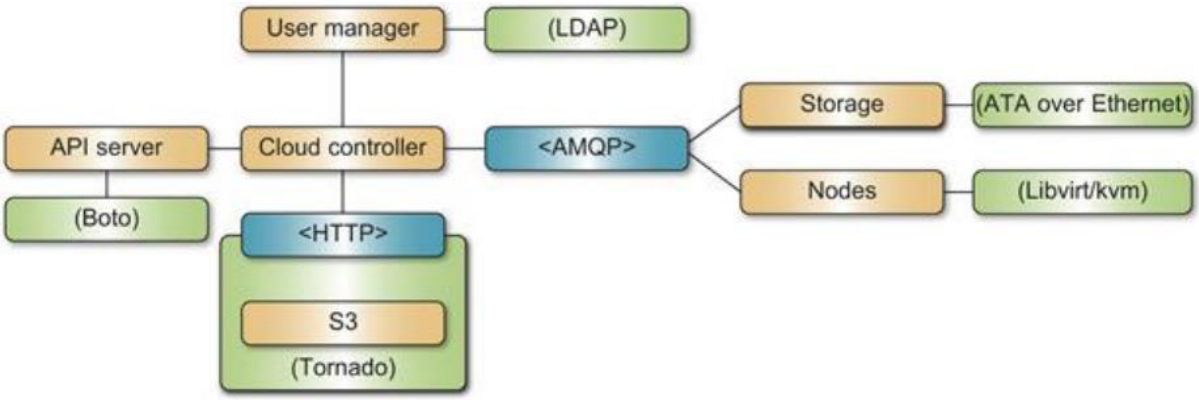
Sector/Sphere is a software platform that supports very large distributed data storage and simplified distributed data proscessing over large clusters of commodity computers, either within a data center or across multiple data centers.

Sector is a distributed file system (DFS) that can be deployed over a wide area and allows users to manage large data sets from any location with a high-speed network connection.

Sphere is a parallel data processing engine designed to work with data managed by Sector. This coupling allows the system to make accurate decisions about job scheduling and data location. Sphere provides a programming framework that developers can use to process data stored in Sector.

85)What is OpenStack? Explain OpenStack Nova architecture and OpenStack Storage.

OpenStack was been introduced by Rackspace and NASA in July 2010. The project is building an open source community spanning technologists, developers, researchers, and industry to share resources and technologies with the goal of creating a massively scalable and secure cloud infrastructure.

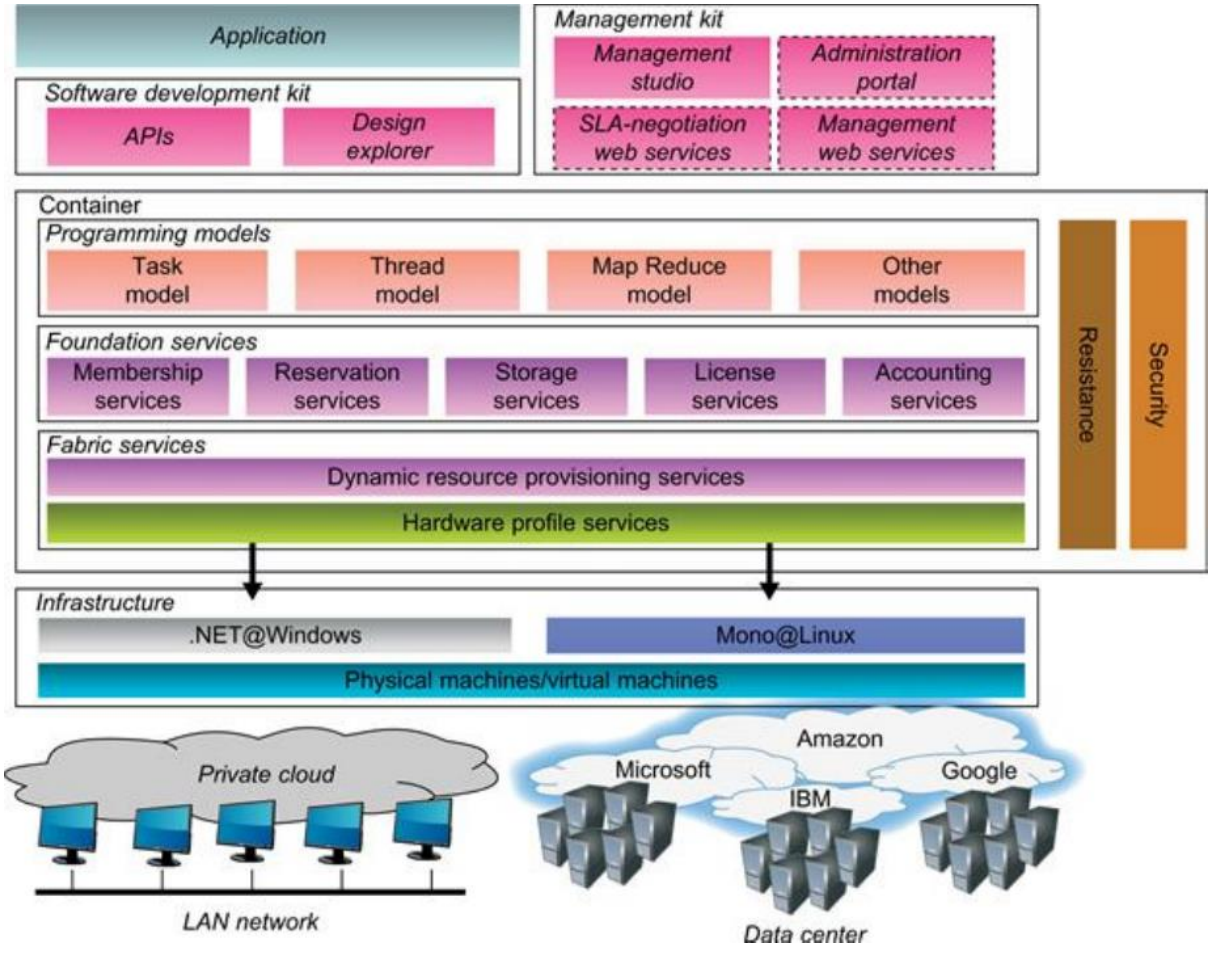


In above architecture, the API Server receives HTTP requests from boto, converts the commands to and from the API format, and forwards the requests to the cloud controller.

The cloud controller maintains the global state of the system, ensures authorization while interacting with the User Manager via Lightweight Directory Access Protocol (LDAP), interacts with the S3 service, and manages nodes, as well as storage workers through a queue. Additionally, Nova integrates networking components to manage private networks, public IP ad- dressing, virtual private network (VPN) connectivity, and firewall rules.

The OpenStack storage solution is built around a number of interacting components and concepts, including a proxy server, a ring, an object server, a container server, an account server, replication, updaters, and auditors. The role of the proxy server is to enable lookups to the accounts, containers, or objects in OpenStack storage rings and route the requests.

86)With the help of a neat diagram, explain the components and architecture of Aneka.



Aneka (www.manjrasoft.com/) is a cloud application platform developed by Manjrasoft, based in Melbourne, Australia. It is designed to support rapid development and deployment of parallel and distributed applications on private or public clouds.

It provides a rich set of APIs for transparently exploiting distributed resources and expressing the business logic of applications by using preferred programming abstractions.

System administrators can leverage a collection of tools to monitor and control the deployed infrastructure.

It can be deployed on a public cloud such as Amazon EC2 accessible through the Internet to its subscribers, or a private cloud constituted by a set of nodes with restricted access

87)What are the advantages of Aneka over other distributed workload solutions? What are the three types of capabilities offered by Aneka to build, accelerate and manage clouds and its applications? What are the important programming models supported by Aneka for cloud and parallel applications?

Aneka acts as a workload distribution and management platform for accelerating applications in both Linux and Microsoft .NET framework environments. Some of the key advantages of Aneka over other workload distribution solutions include:

- Support of multiple programming and application environments
- Simultaneous support of multiple runtime environments
- Rapid deployment tools and framework
- Ability to harness multiple virtual and/or physical machines for accelerating application provisioning based on users' Quality of Service/service-level agreement (QoS/SLA) requirements
- Built on top of the Microsoft .NET framework, with support for Linux environments through Mono

Aneka offers three types of capabilities which are essential for building, accelerating, and managing clouds and their applications:

1. **Build** Aneka includes a new SDK which combines APIs and tools to enable users to rapidly develop applications. Aneka also allows users to build different runtime environments such as enterprise/private cloud by harnessing compute resources in network or enterprise data centers, Amazon EC2, and hybrid clouds by combining enterprise private clouds managed by Aneka with resources from Amazon EC2 or other enterprise clouds built and managed using XenServer.
2. **Accelerate** Aneka supports rapid development and deployment of applications in multiple runtime environments running different operating systems such as Windows or Linux/UNIX. Aneka uses physical machines as much as possible to achieve maximum utilization in local environments. Whenever users set QoS parameters such as deadlines, and if the enterprise resources are insufficient to meet the deadline, Aneka supports dynamic leasing of extra capabilities from public clouds such as EC2 to complete the task within the deadline.
3. **Manage** Management tools and capabilities supported by Aneka include a GUI and APIs to set up, monitor, manage, and maintain remote and global Aneka compute clouds. Aneka also has an accounting mechanism and manages priorities and scalability based on SLA/QoS which enables dynamic provisioning.

Here are three important programming models supported by Aneka for both cloud and traditional parallel applications:

1. Thread programming model, best solution to adopt for leveraging the computing capabilities of multicore nodes in a cloud of computers

2. Task programming model, which allows for quickly prototyping and implementing an independent bag of task applications

3. MapReduce programming model

UNIT 5

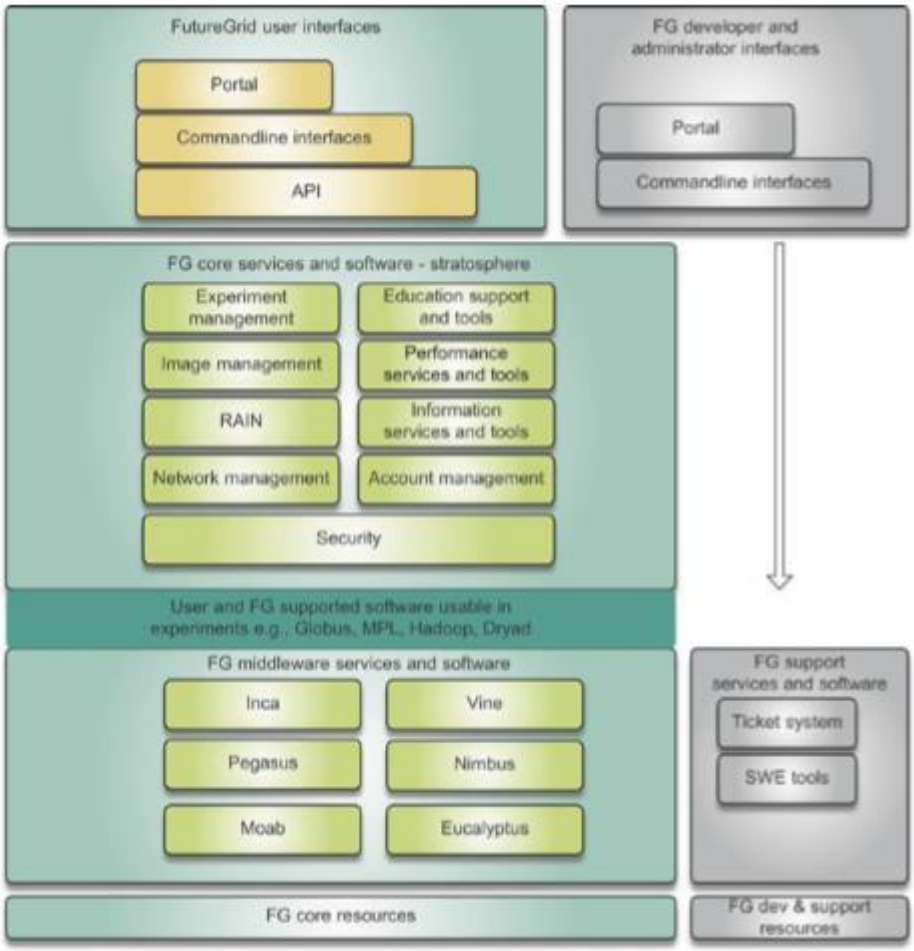
88)With the help of a neat diagram, explain the FutureGrid Software grid.

ANS : .FutureGrid is not a production system, but rather an environment supporting a flexible development and testing platform for middleware and application users looking at interoperability, functionality, and performance issues.

FutureGrid will make it possible for re- searchers to conduct experiments by submitting an experiment plan that is then executed via a sophisticated workflow engine, preserving the provenance and state information necessary to allow reproducibility.

One of the goals of FutureGrid is to understand the behaviour and utility of cloud computing approaches.

Following figure shows the software stack developed in the FutureGrid project



.FutureGrid offers seven distinct systems (clusters) of about 5,000 cores with different optimizations. It also has a dedicated network (except to the Alamo system

in Texas) allowing security isolation and use of a dedicated network impairment device.

Clouds provide great flexibility by building environments on top of hypervisors .

FutureGrid follows a different route by dynamically provisioning software as needed onto “bare metal.”

Above figure shows the software stack developed in the FutureGrid pro- ject.

The images need careful security review and currently include choices at different hypervisor levels (Xen, KVM), op- erating systems (Linux, Windows), cloud environments (Nimbus, Eucalyptus, OpenNebula), middleware (gLite, Unicore, Globus), programming paradigms (MPI, OpenMP, Hadoop, Dryad), and specialized capability such as distributed shared memory from ScaleMP.

FutureGrid expects available image sto increase as users build new and interesting environments and deposit them in the library. Figure 9.9 shows the FutureGrid software stack. Key components include portal interfaces; monitoring capability (INCA, Power [greenIT]); the Experiment Manager, a new feature allowing reproducibility of execution scenarios; image generation and repository feature; VINE intercloud networking (virtualclusters built with virtual networks);a performance library; RAIN or Runtime Adaptable InsertioN Service which schedules and deploys images; and security features including use of isolated network as well as authentication and authorization.

89)Compare the programmer’s perspective of Data-intensive scalable computing and conventional supper computer.

ANS:

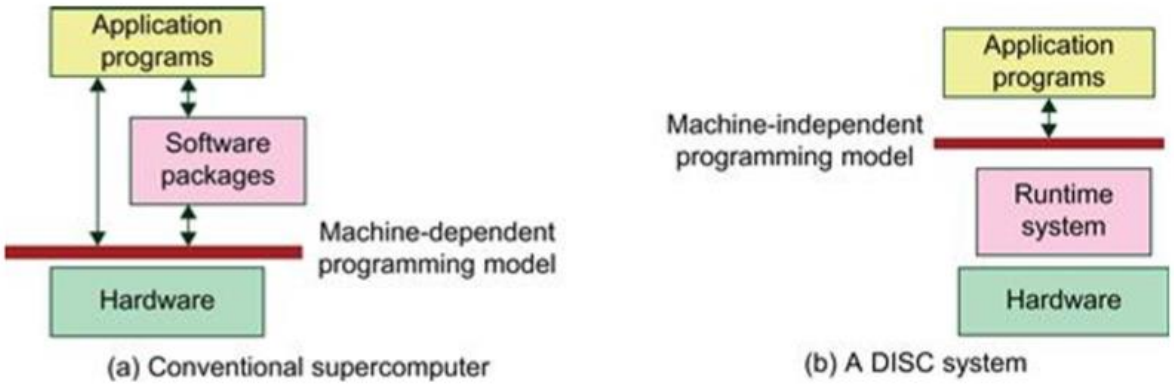


FIGURE 9.10 Programmer’s perspectives of DISC systems against conventional supercomputers. Courtesy of Randal Bryant [5], Reprin-

90)What id Magellen? What are the research issues addressed by it?

ANS: Magellan is the largest of the research clouds and is situated at two U.S. Department of Energy sites: Argonne [6] and the National Energy Research Scientific Computing Center (NERSC) in California. Each consists of a medium-sized cluster and storage facilities.

TheNERSC system has more than 700 node sand about 6,000cores, while Argonne has more than 500nodes and 4,000 cores.

The systems are addressing a wide variety of research issues, including the following:

•Comparing cloud performance and traditional cluster and supercomputing environments for scientific applications. Of particular interest are applications currently running on local/departmental clusters, or midrange applications that donot require more than 1,000 cores.

•Testing solid state (flash) storage for data-intensive applications. Flash storage offers substantially increased bandwidth and IOPS (I/O operation rate), and decreased latency, especially for read-intensive applications.

•Exploring the portability of applications to cloud systems, and in particular, looking at MapReduce.

•Providing portals on clouds to offer easier access to applications, databases, or automated workflows.
•Testingalternativeresourcingmodels.Forexample,whereavailabilityiscrucial,avirtualprivateclustercouldguarantee access to particular research groups for specific periods of time.

•Understanding which scientific applications and user communities are best suited for cloud computing.
•Understandingthedeploymentandsupportissuesrequiredtobuildlargescienceclouds.Is itcost-effectiveandpractical to operate science clouds? How could commercial clouds be leveraged?

•Determining how existing cloud software meets the needs of science and whether extending or enhancing current cloud software improves utility.

•Determining how well cloud computing supports data-intensive scientific applications.

•Determining the challenges in addressing security for a virtualized cloud environment.

91)Explain the following:

i)Grid ‘5000
ii)Open Cirrus
iii)Open Cloud Testbed
iv)Science Clouds
v)Sky Computing
vi)Venus –C

ANS:

i) Grid 5000:

The Grid’ 5000 distributed system links nine sites in France.
Grid’5000 has more than 5,000 cores spread over 1,500 nodes. It has been used in a variety of computer sci- ence research projects including cloud and grid computing and green IT with both software systems and performance goals.
Grid’5000 aims at providing a highly reconfigurable, controllable, and monitored experimental platform to its users.

ii) Open Cirrus:

This is a broad consortium of some 14 partners exploring cloud computing. The quite powerful testbed consists of morethan 600 nodes at four major sites: KIT (Karlsruhe Institute of Technology), the University of Illinois at Urbana Champaign, HP LabsinPaloAlto, California, and the BigDatacluster at IntelResearchinPittsburgh.
The OpenCirrus testbed is designed to support research into the design, provisioning, and management of services at a global, multidata-center scale. The open nature of the testbed is designed to encourage research into all aspects of service and data-center management.
In addition, OpenCirrus aims to foster a collaborative community around the testbed, providing ways to share tools, lessons, and best practices, and ways to benchmark and compare alternative approaches to service management at data-center scale.

iii) Open Cloud Tested

TheOpen Cloud Consortium (OCC)isamember-driven organization that supports the development of standards for cloud computing and frameworks for interoperating between clouds, develops benchmarks for cloud computing, and supports reference implementations for cloud computing.
The OCC also manages a testbed for cloud computing, the Open Cloud Testbed, and operates cloud computing infrastructure to support scientific research, called Open Science Data Cloud, with an emphasis on data-intensive computing.
As of November 2010, the Open Cloud Testbed includes sites at Chicago (two), California (La Jolla, California), and John Hopkins University, with a total of 250 nodes and 1,000 cores.

iv) Science Cloud:

Science Clouds is an open cloud federation providing compute cycles in the cloud for scientific communities exploring the use of Nimbus. The infrastructure federates sites across Europe and the United States. As of November 2010, it consists of the Nimbus Cloud at the University of Chicago, Stratus at the University of Florida, Wispy at Purdue University, and some machines in the Czech Republic.

v) Sky Computing:

This project is aligned with Science Clouds and ,like the European Reservoir project, is designed to federate multiple clouds into a single resource.

Linking separate clouds together can help scalability (increasing the number of resources), but also improves fault tolerance.

The federation also supports the special case of a hybrid cloud—two clouds, one private and one public. Sky Computing has been developed to extend Nimbus to support multiple distinct clouds.

OnegoodexampleoftheSkyComputingprojectisademonstrationthatshowedthebioinformaticsprogramBLASTcon- trolled by Hadoop running on 1,000 cores in six different sites: three in the United States on FutureGrid (discussed earlier) and three on Grid'5000 [5] in Europe.

This distributed federation was enabled by Nimbus for cloud management, offering VM provisioning and contextualization services, from the University of Florida to enable all-to-all communication among multiple clouds, and Hadoop for parallel fault-tolerant execution of BLAST. Network virtualization is also supported in Reservoir.

vi) Venus-C

Venus-C is a European project exploring the applicability of cloud computing in a reas such as building structure analysis ,3D architecture rendering, prediction of marine species population, risk prediction for wildfires, metagenomics, systembiology, and drug development. The project explores both Azure (Windows platforms) and Eucalyptus-based infrastructure at two European HPC centers: the Royal Institute of Technology (KTH, Sweden) and the Barcelona Supercomputing Center (BSC, Spain). The project will develop interoperability tools between different clouds for various applications.

92)How is the Quality of service calculated in cloud computing?

Ans: the cumulative performance of a cloud platform by a compound metric, called quality of cloud services (QoCS) and denoted by θ.

This metric measures the overall quality of the cloud service, regardless of the particular service models applied.

Weuseafive-dimensional Kiviat graph to plot the aggregated cloud performance or to quantify the QoC Sgraphic- ally.

Each Kiviat graph has five orthogonal dimensions, representing the five performance attributes we introduced earlier.

The Kiviat graph provides a holistic view of the capabilities of the cloud service model.

The shaded area of the Kiviat graph represents the cumulative strength or quality of the cloud service being provided. Specifically, the larger the shaded area Ashaded on the Kiviat graph for a given user application running on the cloud plat- form, the better the QoCS, θ, of the cloud platform being tested for a given cloud application. The scale and range of these five dimensions are specified by the template Kiviat graph in Figure 9.11(a). Let Apentagon be the ideal operating area of the pentagon in the Kiviat graph.

The scale on all dimensions isnormalized tobe1.Inotherwords, themaximum radius of the pentagon is set to be 1 along all dimensions.

All five fractional measures are scaled from0to1.The scale of 0 is located at the center of the perfect pentagon, meaning the lowest values of the five performance parameters (α, β, γ, δ, μ).

The parameter value of 1 means the highest performance in each dimension of the measure. We formally defined the QoCS by the following ratio:

$$\theta = A_{shaded}/A_{pentagon}$$

The lower value of the θ measure means poor performance. The higher value of θ toward 1 implies higher performance.

Figures9.11(b–d) show the corresponding Kiviat graphs for three cloud services hypothetically running on three cloud plat- forms.

This is based on the aforementioned danalytical modelling of cloud performance. It would be interesting to validate the analytical results with real-life benchmark experiments on the cloud platforms from Amazon, Go ogle, and Salesforce.com.

94)What are online social networks? Enumerate the ideas for providing online social networking services. What are the benefits of social networks?

Ans: Online social networks (OSNs) are formed with individuals or organizations over the Internet. These individual or organiz- ational entities are related, connected, or associated special interests or specific interdependencies.
A social network is a structure representing the social relationships of individuals. In a social network, nodes represent the individuals, and the ties between the nodes represent the relationships such as friendship, kinship, and colleagueship[3]. OSN services are built or eflect the social relationships among people.

.Here are some ideas for providing OSN services:
•Personal page or profiles for each user linked by social connections
•Social graph traversal along specific social links or networks
•Communication tools among participants or registered users
•Ability to share music, photos, and videos with friends or professional groups
•Operation of a community in special niche areas like health care, sports, and hobbies
•Customized software tools or databases are used in OSN services
•Strong customer loyalty and fast membership growth are seen
•Provider revenue from embedded advertisement and access to premium content

social network community can be built around a forum to result in the following four operational benefits:

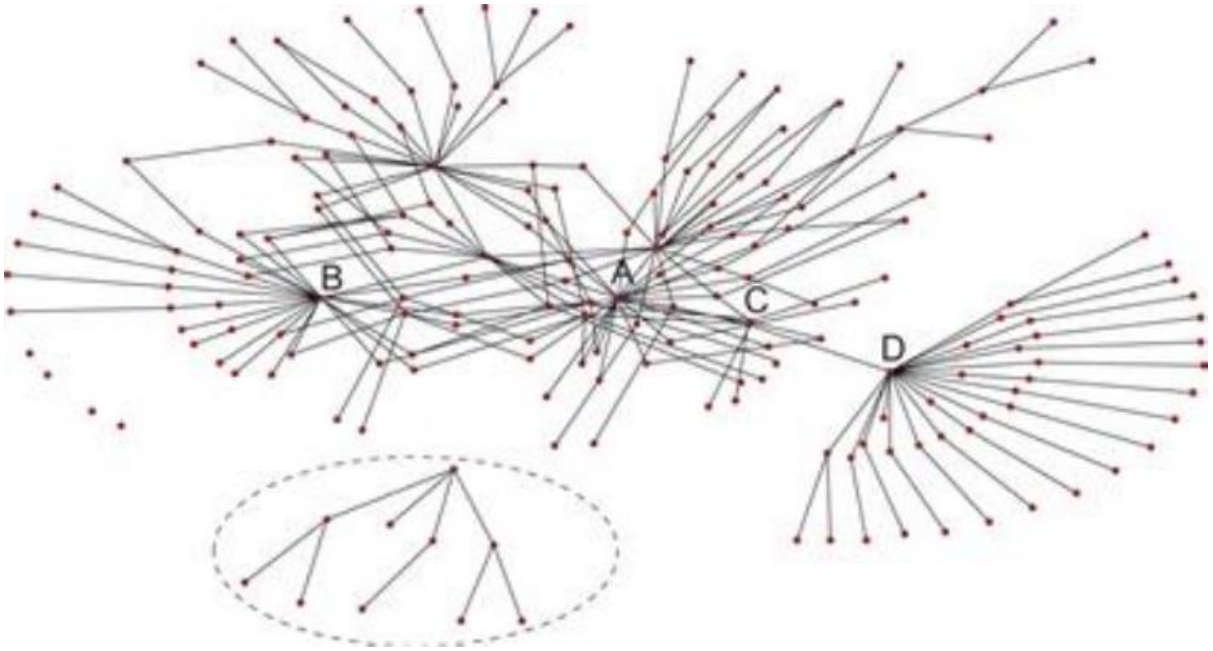
•High return visit rate: Users returntothesocial network community frequently.This opens up the opportunity forgreat page impressions and a huge advertising inventory.

•User loyalty : Users connect to their friends and will not a band on them easily. They do not move over to a competing social network. Instead, they show high customer loyalty.

•Virtual growth Members invite their friends to the social network community. This is effective marketing at a low cost, and the OSN grows by itself.

•Business model With a social network community you can earn revenues through subscriptions to premium content in addition to advertising revenues.

95)List and explain the properties of social network graph.



Social networks play a critical role in problem solving, running an organization, and the degree to which individuals succeed in achieving their goals. A social network is simply a map of all of the relevant ties between all the actor nodes.

The black dots are the nodes (users) and the lines link the nodes under specified relationships.

Node Degree, Reach, Path Length, and Betweenness: The node degree is the number of ties of a node to other actors in the graph. The reach is defined as the degree any member of a network can reach other members of the network. Path length measures the distance between pairs of nodes in the network.

Average path length is the average of these distances between all pairs of nodes. Betweenness reveals the extent to which a node lies between other nodes in the network. The measure reflects the number of people with whom a person is connecting indirectly through his direct links.

Closeness and Cohesion: Closeness is the degree to which an individual is near all other individuals in a network (directly or indirectly). It reflects the ability to access information through network members. Thus, closeness is the inverse of the sum of the shortest distances between each individual and every other person in the network. Cohesion is the degree to which actors are connected directly to one another by cohesive bonds. Groups are identified as “cliques” if every individual is directly tied to every other individual.

Social Circles or Clusters :This refers to some structured groups. If there is less stringency of direct contact or as structurally cohesive blocks, a social circle can be created either loosely or tightly, depending on the stringency rules applied. Those

nodes inside the circle form an isolated cluster. Clustering coefficient is the likelihood that two associates of a node are associates themselves.

Centralized versus Decentralized Networks: Centrality gives a rough indication of the social power of a node based on how well it “connects” the network. Betweenness, closeness, and degree are all measures of centrality. Centralized networks have links dispersed around one or a few nodes, while a decentralized network is one in which there is little variation between the number of links each node possesses.

Bridge and Local Bridge: An edge is a bridge if deleting it would cause its endpoints to lie in different clusters or components of a graph. For example,the edge between nodes C and D is a bridge. The endpoints of a local bridge share no common neighbors. A local bridge is contained in a cycle.

Prestige and Radiality: In a social graph, prestige describes a node’s centrality. Degree prestige, proximity prestige, and status prestige are all measures of prestige. Radiality is the degree to which a network reaches out and provides novel information and influence.

Structural Cohesion, Equivalence, and Holes: Structural cohesion is the minimum number of members who, if removed from a group, would disconnect the group. Structural equivalence refers to the extent to which nodes have a common set of linkages to other nodes. These nodes do not have any ties to one another. A structural hole can be filled by connecting one or more links to reach other nodes. This is related to social capital: By linking two disconnected people, you can control their communication.

96)What are the different communities that can be formed from online social networking?

Ans: .One can form many other communities from online social networking.

• Industry communities Special industrial workers or professionals are often connected with one another. They share knowledge and work experience.

•Artist communities These network communities are specifically composed to enable artists, musicians, or celebrities to personalize and intensify their contact with their existing fans as well as enabling contact among community members.

•Sport communities These are network communities for special interests and activities of athletes and sport fans. People can find friends, celebrate their passion, and exchange ideas.

•Health communitiesThese are dedicated to the needs of actors concerned about health issues.

•Congresses and event communities These are customized to support all preparations necessary for congresses and events, as well as all processes thereafter.

•Alumni communities After completing their studies, alumni can find fellow students, stay in touch, and foster friend- ships.

97)Explain the different application domains of social networks

Ans: There are an infinite number of applications for social networks. In following Table, we classify those applications in to five specific domains.

Examples and their web sites are listed for our readers to explore. The use of social network services in an enter- prise context presents the potential of having a major impact on the world of business and work. Social networks connect people at a low cost; this can be beneficial for entrepreneurs and small businesses looking to expand their contact bases.

These networks of ten act as a customer relationship management to ol for company esselling products and services.

Companies scan al sousesocial networks to advertize. With global operations, social network scan make it easier to promote business around the world.

Applications of Existing Social Networks

Domains	Examples and Web Sites
Business	LinkedIn connects professionals (www.linkedin.com) and Hub Culture connects entrepreneurs (www.hubculture.com).
Education	The National School Boards Association and education topics can be researched online (www.ning.com).
Government	The U.S. Centers for Disease Control and Prevention demonstrated the importance of vaccinations on www.whyville.net, a popular children's site. The National Oceanic and Atmospheric Administration has a virtual island on ht-tp://secondlife.com, where people can explore underground caves or explore the effects of global warming.
Medical and health care	www.patientslikeme.com offers its members the chance to connect with others dealing with similar issues and research patient data related to their condition. www.sobercircle.com, which has joined forces with www.onerecovery.com, gives people in recovery the ability to communicate with one another and strengthen their recovery.
Online dating	Online social networking sites such as Facebook, Second Life, and MySpace are quickly becoming the new way to find dates online.

i) LinkedIn for Professional Networking Social networks for professional connections include LinkedIn.com, Xing.com, and Ecademy.com. LinkedIn is a business- oriented social networking site. LinkedIn's main focus is to manage and build online and/or offline professional networks.
LinkedInhasmorethan80millionmembersinmorethan200countries.AnewmemberjoinsL

inkedInapproximatelyevery second,andabouthalfofthemembersareoutsidetheUnitedStates.ExecutivesfromallFortune500companiesareLinkedIn members. This site allows registered users to maintain a list of contact details of people they know and trust in business. Users can invite anyone to become a member.

ii) Educational Applications The National School Boards Association reports that almost 60 percent of students who use social networking talk about education topics online and, surprisingly, more than 50 percent talk specifically about schoolwork. Yet the vast majority of school districts have stringent rules against nearly all forms of social networking during the school day—even though stu- dentsandparentsreportfewproblembehavioronline.Socialnetworksfocusedonsupportingrelationshipsbetweenteach- ers and between teachers and their students are now used for learning, educator professional development, and content sharing.Ningforteachers,LearnCentral,[60]TeachStreet,andothersitesarebeingbuilttofosterrelationshipsthatinclude educationalblogs,e-portfolios,formalandadhoccommunities,aswellascommunicationsuchaschats,discussionthreads, and synchronous forums.

iii) Government Applications Social networking tools serve as a quick and easy way for the government to get the opinion of the public and to keep the public updated on their activity. The U.S. Centers for Disease Control and Prevention demonstrated the importance of vac-cinationsonthepopularchildren'ssiteWhyvilleandtheNationalOceanicandAtmosphericAdministrationhasavirtualis-landonSecondLifewherepeoplecanexploreundergroundcavesorexploretheeffectsofglobalwarming.Similarly,NASA hastakenadvantageofafewsocial networking tools, includingTwitterandFlickr.Theyusethesetools topromote avigor- ous and sustainable path to achieving its boldest aspirations in space.

iv) Health Care and Medical Applications Socialnetworksarebeginningtobeadoptedbyhealthcareprofessionalsasameanstomanageinstitutionalknowledge,dis-seminateP2Pknowledge,andhighlightindividualphysiciansandinstitutions.Theadvantageofusingadedicated medical social networking site is that all the members are screened against the state licensing board list of practitioners. The role of social networks is especially of interest to pharmaceutical companies who spend approximately 32 percent of their market- ing dollars to influence the opinion leaders of social networks. A new trend is emerging with social networks created to help members with various physical and mental ailments. For people suffering from life-altering diseases, PatientsLikeMe offers its members the chance to connect with others dealing with similar issues and research patient data related to their condition. For alcoholics and addicts, SoberCircle gives people inrecoverytheabilitytocommunicatewithoneanotherandstrengthentheirrecoverythroughtheencouragementofothers who can relate to their situation. DailyStrength is also a

web site that offers support groups for a wide array of topics and conditions, including the support topics offered by PatientsLikeMe and SoberCircle. SparkPeople offers community and so- cial networking tools for peer support during weight loss.

v) Dating Services Many social networks provide an online environment for people to communicate and exchange personal information for dating purposes. Intentions can vary from looking for a one-time date to looking for a short-term or long-term relationship. Mostofthesedatingservicesrequireuserstogiveoutsomepersonalinformationsuchasage,gender,location,interests,and perhaps a picture. Releasing personal information is often discouraged for safety reasons. This allows other users to search or be searched, but at the same time people can maintain a degree of anonymity. Online dating sites are similar to social networks in the sense that users create profiles to meet and communicate with others. Online dating sites usually require a fee, where social networks are free. This difference is one of the reasons the online dating industry is seeing a massive decrease in revenue due to many users opting to use free social networking services. Many popular online dating services such as Match.com, Yahoo! Personals, and eHarmony.com are seeing a decrease in users, where social networks such as MySpace and Facebook are experiencing an increase in users. The number of Internet usersintheU.S.thatvisitonlinedatingsiteshasfallenfromapeakof21percentin2003to10percentin2006.Inotherwords, social networks are gradually taking over most of the dating services.

98)Explain the functionality of different Facebook features.

Ans: Facebook includes six essential items, as shown in Table 9.12. Facebook needs to improve in terms of privacy. Compared with Twitter, Facebook has stricter access control, and thus is more secure to use. However, one must be aware that all per- sonal data is stored on the provider’s servers, including education and career information as well as personal messages and social links. Security and trust are the major concerns that prevent users from joining or staying active on Facebook. Users are willing to share more information because they like to communicate with their real-life friends.

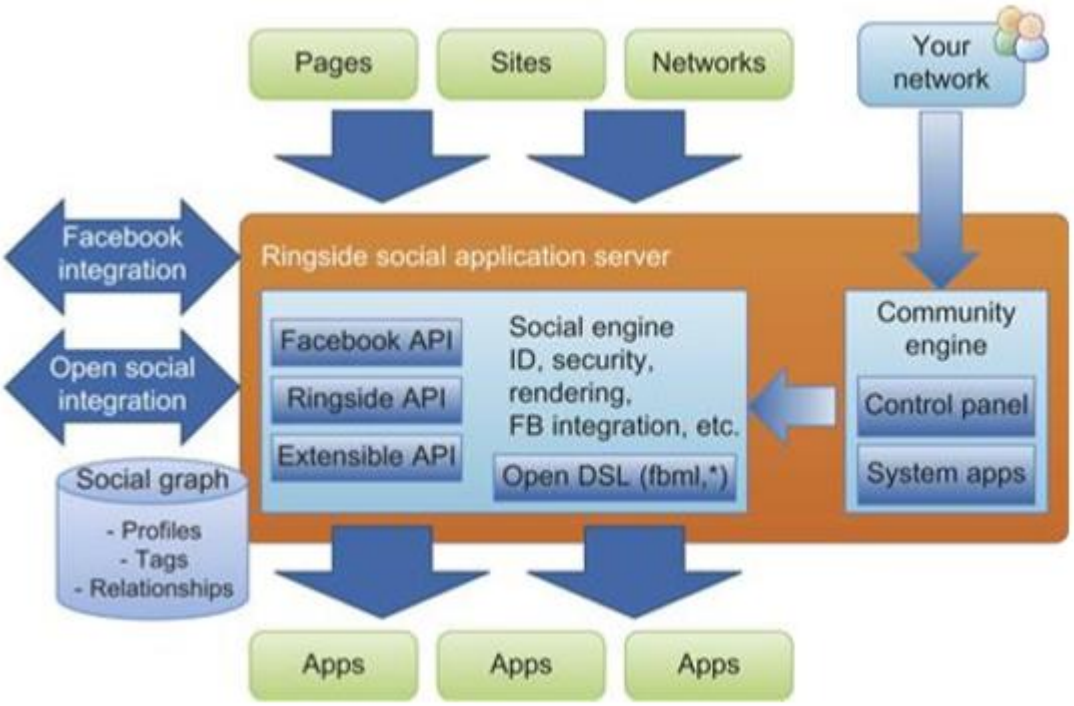
Functionality of Facebook Features

Function	Implementation
Profile page	Combined profile: profile picture, bio information, friends list, user's activity log, public message board, other components selectively displayed
Social graph traversal	Access through user's friends list on profile pages, with access control
Communication tools	Internal e-mail-like message: send and receive private messages among friends; instant messaging: accessed on the web page, or through third-party client; public message board: "Wall," with access control; update status: a short message, like micro-blogging, with access control
Shared information	Photo album: built-in, with access control; links: post links to outside URL, will appear on the activity log; videos: embedded outside videos on profile page
Access control	Every item on the profile page can be set to one of four access control levels: only me, only friends, friends of friends, or everyone
Special APIs	Games, calendars, mobile clients

99)With the help of a neat diagram explain the architecture of Facebook

Ans: With 600 million active users, you can imagine how many personal profiles and multimedia information items are shared by various social groups on Facebook. You can also imagine how much user traffic is flowing into the Facebook web site (www.facebook.com).

The Facebook platform is essentially a hugged at a center with a very largest or age capacity, intelligent file systems, and searching capabilities.Figure 9.29illustrates the structure of the Facebook platform.



The platform is formed with a huge cluster of servers. The semachinesare called ring side social applicati on servers.

The social engine is the core of the server, which performs the functions of identification, security,rendering,and Facebook integration.

Three specific API sare installed to facilitate user access.

A community engine provides networking services tousers. Requests are shown as pages, sites, and networks entering the Facebook server from the top. The social engine executes all sorts of user applications. Open DSL is used to support application execution.

100)What is Twitter? With the help of a neat diagram, explain the architecture of Twitter and access sequence control.

Ans: Twitter is a microblogging service launchedin2006 by JackDorsey, BizStone, and EvanWilliams.

Twitter can be no more than140characters in length. This idea originated from Short Message Service (SMS) technology over mobile devices such as smart phones.

A user can use SMS to send a message toTwitter,and the message will be forwarded to many other Twitter users that are related to the sender.

Twitter was built to share thoughts or ideas instantly. But gradually majorTV networks and newspaper also started to use Twitter to provide updated news to the public.

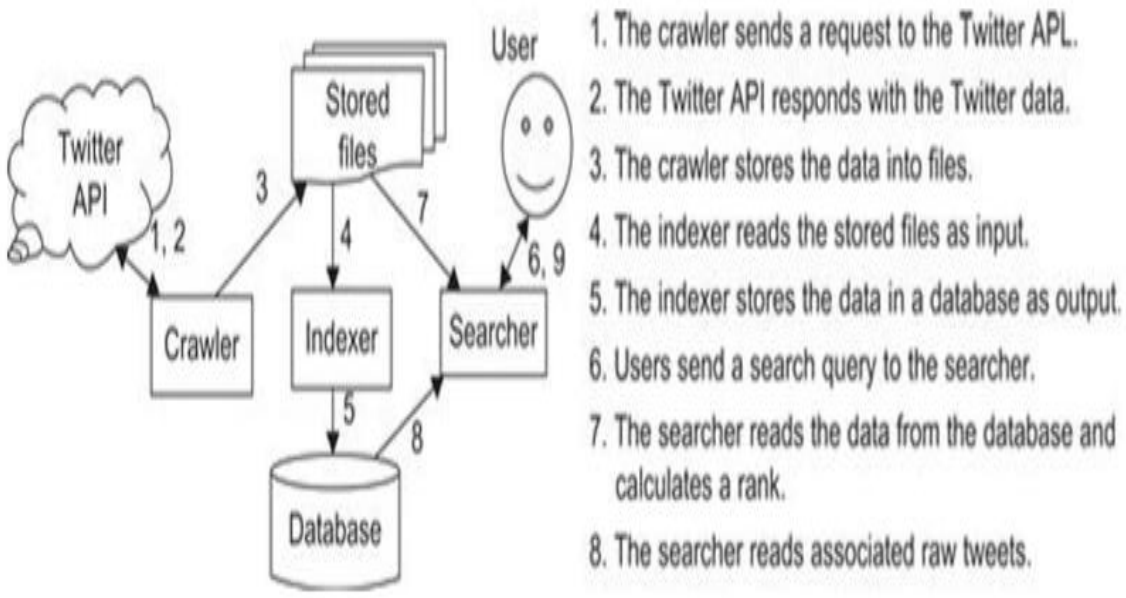
Twitter has become a media platform in which ordinary users can be in the front line of breaking news.

* Twitter Architecture and Access Control:

There are three major component sin the Twitter system: crawler ,indexer, and searcher.

Each component differs in the functions it performs. They are used to get her through balanced coordination to formacomplete system.Figure shows theTwitter access architecture in eight steps.

Twitter.com started as a status sharing site that people accessed via their mobile phones. Twitter soon became a Web 2.0 microblogging site for information sharing and news reporting. As of late 2010, Twitter had 175 million users. Furthermore, HubSpot reported that Twitter was enjoying an astonishing 18,000 percent growth rate.



101)Explain the different functions of twitter and their implementations.

Ans: Twitter provides six essential functions in Table 9.13, two of which are access control and API applications. Simplicity and opennessarethekeystoTwitter'spopularity.ThecoreserviceofTwitterisverysimple:justsendingmessagestoagroupof people.Allotherusesarecontributedbythird-partydevelopers.Twitterprovidesaveryopen API,sothird-partydevelopers cancreatepowerfulclientsorspecialapplicationsforTwitter.TheconnectionsonTwitterare notofflinesocialrelationships. Users can follow someone they do not know in real life. Not everyone on Twitter is using his real name, and even when realnamesareused,thereisnoeffectivewaytoidentifyusers.Eversincelaunchedin2009,Twitterhasprovidedaverified- account service for celebrities and official agencies. However, the revealing of personal information on Twitter still poses a privacy problem.

Functions of Twitter and Their Implementations

Function	Implementation
Profile page	Simple profile: profile picture, bio information, followers list, following list, user's message timeline
Social graph traversal	Access through the following and follower's lists on users' profile pages, without access control
Communication tools	Internal e-mail: send private direct messages to followers; instant messaging: not really instant, can be implemented by direct messages; public message board: message timeline, with access control
Share information	Photo album: third party, like Twitpic, but no combined access control; links: post links to outside URL, will appear as a message on timeline
Access control	Message timeline on profile page can be set access control in two levels: private or public
API applications	Mobile clients, PC clients, photo sharing, automatic news feed

Twitter can not be regarded as a fully trusted network or areliable acquaintancebasis.

Consider two types of accounts :pub- lic and protected accounts. Anyone can see updates at public accounts. For protected accounts, only the users that have been approved can see their updates. In order to prevent identity confusion, Twitter selects accounts from celebrities to provide verification. The verified notice is posted on verified accounts. Twitter provides two new features, Twitter Lists and Location, to support this verification process.

Twitter Lists all users to create a list of Twitter accounts.The list can help us ersorgan- izethe people they follow.The lists can also support location-based services.Location-sensitiveapplicationslikethoseaided by GPS devices can be greatly simplified with Twitter services.

102)Explain the different applications of Twitter.

Ans: Many interesting Twitter applications exist.

For instance, one can use the public to help determine business or fashion trends and identify popular music and movies. Twitter was even used during the 2008 presidential campaign in the United States to attract young voters to spread then-candidate Barack Obama's "Change" message. Twitter also gives users the option to send their current locations along with their tweets.

Twitter provides the Twitter API for developers to request Twitter data. The API supports REST calls.

Two types of APIs are supported: normal APIs and streaming APIs.

Lastly, Twitter also collects information about programming libraries that help create Twitter-based applications.

Two specific Twitter applications are described in the following list:

•Fast News Release Twitter is becoming the fastest way to spread breaking news. Massive user collaboration has given Twitter a clear edge over most news centers. Some news center shave setup Twitter accounts to encourage users to spread breaking news. For example, CNN maintains 45 official Twitter accounts with more than 5 million followers. During the election periods in various countries, Twitter attracted more attention than news centers. Some governments even warned Twitter.

•Alert Systems Twitter provides a system that can connect residents of a city with virtually no cost. It also increases the capabilities of an alert system by inputting more user-generated data. Some cities have already opted for Twitter to alert their residents. The Virginia Tech killings in 2007 highlighted the security issues on university campuses. To integrate thee 2 Campus emergency notification network with popular social networks, Pacific University of Forest Grove, Oregon, implemented a Twitter-based alert system for its students, and the trend is likely to grow.