U18PCCS401
CLOUD COMPUTING

UNIT I
UNDERSTANDING CLOUD COMPUTING

Cloud Computing : History of CC – Cloud Architecture – Cloud Storage – Why Cloud Computing Matters – Advantages CC – Disadvantages CC – Companies in the Cloud Today – Cloud Services - Web-Based Application.

UNIT-II
CLOUD ARCHITECTURE AND MODEL

Technologies for Network Based system - System Models for Distributed and Cloud Computing - NIST Cloud Computing Reference Architecture

Cloud models : Characteristics - Cloud Models (IaaS, PaaS, SaaS)

Public vs. Private Cloud-Cloud Solutions-Cloud ecosystem-Service Management-Computing on demand.

UNIT-III
CLOUD INFRASTRUCTURE

Architectural Design of compute and Storage Clouds-Layered Cloud Architecture Development-Design Challenges-Inter Cloud Resource Management-Resource Provisioning and Platform Deployment-Global Exchange of Cloud Resources.

UNIT-IV
PROGRAMMING MODEL

Parallel and Distributed Programming Paradigms-Map Reduce-Twister and Iterative Map Reduce-Hadoop Library from Apache-Mapping Applications-Programming Support-Google App Engine, Amazon AWS-Cloud Software Environments-Eucalyptus, Open Nebula, Open Stack, Aneka, CloudSim.

UNIT-V
CLOUD SECURITY

Security Overview-Cloud Security Challenges and Risks-Software-as-a-Service-Security Security Governance-Risk Management-Security Monitoring-Security Architecture Design-Data Security-Application Security-Virtual Machine Security-Identity Management and Access Control-Autonomic Security.

UNIT I
UNDERSTANDING CLOUD COMPUTING

- History of CC
- Cloud Architecture
- Cloud Storage
- Why Cloud Computing Matters
- Advantages CC
- Disadvantages CC
- Companies in the Cloud Today
- Cloud Services
- Web-Based Application

History of CC

In 1963, DARPA (the Defense Advanced Research Projects Agency), presented MIT with $2 million for Project MAC. The funding included a requirement MIT develop technology allowing for a "computer to be used by two or more people, simultaneously." In this case, one of those gigantic, archaic computers using reels of magnetic tape for memory and was the precursor to what has now become collectively known as Cloud Computing. It acted as a primitive Cloud with two or three people accessing it. The word "Virtualization" was used to describe this situation, though the word's meaning later expanded.

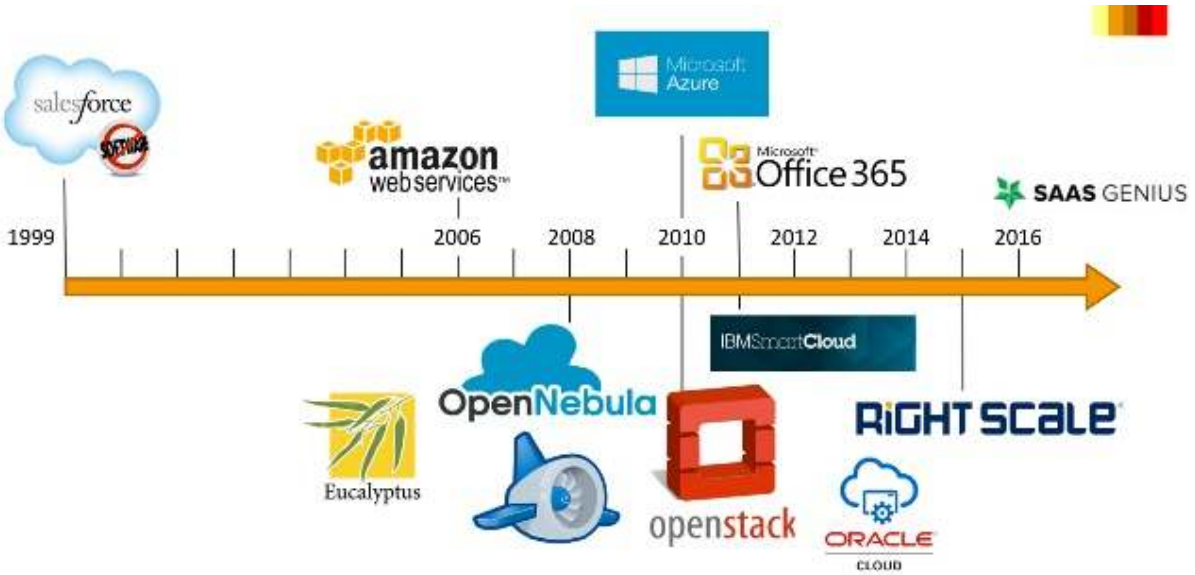
In 1969, J. C. R. Licklider helped develop the ARPANET (Advanced Research Projects Agency Network), a "very" primitive version of the Internet. JCR, or "Lick" was both a psychologist and a computer scientist, and promoted a vision called the "Intergalactic Computer Network," in which everyone on the planet would be interconnected by way of computers, and able to access information from anywhere. (What could such an unrealistic, impossible-to-pay-for, fantasy of the future look like?) The Intergalactic Computer Network, otherwise known as the Internet, is necessary for access to the Cloud.

The meaning of Virtualization began shifting in the 1970s, and now describes the creation of a virtual machine, that acts like a real computer, with a fully functional operating system. The concept of Virtualization has evolved with the Internet, as businesses began offering "virtual" private networks as a rentable service. The use of virtual computers became popular in the 1990s, leading to the development of the modern Cloud Computing infrastructure.

The Late 1990s

In its early stages, the Cloud was used to express the empty space between the end user and the provider. In 1997, Professor Ramnath Chellapa of Emory University defined Cloud Computing as the new "computing paradigm, where the boundaries of computing will be determined by economic rationale, rather than technical limits alone." This somewhat ponderous description rings true in describing the Cloud's evolution.

The Cloud gained popularity as companies gained a better understanding of its services and usefulness. In 1999, Salesforce became a popular example of using Cloud Computing successfully. They used it to pioneer the idea of using the Internet to deliver software programs to the end users. The program (or application) could be accessed and downloaded by anyone with Internet access. Businesses could purchase the software in an on-demand, cost-effective manner, without leaving the office.



The Early 2000s

In 2002, Amazon introduced its web-based retail services. It was the first major business to think of using only 10% of their capacity (which was commonplace at the time) as a problem to be solved. The Cloud Computing Infrastructure Model gave them the flexibility to use their computer's capacity much more efficiently. Soon after, other large organizations followed their example.

In 2006, Amazon launched Amazon Web Services, which offers online services to other websites, or clients. One of Amazon Web Services' sites, called Amazon Mechanical Turk, provides a variety of Cloud-based services including storage, computation and "human intelligence." Another of Amazon Web Services' sites is the Elastic Compute Cloud (EC2), allowing individuals to rent virtual computers and use their own programs and applications.

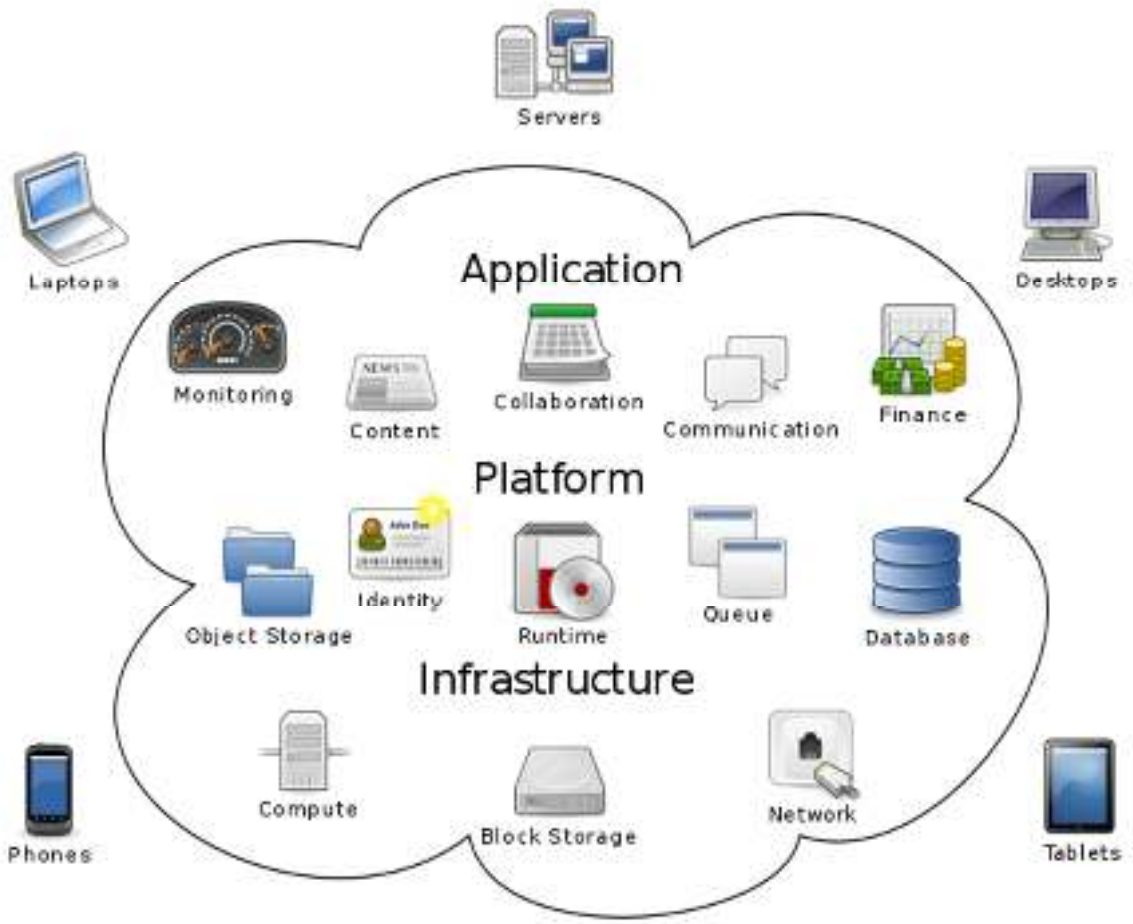
In the same year, Google launched the Google Docs services. Google Docs was originally based on two separate products, Google Spreadsheets and Writely. Google purchased Writely, which offers renters the ability to save documents, edit documents, and transfer them into blogging systems. (These documents are compatible with Microsoft Word.) Google Spreadsheets (acquired from 2Web Technologies, in 2005) is an Internet-based program allowing users to develop, update, and edit spreadsheets, and to share the data online. An Ajax-based program is used, which is compatible with Microsoft Excel. The spreadsheets can be saved in an HTML format.

In 2007, IBM, Google, and several universities joined forces to develop a server farm for research projects needing both fast processors and huge data sets. The University of Washington was the first to sign up and use resources provided by IBM and Google. Carnegie Mellon University, MIT, Stanford University, the University of Maryland, and the University of California at Berkeley, quickly followed suit. The universities immediately realized computer experiments can be done faster and for less money, if IBM and Google were supporting their research. Since much of the research was focused on problems IBM and Google had interests in, they also benefitted from the arrangement. 2007 was also the year when Netflix launched it's streaming video service, using the Cloud, and provided support for the practice of "binge-watching."

Eucalyptus offered the first AWS API compatible platform, which was used for distributing private Clouds, in 2008. In the same year, NASA's OpenNebula provided the first open-source software for deploying Private and Hybrid Clouds. Many of its most innovative features focused on the needs of major businesses.

In 2011, IBM introduced the IBM SmartCloud framework, in support of Smarter Planet (a cultural thinking project). Then, Apple launched the ICloud, which focuses on storing more personal information (photos, music, videos, etc.). Also, during this year, Microsoft began advertising the Cloud on television, making the general public aware of its ability to store photos, or video, with easy access.

Oracle introduced the Oracle Cloud in 2012, offering the three basics for business, IaaS (Infrastructure-as-a-Service), PaaS (Platform-as-a-Service), and SAAS (Software-as-a-Service).



Cloud Computing tutorial provides basic and advanced concepts of Cloud Computing. Our Cloud Computing tutorial is designed for beginners and professionals.

Cloud computing is a virtualization-based technology that allows us to create, configure, and customize applications via an internet connection. The cloud technology includes a development platform, hard disk, software application, and database.

What is Cloud Computing

The term cloud refers to a network or the internet. It is a technology that uses remote servers on the internet to store, manage, and access data online rather than local drives. The data can be anything such as files, images, documents, audio, video, and more.

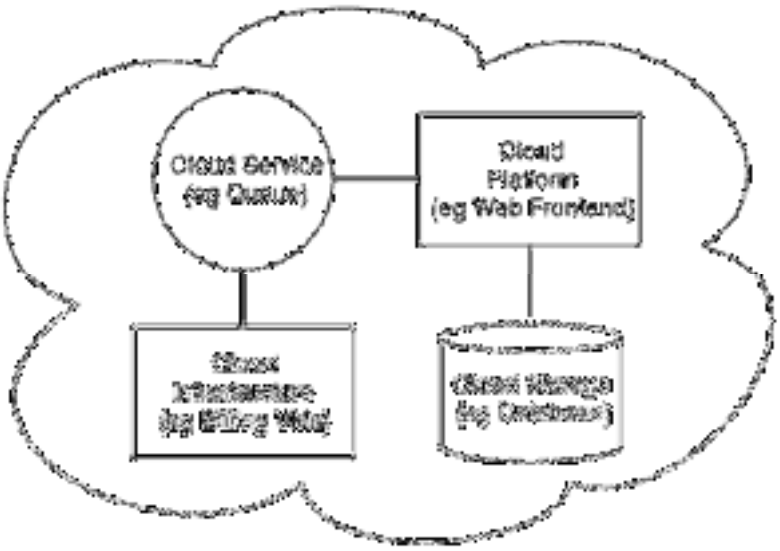
There are the following operations that we can do using cloud computing:

- Developing new applications and services
- Storage, back up, and recovery of data

- Hosting blogs and websites
- Delivery of software on demand
- Analysis of data
- Streaming videos and audios

Cloud Architecture

Cloud computing technology is used by both small and large organizations to **store the information** in cloud and **access** it from anywhere at anytime using the internet connection.



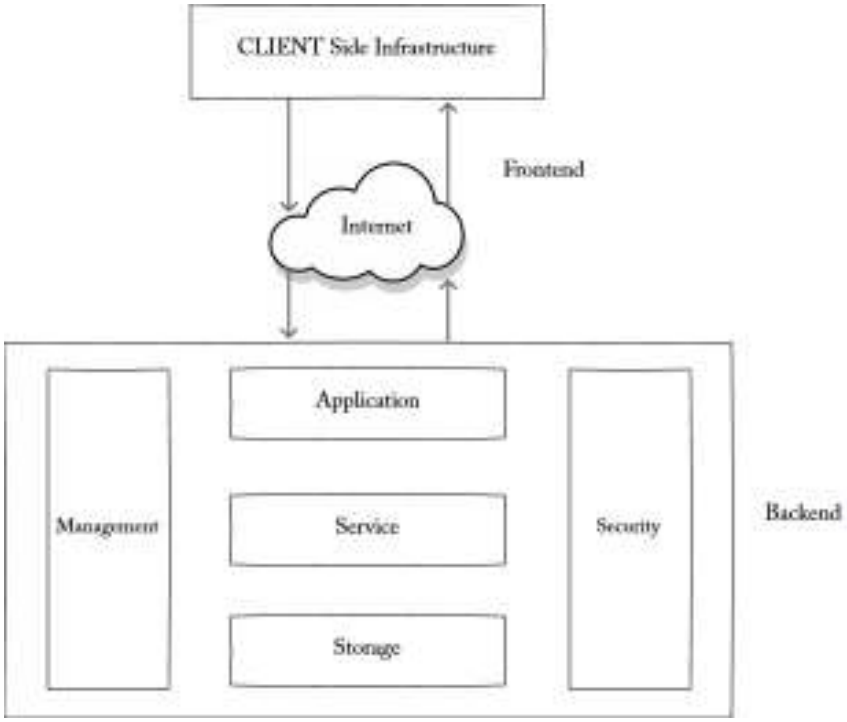
Cloud computing architecture is a combination of **service-oriented architecture** and **event-driven architecture**.

Cloud computing architecture refers to the components and subcomponents required for cloud computing. These components typically consist of a front end platform (fat client, thin client, mobile device), back end platforms (servers, storage), a cloud based delivery, and a network (Internet, Intranet, Intercloud). Combined, these components make up cloud computing architecture.

Cloud computing architecture is divided into the following two parts -

- Front End
- Back End

The below diagram shows the architecture of cloud computing -



Components of Cloud Computing Architecture

There are the following components of cloud computing architecture -

1. Client Infrastructure

Client Infrastructure is a Front end component. It provides GUI (Graphical User Interface) to interact with the cloud.

2. Application

The application may be any software or platform that a client wants to access.

3. Service

A Cloud Services manages that which type of service you access according to the client's requirement.

Cloud computing offers the following three type of services:

i. Software as a Service (SaaS) – It is also known as **cloud application services**. Mostly, SaaS applications run directly through the web browser means we do not require to download and install these applications. Some important example of SaaS is given below –

Example: Google Apps, Salesforce Dropbox, Slack, Hubspot, Cisco WebEx.

ii. Platform as a Service (PaaS) – It is also known as **cloud platform services**. It is quite similar to SaaS, but the difference is that PaaS provides a platform for software creation, but using SaaS, we can access software over the internet without the need of any platform.

Example: Windows Azure, Force.com, Magento Commerce Cloud, OpenShift.

iii. Infrastructure as a Service (IaaS) – It is also known as **cloud infrastructure services**. It is responsible for managing applications data, middleware, and runtime environments.

Example: Amazon Web Services (AWS) EC2, Google Compute Engine (GCE), Cisco Metapod.

4. Runtime Cloud

Runtime Cloud provides the **execution and runtime environment** to the virtual machines.

5. Storage

Storage is one of the most important components of cloud computing. It provides a huge amount of storage capacity in the cloud to store and manage data.

6. Infrastructure

It provides services on the **host level**, **application level**, and **network level**. Cloud infrastructure includes hardware and software components such as servers, storage, network devices, virtualization software, and other storage resources that are needed to support the cloud computing model.

7. Management

Management is used to manage components such as application, service, runtime cloud, storage, infrastructure, and other security issues in the backend and establish coordination between them.

8. Security

Security is an in-built back end component of cloud computing. It implements a security mechanism in the back end.

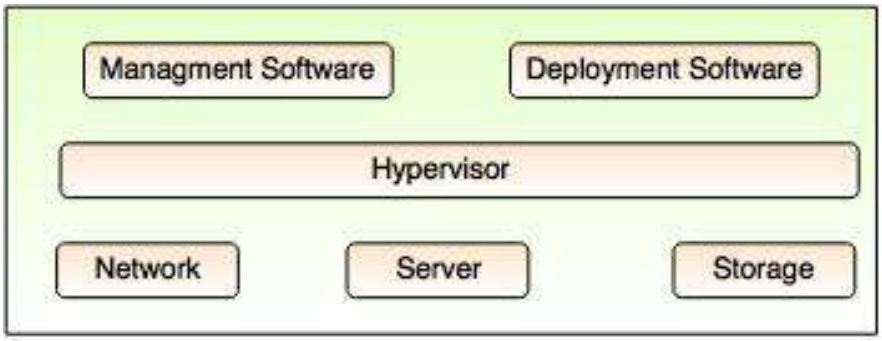
9. Internet

The Internet is medium through which front end and back end can interact and communicate with each other.

Cloud Storage or Infrastructure

- Cloud infrastructure means the hardware and software components.
- These components are server, storage, networking and virtualization software.
- These components are required to support the computing requirements of a cloud computing model.

Components of Cloud infrastructure



1) Hypervisor

- Hypervisor is a firmware or low-level program. It acts as a Virtual Machine Manager.
- It enables to share a physical instance of cloud resources between several customers.

2) Management Software

- Management software assists to maintain and configure the infrastructure.

3) Deployment Software

- Deployment software assists to deploy and integrate the application on the cloud.

4) Network

- Network is the key component of the cloud infrastructure.
- It enables to connect cloud services over the Internet.
- The customer can customize the network route and protocol i.e possible to deliver network as a utility over the Internet.

5) Server

- The server assists to compute the resource sharing and offers other services like resource allocation and deallocation, monitoring the resources, provide the security etc.

6) Storage

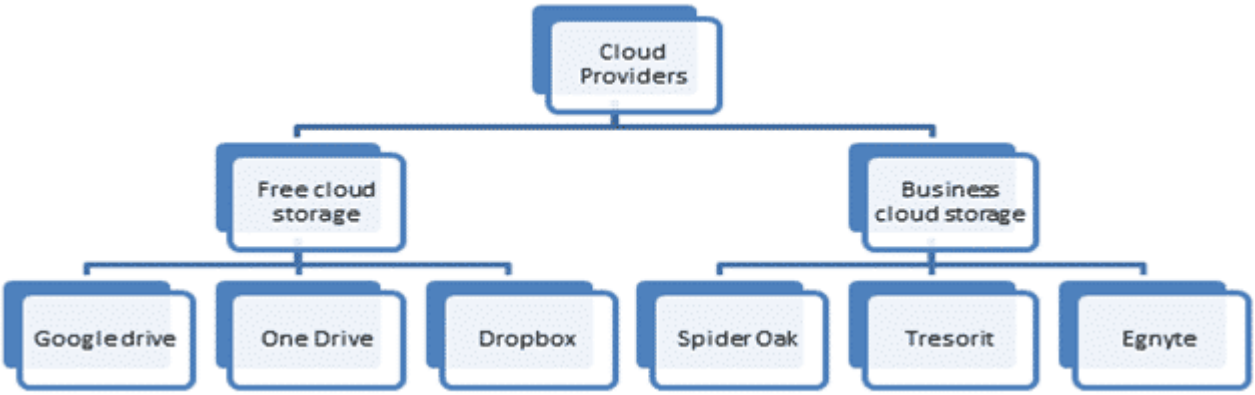
- Cloud keeps many copies of storage. Using these copies of resources, it extracts another resource if any one of the resources fails.

Example of Cloud Storage

- **Google's data servers.** Users can edit files using a Google application. Users can also publish documents so that other people can read them or even make edits, which means Google Docs is also an example of cloud computing.
- **Web e-mail** providers like Gmail, Hotmail and Yahoo! Mail store e-mail messages on their own servers. Users can access their e-mail from computers and other devices connected to the Internet.

- Sites like Flickr and Picasa host millions of digital photographs. Their users create online photo albums by uploading pictures directly to the services' servers.
- **YouTube** hosts millions of user-uploaded video files.
- **Social networking** sites like Facebook and MySpace allow members to post pictures and other content. All of that content is stored on the respective site's servers.

Cloud Storage Providers



Types of Cloud Storage

1. Personal Cloud Storage

Also known as mobile cloud storage, personal cloud storage is a subset of public cloud storage that applies to storing an individual's data in the cloud and providing the individual with access to the data from anywhere. It also provides data syncing and sharing capabilities across multiple devices. Apple's iCloud is an example of personal cloud storage.

2. Public Cloud Storage

Public cloud storage is where the enterprise and storage service provider are separate and there aren't any cloud resources stored in the enterprise's data center. The cloud storage provider fully manages the enterprise's public cloud storage.

3. Private Cloud Storage

A form of cloud storage where the enterprise and cloud storage provider are integrated in the enterprise's data center. In private cloud storage, the storage provider has infrastructure in the enterprise's data center that is typically managed

by the storage provider. Private cloud storage helps resolve the potential for security and performance concerns while still offering the advantages of cloud storage.

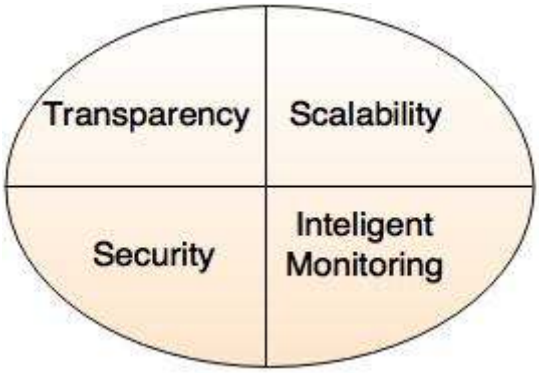
4. Hybrid Cloud Storage

Hybrid cloud storage is a combination of public and private cloud storage where some critical data resides in the enterprise's private cloud while other data is stored and accessible from a public cloud storage provider.

Infrastructural Constraints

Following are the basic constraints:

- Transparency
- Scalability
- Intelligent Monitoring
- Security



Transparency

- In a Cloud environment, virtualization is the key to share the resources.
- It is impossible to satisfy the demand with single resource or server. Hence, there should be transparency in resources, load balancing and application, so that we can scale the resources on demand.

Scalability

- Scaling up an application is easy but scaling up an application delivery solution is not easy, because it includes configuration overhead or even rearchitecting the network.
- Hence, application delivery is required to be scalable that need the virtual infrastructure so that, resources are provisioned and de-provisioned easily.

Intelligent Monitoring

- Application solution delivery is required to be capable of intelligent monitoring to achieve transparency and scalability.

Security

- In security, the mega data center in the cloud must be securely architected.
- The control node, an entry point in mega data center also requires to be secure.

Advantages of Cloud Storage

- **File Accessibility** – The files can be accessed at any time from any place so long as you have Internet access.
- **Offsite Backup** – Cloud Storage provides organizations with offsite (remote) backups of data which in turn reduces costs.
- **Effective Use of Bandwidth** – Cloud storage uses the bandwidth effectively i.e. instead of sending files to recipients, a web link can be sent through email.
- **Security of Data** – Helps in protecting the data against ransomware or malware as it is secured and needs proper authentication to access the stored data.

Disadvantages of Cloud Storage

- **Dependency on Internet Speed** – If the Internet connection is slow or unstable, we might have problems accessing or sharing the files.
- **Dependency on a Third Party** – A third party service provider (company) is responsible for the data stored and hence it becomes an important pre-requisite in selecting a vendor and to examine the security standards prior investing.
- **High Cost for Huge Data** – Organizations that require a large amount of storage may also find costs increase significantly even after the first few gigabytes of data stored.
- **No/ Minimal Control over Data Storage Framework** – Since the cloud storage framework is entirely managed and monitored by the service provider, the customer has minimal control over it .

Why Cloud Computing Matters

Small as well as large IT companies, follow the traditional methods to provide the IT infrastructure. That means **for any IT company, we need a Server Room that is the basic need of IT companies**.

In that server room, there should be a database server, mail server, networking, firewalls, routers, modem, switches, QPS (Query Per Second means how much queries or load will be handled by the server), configurable system, high net speed, and the maintenance engineers.

To establish such IT infrastructure, we need to spend lots of money. To overcome all these problems and to reduce the IT infrastructure cost, Cloud Computing comes into existence

The characteristics of cloud computing are given below:

1) Agility

The cloud **works in a distributed computing environment**. It shares resources among users and works very fast.

2) High availability and reliability

The availability of servers is high and more reliable because the **chances of infrastructure failure are minimum**.

3) High Scalability

Cloud offers "**on-demand**" **provisioning of resources on a large scale**, without having engineers for peak loads.

4) Multi-Sharing

With the help of cloud computing, **multiple users and applications can work more efficiently** with cost reductions by sharing common infrastructure.

5) Device and Location Independence

Cloud computing enables the users to access systems using a web browser regardless of their location or what device they use e.g. PC, mobile phone, etc. **As infrastructure is off-site** (typically provided by a third-party) **and accessed via the Internet, users can connect from anywhere**.

6) Maintenance

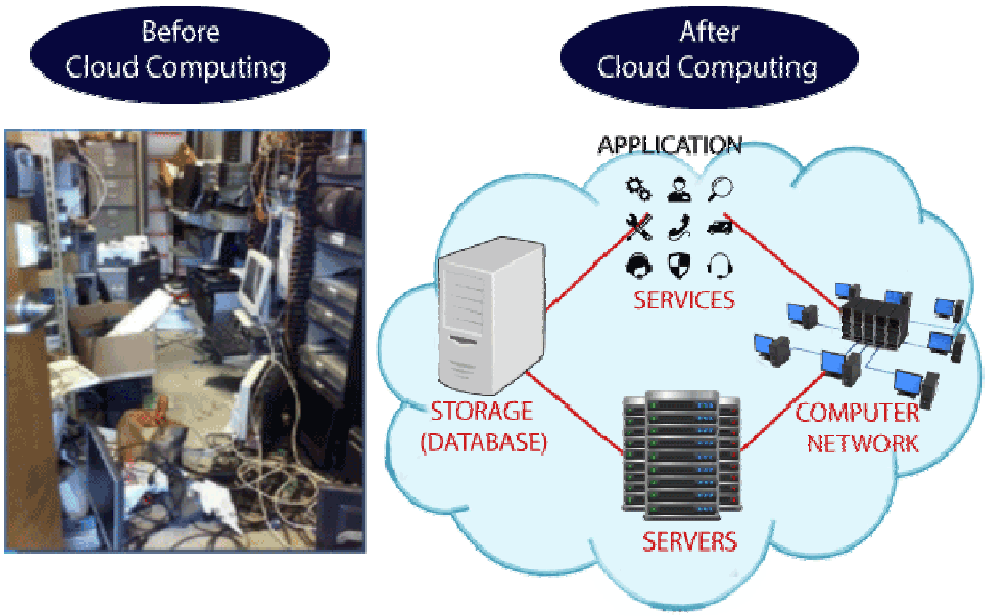
Maintenance of cloud computing applications is easier, since they **do not need to be installed on each user's computer and can be accessed from different places**. So, it reduces the cost also.

7) Low Cost

By using cloud computing, the cost will be reduced because to take the services of cloud computing, **IT company need not to set its own infrastructure** and pay-as-per usage of resources.

8) Services in the pay-per-use mode

Application Programming Interfaces **(APIs) are provided to the users so that they can access services on the cloud** by using these APIs **and pay the charges as per the usage of services**.



Advantages CC

1) Back-up and restore data

Once the data is stored in the cloud, it is easier to get back-up and restore that data using the cloud.

2) Improved collaboration

Cloud applications improve collaboration by allowing groups of people to quickly and easily share information in the cloud via shared storage.

3) Excellent accessibility

Cloud allows us to quickly and easily access store information anywhere, anytime in the whole world, using an internet connection. An internet cloud infrastructure increases organization productivity and efficiency by ensuring that our data is always accessible.

4) Low maintenance cost

Cloud computing reduces both hardware and software maintenance costs for organizations.

5) Mobility

Cloud computing allows us to easily access all cloud data via mobile.

6) IServices in the pay-per-use model

Cloud computing offers Application Programming Interfaces (APIs) to the users for access services on the cloud and pays the charges as per the usage of service.

7) Unlimited storage capacity

Cloud offers us a huge amount of storing capacity for storing our important data such as documents, images, audio, video, etc. in one place.

8) Data security

Data security is one of the biggest advantages of cloud computing. Cloud offers many advanced features related to security and ensures that data is securely stored and handled.

Disadvantages CC

1) Internet Connectivity

As you know, in cloud computing, every data (image, audio, video, etc.) is stored on the cloud, and we access these data through the cloud by using the internet connection. If you do not have good internet connectivity, you cannot access these data. However, we have no any other way to access data from the cloud.

2) Vendor lock-in

Vendor lock-in is the biggest disadvantage of cloud computing. Organizations may face problems when transferring their services from one vendor to another. As different vendors provide different platforms, that can cause difficulty moving from one cloud to another.

3) Limited Control

As we know, cloud infrastructure is completely owned, managed, and monitored by the service provider, so the cloud users have less control over the function and execution of services within a cloud infrastructure.

4) Security

Although cloud service providers implement the best security standards to store important information. But, before adopting cloud technology, you should be aware that you will be sending all your organization's sensitive information to a third party, i.e., a cloud computing service provider. While sending the data on the cloud, there may be a chance that your organization's information is hacked by Hackers.

Companies in the Cloud Today

Many companies are delivering services from the cloud. Some notable examples include the following:

• **Google** — Has a private cloud that it uses for delivering Google Docs and many other services to its users, including email access, document applications, text translations, maps, web analytics, and much more.

• **Microsoft** — Has Microsoft® Office 365® online service that allows for content and business intelligence tools to be moved into the cloud, and Microsoft currently makes its office applications available in a cloud.

• **Salesforce.com** — Runs its application set for its customers in a cloud, and its Force.com and Vmforce.com products provide developers with platforms to build customized cloud services.

Cloud Services

A **cloud service** is any **service** made available to users on demand via the Internet from a **cloud computing** provider's servers as opposed to being provided from a company's own on-premises servers

Simply put, **cloud computing** is the delivery of **computing services** including

- servers
- storage
- databases
- networking
- software
- analytics
- and intelligence

Over the Internet ("the **cloud**") to offer faster innovation, flexible resources, and economies of scale.

- **Virtual IT (information technology)**: Configure and deploy remote external servers as extensions to a company's local IT network.
- **Software**: Use commercial software applications, or develop and remotely host custom-built applications,
- **Network storage**: Network storage archives data across the internet to a provider without needing to know the physical location of storage.

Cloud computing systems all generally are designed to support large numbers of customers and surges in demand.

Software-As-A-Service Models
Cloud services using a software-as-a-service model, or SaaS, provide fully functional programs to end users even though the programs may not be resident on their local computers. Email providers like Gmail and Outlook.com are SaaS applications, as well as just about any computer program that runs inside a browser. As such, SaaS is most familiar to home consumers.

Platform-as-a-Service Models
A SaaS solution sits atop a platform. Vendors that offer platform-as-a-service portfolios generally face corporate clients. PaaS products include virtual servers, operating environments, database environments, and any other middleware component that sits between the hardware and the consumer-facing application.

Infrastructure-as-a-Service Models
Platforms, in turn, sit upon infrastructure. Infrastructure-as-a-service solutions generally get to the level of 'bare metal' — the physical servers, networking components, and device storage necessary to make platforms (and, hence, services) functional. IaaS is popular with corporate clients, with tradeoffs between speed, cost, and privacy that each vendor balances in different ways.

Examples of Cloud Computing Services
Many different vendors offer various types of cloud-computing services:

Amazon EC2 — Virtual IT
Google App Engine — Application hosting
Google Apps and Microsoft Office Online — SaaS
Apple iCloud — Network storage
DigitalOcean — Servers (Iaas/PaaS)

Examples of Cloud Storage (BRIEF)
Ex: Dropbox, Gmail, Facebook

The number of cloud storage providers online seems to grow every day. Each competing over the amount of storage they can provide to clients.

Right now, **Dropbox** is the clear leader in streamlined cloud storage allowing users to access files on any device through its application or website with up to 1 terabyte of free storage.

Google's email service provider Gmail, on the other hand, provides unlimited storage on the cloud. Gmail has revolutionized the way we send emails and largely responsible for the increased usage of email worldwide.

Facebook is a mix of the two, in that it can store an infinite amount of information, images, and videos on your profile. They can then be easily accessed on multiple devices. Facebook goes a step further with their Messenger app, which allows for profiles to exchange data.

Marketing Cloud Platforms
Ex: Maropost for Marketing, Hubspot, Adobe Marketing Cloud

A marketing cloud is an end-to-end digital marketing platform for clients to manage contacts and target leads. Maropost Marketing Cloud combines easy-to-use marketing automation and hyper-targeting of leads. At the same time, ensuring emails actually arrive in the inbox, thanks to its advanced email deliverability capabilities.

In general, marketing clouds fulfill a need for personalization. This is important in a market that demands messaging be "more human." That's why communicating that your brand is here to help, will make all the difference in closing.

Cloud Computing in Education
Ex: SlideRocket, Ratatype, Amazon Web Services

Education is increasingly adopting advanced technology because students already are. So, in an effort to modernize classrooms, educators have introduced e-learning software like SlideRocket.

SlideRocket is a platform that students can use to build presentations and submit them. Students can even present them through web conferencing all on the cloud. Another tool teachers use is Ratatype, which helps students learn to type faster and offers online typing tests to track their progress.

For school administration, Amazon's AWS Cloud for K12 and Primary Education features a virtual desktop infrastructure (VDI) solution. Through the cloud, allows instructors and students to access teaching and learning software on multiple devices.

Examples of Cloud Computing in Healthcare
Ex: ClearDATA, Dell's Secure Healthcare Cloud, IBM Cloud

Cloud computing lets nurses, physicians, and administrators share information quickly from anywhere. It also saves on costs by allowing large data files to be shared instantly for maximum convenience. This is a major boost for efficiency.

Ultimately, cloud technology ensures patients receive the best possible care without unnecessary delay. The patient's condition can also be updated in seconds through remote conferencing.

However, many modern hospitals have yet to implement cloud computing but are forecasted to do so in the near future.

Examples of Cloud Computing for Government
Uses: IT consolidation, shared services, citizen services

The U.S. government and military were early adopters of cloud computing. The U.S. Federal Cloud Computing Strategy, introduced under the Obama administration, was instituted to accelerate cloud adoption in all departments.

According to the strategy: "focus will shift from the technology itself to the core competencies and mission of the agency."

The U.S. government's cloud incorporates social, mobile and analytics technologies. However, they must adhere to strict compliance and security measures (FIPS, FISMA, and FedRAMP). This is to protect against cyber threats both domestic and abroad.

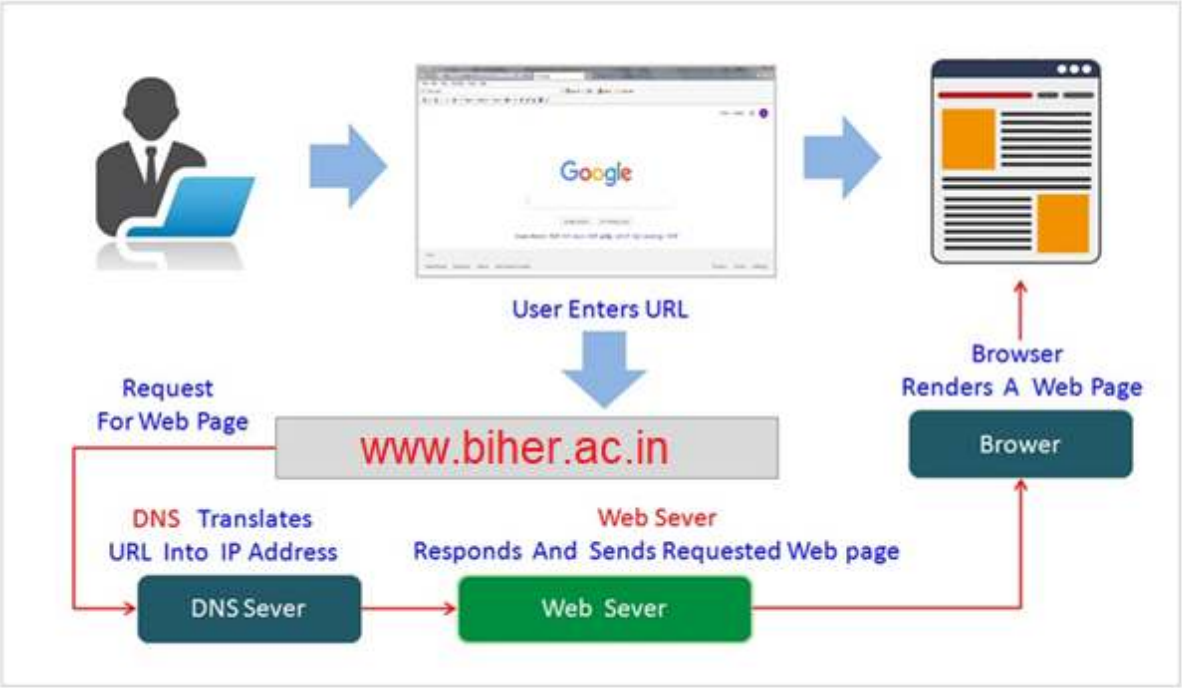
Cloud computing is the answer for any business struggling to stay organized, increase ROI, or grow their email lists. Maropost has the digital marketing solutions you need to transform your business.

Web-Based Application

Cloud services, in the form of centralized web-based applications, also appeal to the IT professional. One instance of an application hosted in the cloud is cheaper and easier to manage than individual copies of similar software installed on each user's desktop PC. Upgrading a cloud app only has to be done one time, where upgrading traditional software has to be done for each PC on which that software is installed. Then, of course, we have the promise of cloud-enabled collaboration, which just can't be done with traditional desktop apps.

How Websites Work

Let's start with the most obvious way of using the internet: You visit a website like **www.biher.ac.in**.



The computer user initiates the request for a web page . The browser then sends this request to the **DNS Sever** .

The DNS Server locates the Web Server where the website is hosted and translates the URL into IP address of the **Web Server**.

The Web Sever then responds with the requested web page and send this to the web browser that has initiated the request . The web browser then displays the web page to the user .

The **web server** also needs to have the operating system , **DBMS** (Database Management System) to manage the database stored on the server and server side programming language (scripting language) .

The server software enables the web server to offer various services and respond to the http requests sent by the client machines .

The web server operating system supports the client server architecture and provides a secured hosting environment to the websites hosted on the web server .

Web Server Solution Stack : LAMP , WAMP , XAMPP

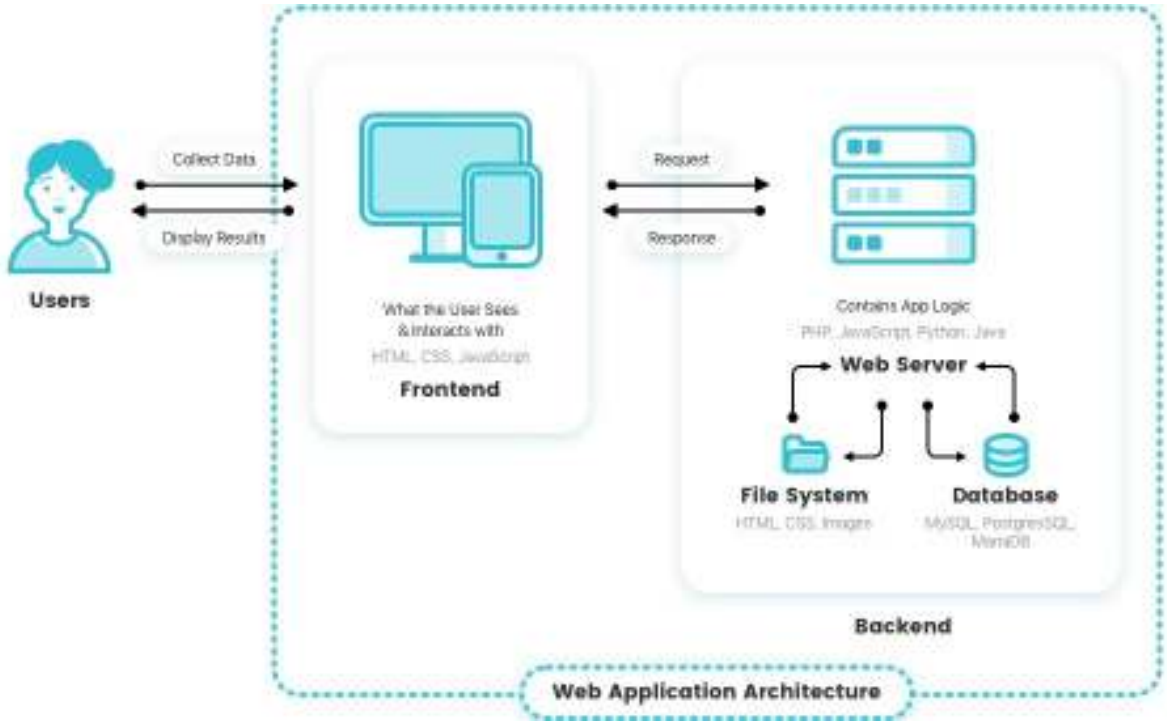
A web server software package (Solution Stack) consist of four components bundled together in addition to the Sever hardware (Computer system) .

These four server components include the operating system, Http web server, **DBMS** (Database) and a server side scripting language. All these four components together provide server side run-time environment.

Some of the most popular web server solution stack (bundled together web server software package) include LAMP , XAMPP and WAMP .

The LAMP web server package include combinations of Linux, Apache, MySQL (RDBMS), and PHP, abbreviated and named in the same order.

Another commonly used web server is WAMP for Windows based platforms . The WAMP includes Windows, Apache, MySQL and PHP.



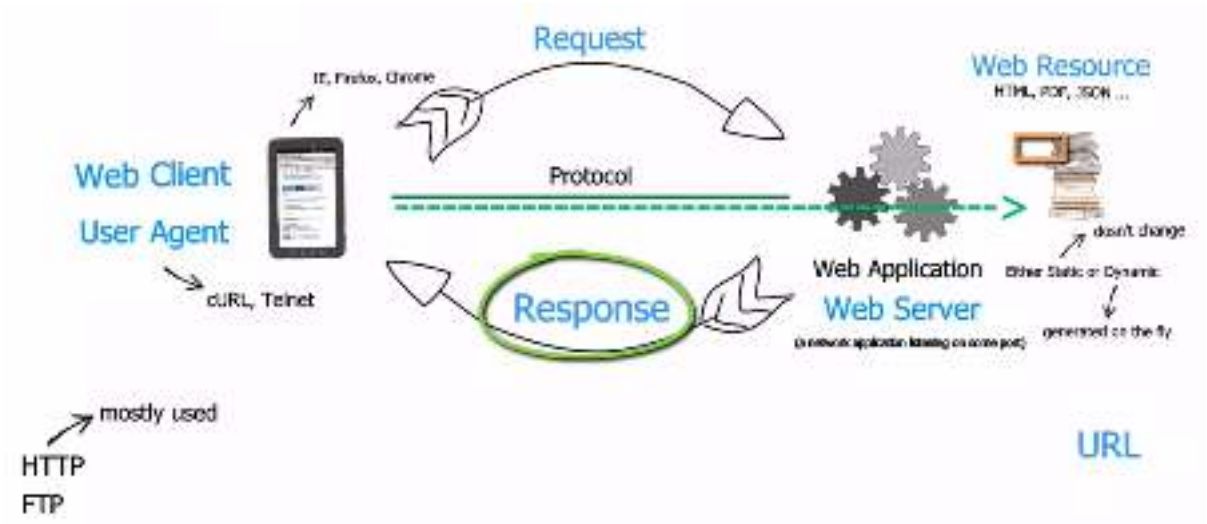
Web based application refers to any program that is accessed over a network using HTTP (Hyper Text Transfer Protocol). **Web based application** often run inside a **web browser**. We can often say that **web based application** are user **based** as we can download some part of the **application** to view it on the desktop

Components of Web application

All **web**-based database **applications** have three primary **components**: A **web** browser (or client), a **web application** server, and a database server.

How a web application works

Web applications are usually coded in browser-supported language such as JavaScript and HTML as these languages rely on the browser to render the program executable. Some of the applications are dynamic, requiring server-side processing. Others are completely static with no processing required at the server.



The web application requires a web server to manage requests from the client, an application server to perform the tasks requested, and, sometimes, a database to store the information. Application server technology ranges from ASP.NET, ASP and ColdFusion, to PHP and JSP.

Here's what a typical web application flow looks like:

1. **User** triggers a request to the **web server** over the **Internet**, either through a web browser or the application's user interface
2. **Web server** forwards this request to the appropriate **web application server**

3. **Web application server** performs the requested task – such as querying the **database** or processing the data – then generates the results of the requested data
4. **Web application server** sends results to the **web server** with the requested information or processed data
5. **Web server** responds back to the client with the requested information that then appears on the user's display

Example

Web applications include online forms, shopping carts, word processors, spreadsheets, video and photo editing, file conversion, file scanning, and email programs such as Gmail, Yahoo and AOL. Popular **applications** include Google Apps and Microsoft 365

Advantages

Advantages of **Web Applications**. **Web Applications** deliver many business benefits compared to office **based** solutions. Reduce business costs - less time spent talking to customers over the phone; eliminate printed materials; allow users to update their own details. Centralised data is secure and easy to backup.

UNIT-II
CLOUD ARCHITECTURE AND MODEL

- Technologies for Network Based system
- System Models for Distributed and Cloud Computing
- NIST Cloud Computing Reference Architecture
- **Cloud models** :
- Characteristics - Cloud Models (IaaS, PaaS, SaaS)
- Public vs. Private Cloud
- Cloud Solutions
- Cloud ecosystem
- Service Management
- Computing on demand.

Introduction

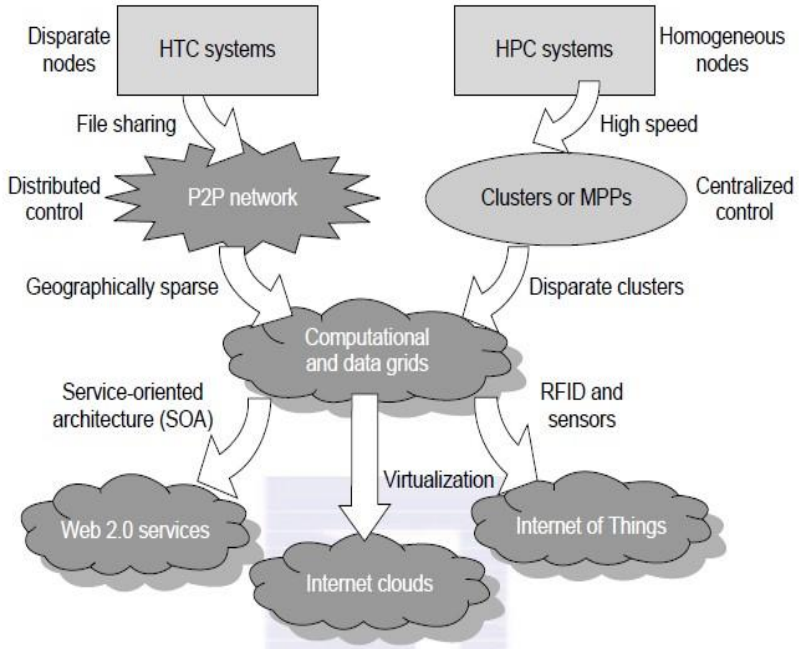


FIGURE 1.1

Evolutionary trend toward parallel, distributed, and cloud computing with clusters, MPPs, P2P networks, grids, clouds, web services, and the Internet of Things.

Technologies for Network Based system

With the concept of scalable computing under our belt, it's time to explore hardware, software, and network technologies for distributed computing system design and applications. In particular, we will focus on viable approaches to building distributed operating systems for handling massive parallelism in a distributed environment.

- Multicore CPUs and Multithreading Technologies
- GPU Computing to Exascale and Beyond
- Memory, Storage, and Wide-Area Networking
- Virtual Machines and Virtualization Middleware
- Data Center Virtualization for Cloud Computing

1.Multicore CPUs and Multithreading Technologies

1.a.Advance in CPU Processor

- Processor speed is measured in millions of instructions per second (MIPS) and network bandwidth is measured in megabits per second (Mbps) or gigabits per second (Gbps). The unit GE refers to 1 Gbps Ethernet bandwidth.

- Advances in CPU Processors advanced CPUs or microprocessor chips assume a multicore architecture with dual, quad, six, or more processing cores. These processors exploit parallelism at ILP and TLP levels.

- ILP mechanisms include multiple-issue superscalar architecture, dynamic branch prediction, and speculative execution, among others. These ILP techniques demand hardware and compiler support.

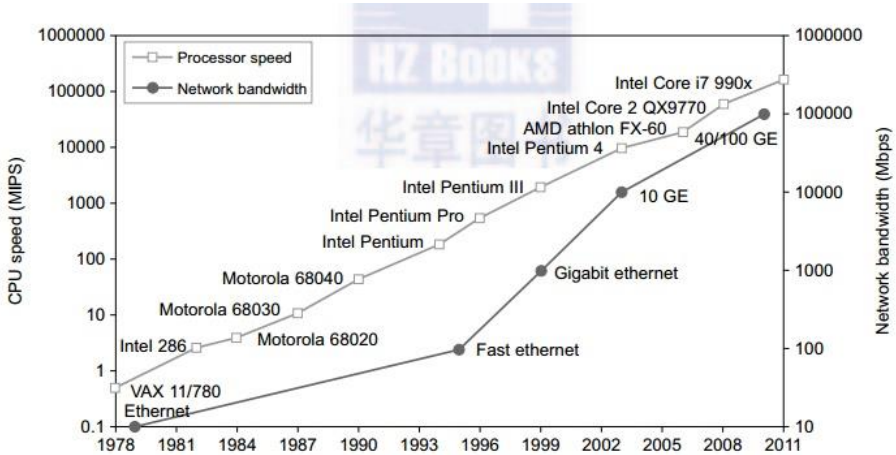


FIGURE 1.4
Improvement in processor and network technologies over 33 years.

- In addition, DLP and TLP are highly explored in graphics processing units (GPUs) that adopt a many-core architecture with hundreds to thousands of simple cores.

- Both multi-core CPU and many-core GPU processors can handle multiple instruction threads at different magnitudes today.

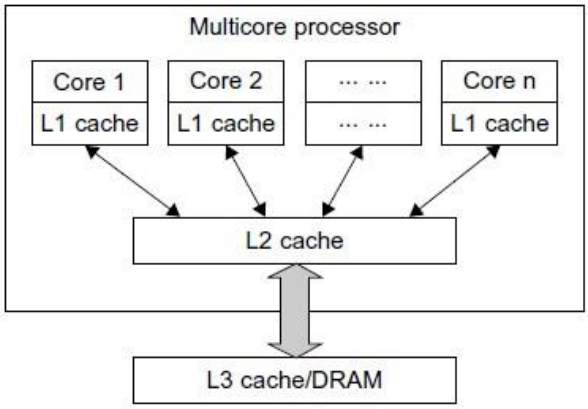


FIGURE 1.5

Schematic of a modern multicore CPU chip using a hierarchy of caches, where L1 cache is private to each core, on-chip L2 cache is shared and L3 cache or DRAM Is off the chip.

- Figure 1.5 shows the architecture of a typical multicore processor.
- Each core is essentially a processor with its own private cache (L1 cache).
- Multiple cores are housed in the same chip with an L2 cache that is shared by all cores.
- In the future, multiple CMPs could be built on the same CPU chip with even the L3 cache on the chip.
- Multicore and multithreaded CPUs are equipped with many high-end processors, including the Intel i7, Xeon, AMD Opteron, Sun Niagara, IBM Power 6, and X cell processors. Each core could be also multithreaded.

1.b. Multicore CPU and Many-Core GPU Architectures

- Multicore CPUs may increase from the tens of cores to hundreds or more in the future.
- But the CPU has reached its limit in terms of exploiting massive DLP due to the aforementioned memory wall problem. This has triggered the development of many-core GPUs with hundreds or more thin cores.

- Both IA-32 and IA-64 instruction set architectures are built into commercial CPUs. Now, x-86 processors have been extended to serve HPC and HTC systems in some high-end server processors.
- Many RISC processors have been replaced with multicore x-86 processors and many- core GPUs in the Top 500 systems. This trend indicates that x-86 upgrades will dominate in data centers and supercomputers.
- The GPU also has been applied in large clusters to build supercomputers in MPPs.

F

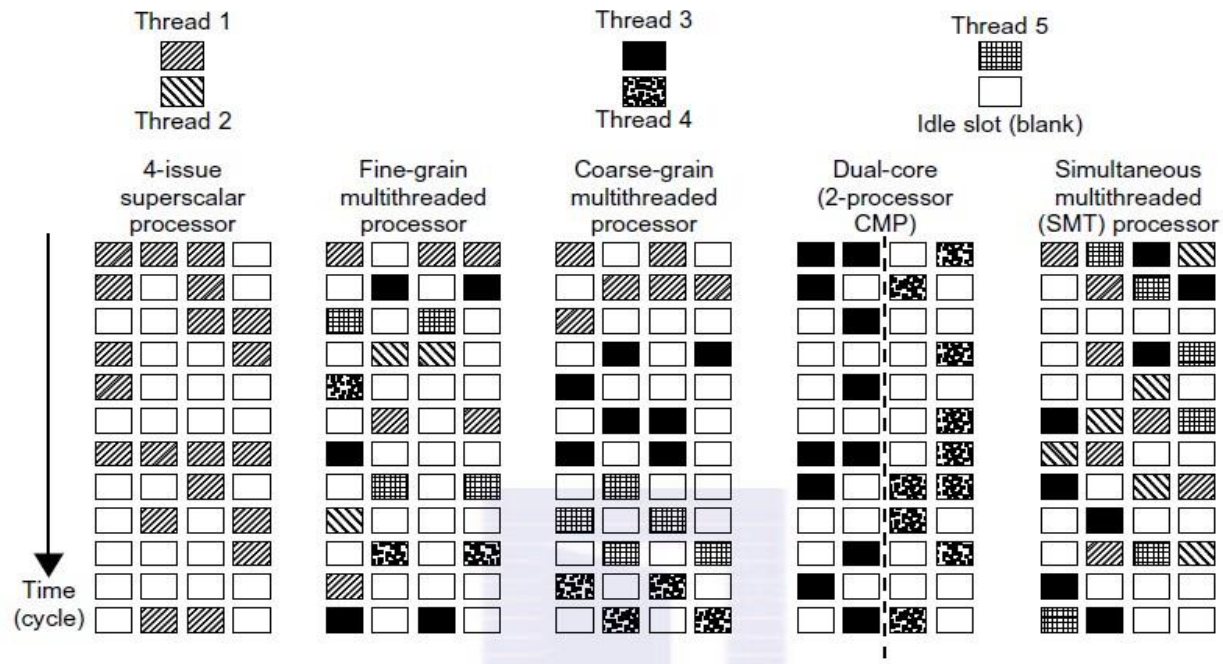


FIGURE 1.6

Five micro-architectures in modern CPU processors, that exploit ILP and TLP supported by multicore and multithreading technologies.

1.c. Multithreading Technology

- Consider in Figure 1.6 the dispatch of five independent threads of instructions to four pipelined data paths (functional units) in each of the following five processor categories,
- from left to right: a four-issue superscalar processor, a fine-grain multithreaded processor, a coarse-grain multithreaded processor, a two-core CMP, and a simultaneous multithreaded (SMT) processor.
- The superscalar processor is single-threaded with four functional units.
- Each of the three multithreaded processors is four-way multithreaded over four functional data paths.
- In the dual-core processor, assume two processing cores, each a single-threaded two-way superscalar processor.

- Instructions from different threads are distinguished by specific shading patterns for instructions from five independent threads.
- Typical instruction scheduling patterns are shown here. Only instructions from the same thread are executed in a superscalar processor.
- Fine-grain multithreading switches the execution of instructions from different threads per cycle.
- Course-grain multithreading executes many instructions from the same thread for quite a few cycles before switching to another thread.
- The multicore CMP executes instructions from different threads completely.
- The SMT allows simultaneous scheduling of instructions from different threads in the same cycle.
- These execution patterns closely mimic an ordinary program.
- The blank squares correspond to no. of available instructions for an instruction data path at a particular processor cycle.
- More blank cells imply lower scheduling efficiency. The maximum ILP or maximum TLP is difficult to achieve at each processor cycle.

2. GPU Computing to Exascale and Beyond

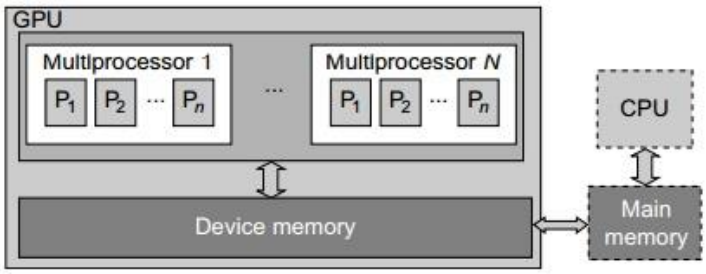


FIGURE 1.7

The use of a GPU along with a CPU for massively parallel execution in hundreds or thousands of processing cores.

- Figure 1.7 shows the interaction between a CPU and GPU in performing parallel execution of floating-point operations concurrently.
- The CPU is the conventional multicore processor with limited parallelism to exploit.
- The GPU has a many-core architecture that has hundreds of simple processing cores organized as multiprocessors.
- Each core can have one or more threads.
- Essentially, the CPU's floating-point kernel computation role is largely offloaded to the many- core GPU.
- The CPU instructs the GPU to perform massive data processing.

- The bandwidth must be matched between the on-board main memory and the on-chip GPU memory.
- This process is carried out in NVIDIA's CUDA programming using the GeForce 8800 or Tesla and Fermi GPUs.

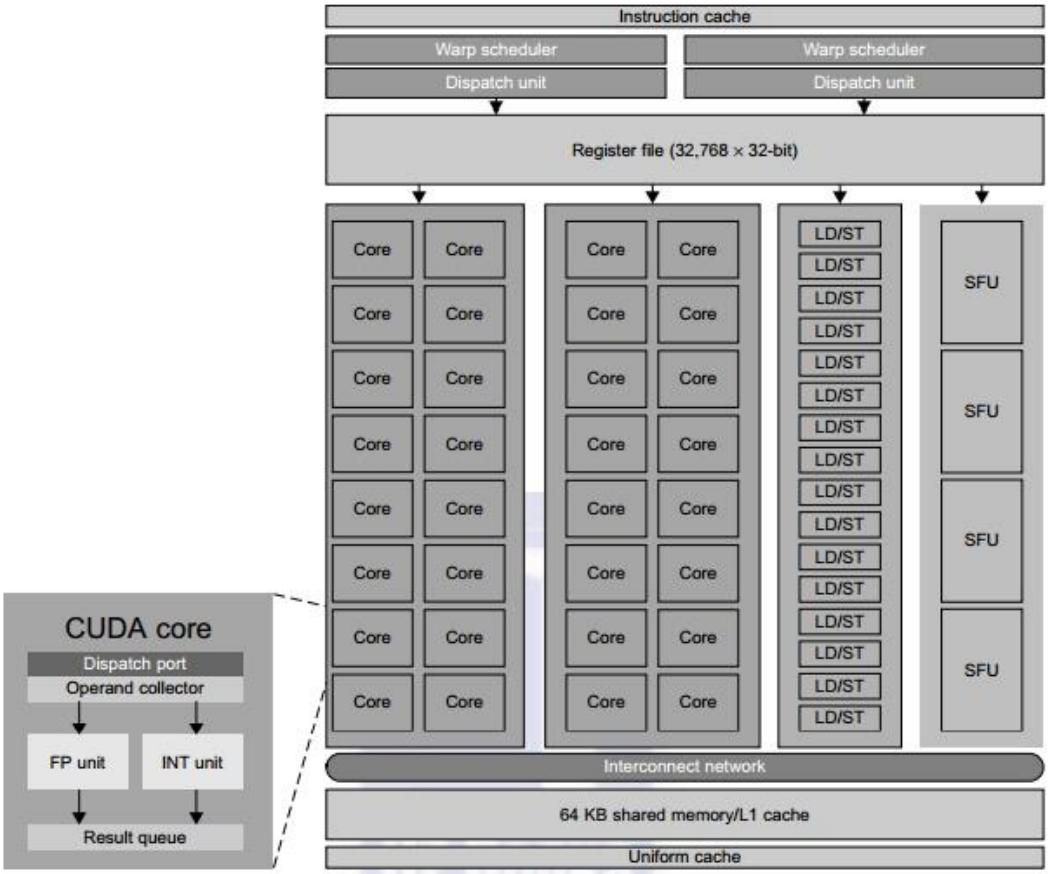


FIGURE 1.8

NVIDIA Fermi GPU built with 16 streaming multiprocessors (SMs) of 32 CUDA cores each; only one SM Is shown. More details can be found also in [49].

The Fermi GPU is a newer generation of GPU, first appearing in 2011. The Tesla or Fermi GPU can be used in desktop workstations to accelerate floating-point calculations or for building large-scale data cen-ters. The architecture shown is based on a 2009 white paper by NVIDIA. There are 32 CUDA cores per SM. Only one SM is shown in Figure 1.8.

Each CUDA core has a simple pipelined integer ALU and an FPU that can be used in parallel. Each SM has 16 load/store units allowing source and destination addresses to be calculated for 16 threads per clock. There are four special function units (SFUs) for executing transcendental instructions.

How GPUs Work

- Today, the NVIDIA GPU has been upgraded to 128 cores on a single chip. Furthermore, each core on a GPU can handle eight threads of instructions.
- This translates to having up to 1,024 threads executed concurrently on a single GPU.
- This is true massive parallelism, compared to only a few threads that can be handled by a conventional CPU.
- The CPU is optimized for latency caches, while the GPU is optimized to deliver much higher throughput with explicit management of on-chip memory.
- GPU offloads the CPU from all data-intensive calculations, not just those that are related to video processing.
- Conventional GPUs are widely used in mobile phones, game consoles, embedded systems, PCs, and servers.
- The NVIDIA CUDA Tesla or Fermi is used in GPU clusters or in HPC systems for parallel processing of massive floating-pointing data.

GPU Programming Model

- Four challenges are identified for exascale computing:
- (1) energy and power,
- (2) memory and storage,
- (3) concurrency and locality
- (4) system resiliency (system elasticity)

Power Efficiency of the GPU

- Power constrains what we can put in a CPU or GPU chip.
- Dally has estimated that the CPU chip consumes about 2 nJ/instruction, while the GPU chip requires 200 pJ/instruction, which is 1/10 less than that of the CPU.
- The CPU is optimized for latency in caches and memory, while the GPU is optimized for throughput with explicit management of on-chip memory.

3. Memory, Storage, and Wide-Area Networking

3.a. Memory Technology

- The capacity increase of disk arrays will be even greater in the years to come.
- Faster processor speed and larger memory capacity result in a wider gap between processors and memory.
- The memory wall may become even worse a problem limiting the CPU performance in the future.

3.b.Disks and Storage Technology

- Eventually, power consumption, cooling, and packaging will limit large system development.
- Power increases linearly with respect to clock frequency and quadratic ally with respect to voltage applied on chips.
- Clock rate cannot be increased indefinitely.
- Lowered voltage supplies are very much in demand.

3.c.System-Area Interconnects

- The nodes in small clusters are mostly interconnected by an Ethernet switch or a local area network (LAN).
- As Figure 1.11 shows, a LAN typically is used to connect client hosts to big servers.
- A storage area network (SAN) connects servers to network storage such as disk arrays.
- Network attached storage (NAS) connects client hosts directly to the disk arrays.
- All three types of networks often appear in a large cluster built with commercial network components.
- If no large distributed storage is shared, a small cluster could be built with a multiport Gigabit Ethernet switch plus copper cables to link the end machines.
- All three types of networks are commercially available.

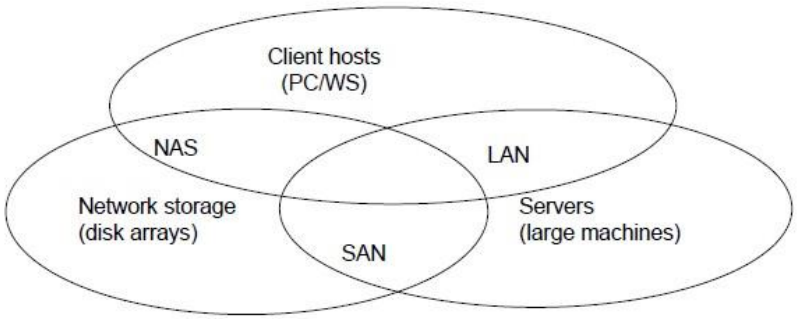


FIGURE 1.11

Three interconnection networks for connecting servers, client hosts, and storage devices; the LAN connects client hosts and servers, the SAN connects servers with disk arrays, and the NAS connects clients with large storage systems in the network environment.

3.d. Wide-Area Networking

- The lower curve in Figure 1.10 plots the rapid growth of Ethernet bandwidth from 10 Mbps in 1979 to 1 Gbps in 1999, and 40 ~ 100 GE in 2011.
- It has been speculated that 1 Tbps network links will become available by 2013.
- Network links with 1,000, 1,000, 100, 10, and 1 Gbps bandwidths were reported, respectively, for international, national, organization, optical desktop, and copper desktop connections in 2006.
- An increase factor of two per year on network performance was reported, which is faster than Moore's law on CPU speed doubling every 18 months.
- The implication is that more computers will be used concurrently in the future.
- High-bandwidth networking increases the capability of building massively distributed systems.
- Most data centers are using Gigabit Ethernet as the interconnect in their server clusters.

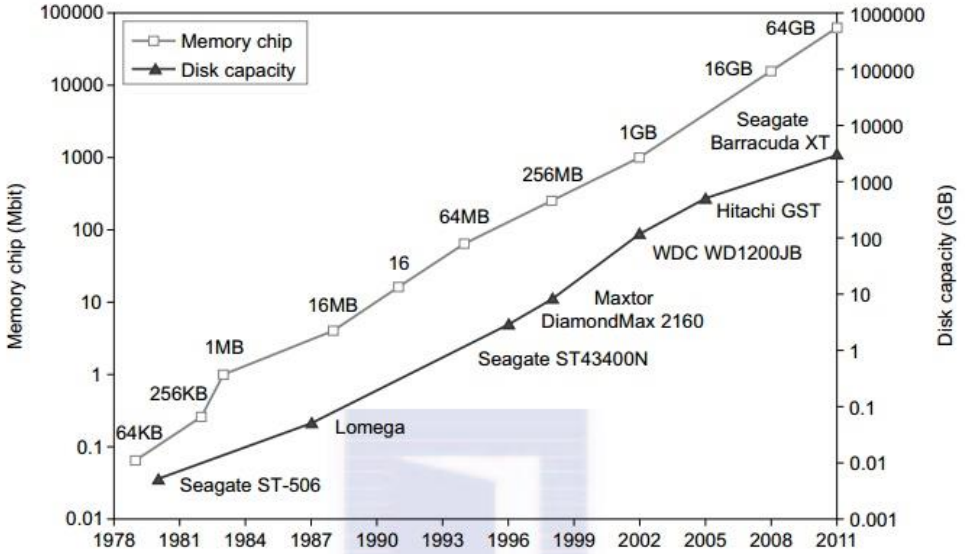


FIGURE 1.10

Improvement in memory and disk technologies over 33 years. The Seagate Barracuda XT disk has a capacity of 3 TB in 2011.

4.Virtual Machines and Virtualization Middleware

- A conventional computer has a single OS image. This offers a rigid architecture that tightly couples application software to a specific hardware platform.

- Some software running well on one machine may not be executable on another platform with a different instruction set under a fixed OS.

- Virtual machines (VMs) offer novel solutions to underutilized resources, application inflexibility, software manageability, and security concerns in existing physical machines.

- Today, to build large clusters, grids, and clouds, we need to access large amounts of computing, storage, and networking resources in a virtualized manner.

- We need to aggregate those resources, and hopefully, offer a single system image.

- In particular, a cloud of provisioned resources must rely on virtualization of processors, memory, and I/O facilities dynamically.

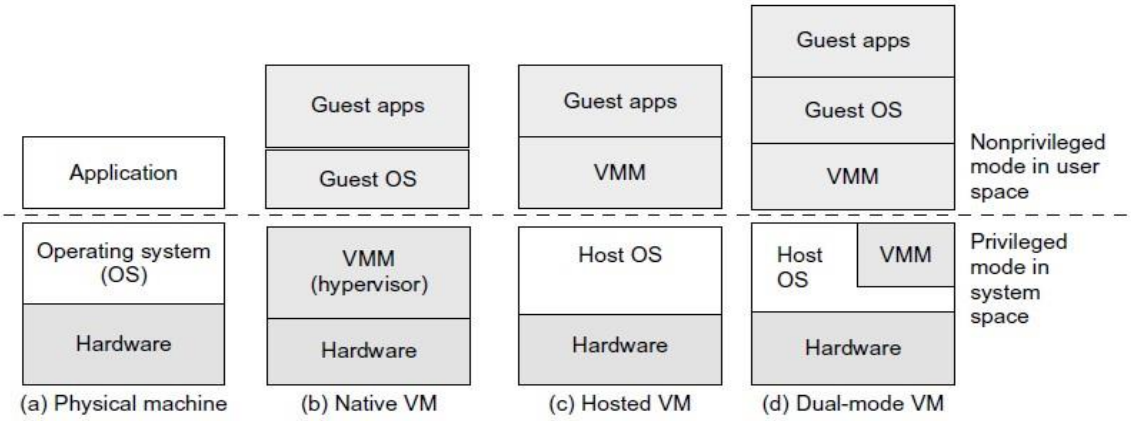


FIGURE 1.12

Three VM architectures in (b), (c), and (d), compared with the traditional physical machine shown in (a).

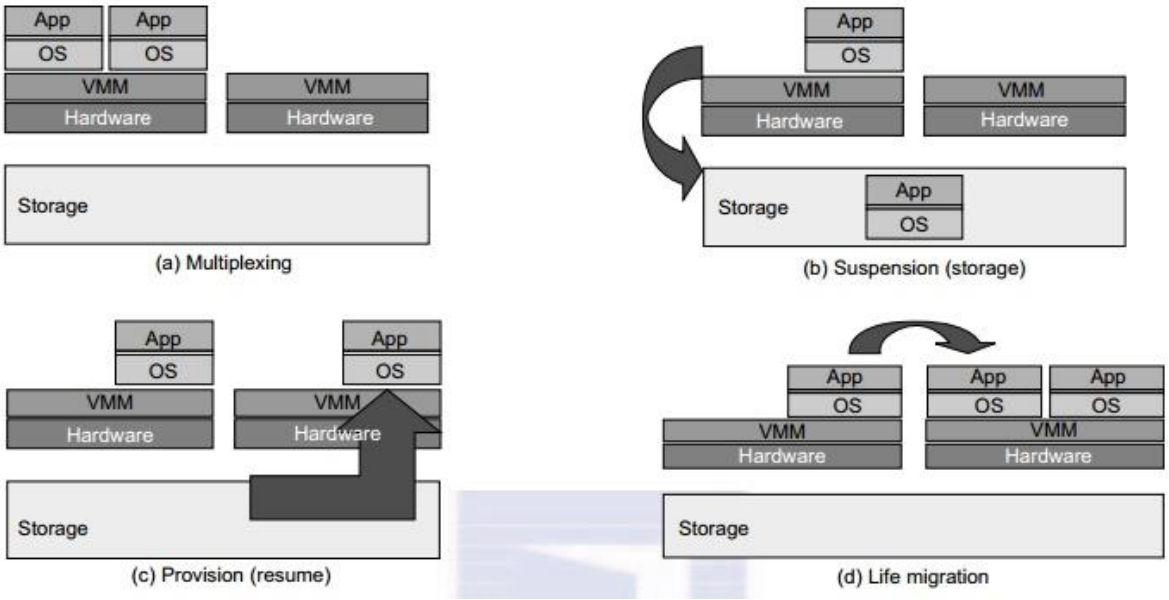
(Courtesy of M. Abde-Majeed and S. Kulkarni, 2009 USC)

4.a.Virtual Machines

- Virtualization is a computer architecture technology by which multiple virtual machines (VMs) are multiplexed in the same hardware machine.
- Virtual Machine (VM) is not a real machine, but it can be created from another physical machine.

- More than one VM may share the resources (CPU, Memory, Hard Disk & Network Resources) of the same physical machine.
- One physical machine may have more than one VM.
- VM used to create different execution environment which may entirely different from physical host execution environment.

4.b.VM Primitive Operations



- First, the VMs can be **multiplexed between hardware machines**, as shown in Figure 1.13(a).
- Second, a VM can be **suspended and stored in stable storage**, as shown in Figure 1.13(b).
- Third, a **suspended VM can be resumed or provisioned to a new hardware platform**, as shown in Figure 1.13(c).
- Finally, a VM can be migrated from one hardware platform to another, as shown in Figure 1.13(d).
- VM approach will significantly **enhance the utilization of server resources**.
- Multiple server functions can be consolidated on the same hardware platform to achieve higher system efficiency.

4.c. Virtual Infrastructures

- A virtual infrastructure is a software-based IT infrastructure being hosted on another physical infrastructure and meant to be distributed as a service as in cloud computing's infrastructure as a service (IaaS) delivery model.
- Virtual infrastructure is what connects resources to distributed applications.
- It is a dynamic mapping of system resources to specific applications.
- The result is decreased costs and increased efficiency and responsiveness.

5.Data Center Virtualization for Cloud Computing

Data Center Growth and Cost Breakdown

- A large data center may be built with thousands of servers.
- Smaller data centers are typically built with hundreds of servers.
- The cost to build and maintain data center servers has increased over the years.

Low-Cost Design Philosophy

- High-end switches or routers may be too cost-prohibitive for building data centers.
- Thus, using high-bandwidth networks may not fit the economics of cloud computing.

Convergence of Technologies

- Essentially, cloud computing is enabled by the convergence of technologies in four areas:-
  (1) Hardware virtualization and multi-core chips,
  (2) Utility and grid computing,
  (3) SOA, Web 2.0, and WS Mashups, and
  (4) Autonomic computing and data center automation.

System Models for Distributed and Cloud Computing

- *Clusters of Cooperative Computers*
- *Grid Computing Infrastructures*
- *Peer-to-Peer Network Families*
- *Cloud Computing over the Internet*

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

A.Clusters of Cooperative Computers

- A computing cluster consists of interconnected stand-alone computers which work cooperatively as a single integrated computing resource

Cluster Architecture

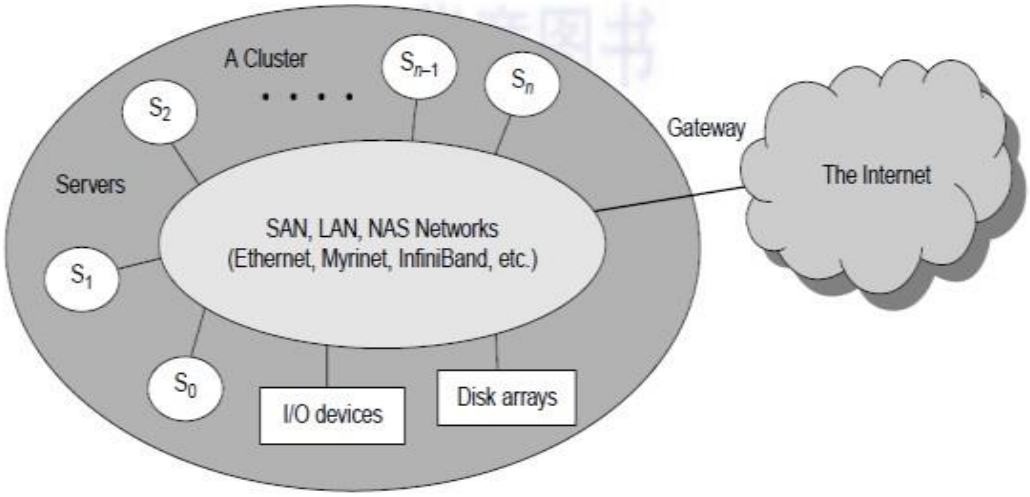


FIGURE 1.15

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

Single-System Image

The architecture of a typical server cluster built around a low-latency, high bandwidth interconnection network. This network can be as simple as a SAN (e.g., Myrinet) or a LAN (e.g., Ethernet)

- The cluster is connected to the Internet via a virtual private network (VPN) gateway. **The gateway IP address locates the cluster.**
- An ideal cluster should merge multiple system images into a **S**ingle **S**ystem **I**mage (SSI).
- SSI is an illusion created by software or hardware that presents a collection of resources as one integrated, powerful resource.
- SSI makes the cluster appear like a single machine to the user.

Table 1.3 Critical Cluster Design Issues and Feasible Implementations

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

Hardware, Software, and Middleware Support

- Special cluster middleware supports are needed to create SSI or **H**igh **A**vailability (HA).
- Both sequential and parallel applications can run on the cluster
- For example, distributed memory has multiple images. Users may want all distributed memory to be shared by all servers by forming **D**istributed **S**hared **M**emory (DSM).

Major Cluster Design Issues

- Middleware or OS extensions were developed at the user space to achieve SSI at selected functional levels.
- The cluster benefits come from scalable performance, efficient message passing, high system availability, seamless fault tolerance, and cluster-wide job management.

B.Grid Computing Infrastructures

- Internet services such as the Telnet command enables a local computer to connect to a remote computer.
- A web service such as HTTP enables remote access of remote web pages.
- Grid computing is envisioned to allow close interaction among applications running on distant computers simultaneously.

Computational Grids

- 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.
- 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.
- At the server end, the grid is a network. At the client end, we see wired or wireless terminal devices.
- The grid integrates the computing, communication, contents, and transactions as rented services.

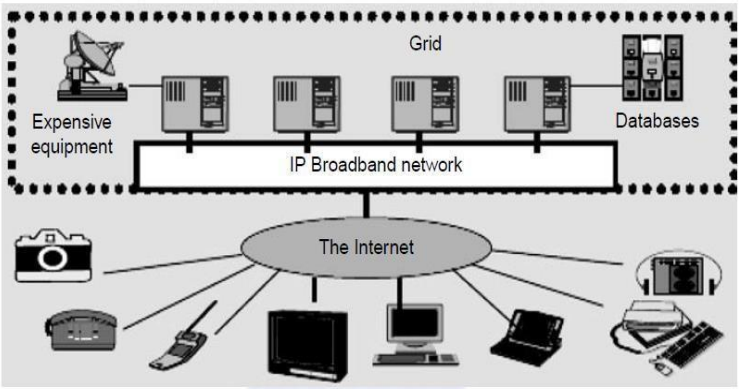


FIGURE 1.16

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

- Figure 1.16 shows an example computational grid built over multiple resource sites owned by different organizations.

- o 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.

- At the server end, the grid is a network. At the client end, we see wired or wireless terminal devices. The grid integrates t h e computing , communication , contents, and transactions

- Enterprises and consumers form the user base, which then defines the usage trends and service characteristics.

- The NSF TeraGrid in US, EGEE in Europe, and ChinaGrid in China for various distributed scientific grid applications.

Grid Families

Grid systems are classified in essentially two categories: -

- Computational or Data Grids and
- P2P Grids.

Computing or data grids are built primarily at the national level.

Table 1.4 Two Grid Computing Infrastructures and Representative Systems

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

C.Cloud Computing over the Internet

- 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

- The cloud **supports** redundant, self-recovering, highly scalable programming models that allow workloads to recover from many unavoidable hardware/software failures.

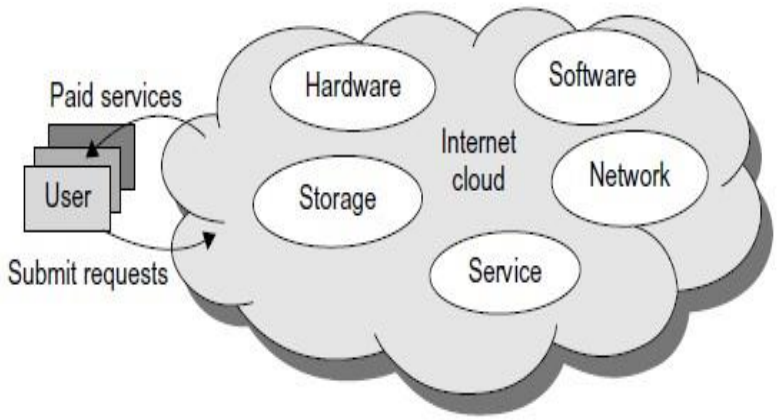
Internet Clouds

- 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.

The Cloud Landscape

- Traditional systems have encountered several performance bottlenecks: constant system maintenance, poor utilization, and increasing costs associated with hardware/software upgrades.
- Cloud computing as an on-demand computing paradigm resolves or relieves us from these problems.



1. 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 Operating Systems 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.

2. 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.**

3. 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.**

- Internet clouds offer **four deployment models: -**
    1) Private,
    2) Public,
    3) Community, and
    4) Hybrid

Reasons to adapt the cloud for upgraded Internet applications and web services:-

1. Desired location in areas with protected space and higher energy efficiency.

2. Sharing of peak-load capacity among a large pool of users, improving overall utilization.

3. Separation of infrastructure maintenance duties from domain-specific application development.

4. Significant reduction in cloud computing cost, compared with traditional computing paradigms.

5. Cloud computing programming and application development.

6. Service and data discovery and content/service distribution.

7. Privacy, security, copyright, and reliability issues.

8. Service agreements, business models, and pricing policies.

UNIT-III CLOUD INFRASTRUCTURE

4.3 ARCHITECTURAL DESIGN OF COMPUTE AND STORAGE CLOUDS

This section presents basic cloud design principles. We start with basic cloud architecture to process massive amounts of data with a high degree of parallelism. Then we study virtualization support, resource provisioning, infrastructure management, and performance modeling.

4.3.1 A Generic Cloud Architecture Design

An Internet cloud is envisioned as a public cluster of servers provisioned on demand to perform collective web services or distributed applications using data-center resources. In this section, we will discuss cloud design objectives and then present a basic cloud architecture design.

4.3.1.1 Cloud Platform Design Goals

Scalability, virtualization, efficiency, and reliability are four major design goals of a cloud computing platform. Clouds support Web 2.0 applications. Cloud management receives the user request, finds the correct resources, and then calls the provisioning services which invoke the resources in the cloud. The cloud management software needs to support both physical and virtual machines. Security in shared resources and shared access of data centers also pose another design challenge.

The platform needs to establish a very large-scale HPC infrastructure. The hardware and software systems are combined to make it easy and efficient to operate. System scalability can benefit from cluster architecture. If one service takes a lot of processing power, storage capacity, or network traffic, it is simple to add more servers and bandwidth. System reliability can benefit from this architecture. Data can be put into multiple locations. For example, user e-mail can be put in three disks which expand to different geographically separate data centers. In such a situation, even if one of the data centers crashes, the user data is still accessible. The scale of the cloud architecture can be easily expanded by adding more servers and enlarging the network connectivity accordingly.

4.3.1.2 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 Table 4.3.

Scanned with CamScanner

Table 4.3 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.

4.3.1.3 A Generic Cloud Architecture

Figure 4.14 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.

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

Scanned with CamScanner

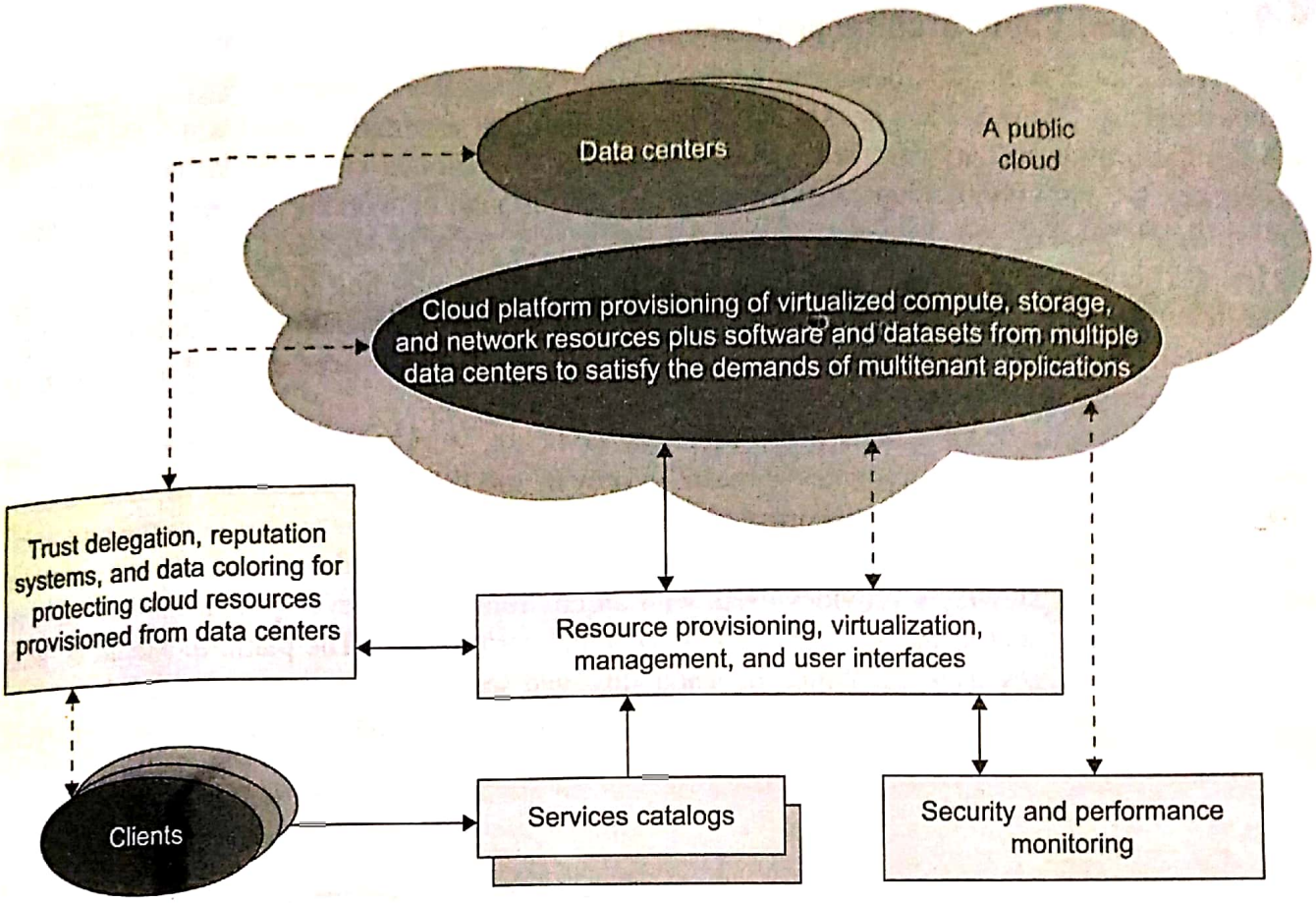


FIGURE 4.14

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

(Courtesy of K. Hwang and D. Li, 2010 [36])

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. We will study cloud security and privacy issues at the end of this chapter.

Scanned with CamScanner

4.3.2 Layered Cloud Architectural Development

The architecture of a cloud is developed at three layers: infrastructure, platform, and application, as demonstrated in Figure 4.15. 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.

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.

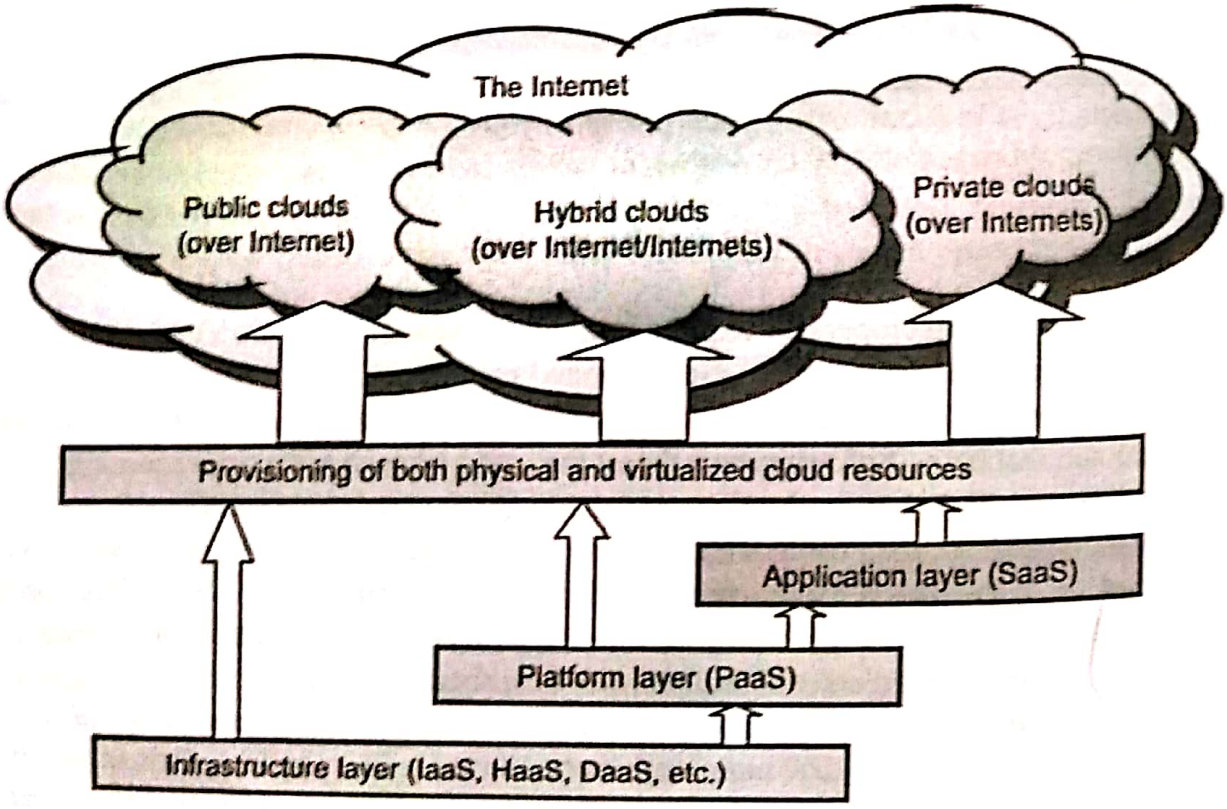


FIGURE 4.15

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

Scanned with CamScanner

(1) Market - oriented cloud Architecture :-

⇒ The cloud is basically built with the following entities :-

i) 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.

ii) SLA Resource allocator - The SLA resource allocator acts as the interface between the data center / cloud service provider & external users/ brokers.

Scanned with CamScanner

(iii) Request Examiner - The request examiner ensures that there is no overloading of resources whereby many service requests cannot be fulfilled successfully due to limited resources.

(iv) Service Request monitor - In order to make resource allocation decisions effectively. Then it assigns requests to VMs and determines resource entitlements for allocated VMs. Keeps track of the execution progress of service requests.

(V) Pricing :- Serves as a basis for managing the supply & demand of computing resources within the data center & facilitates in prioritizing resource allocations effectively.

(vi) Accounting mechanism :- maintains the actual usage of resources by requests so that the final cost can be computed & charged to users.

(vii) VM monitor mechanism :- keeps track of the availability of VMs and their resource entitlements.

(viii) Dispatcher mechanism starts the execution of accepted service requests on allocated VM's.

Scanned with CamScanner

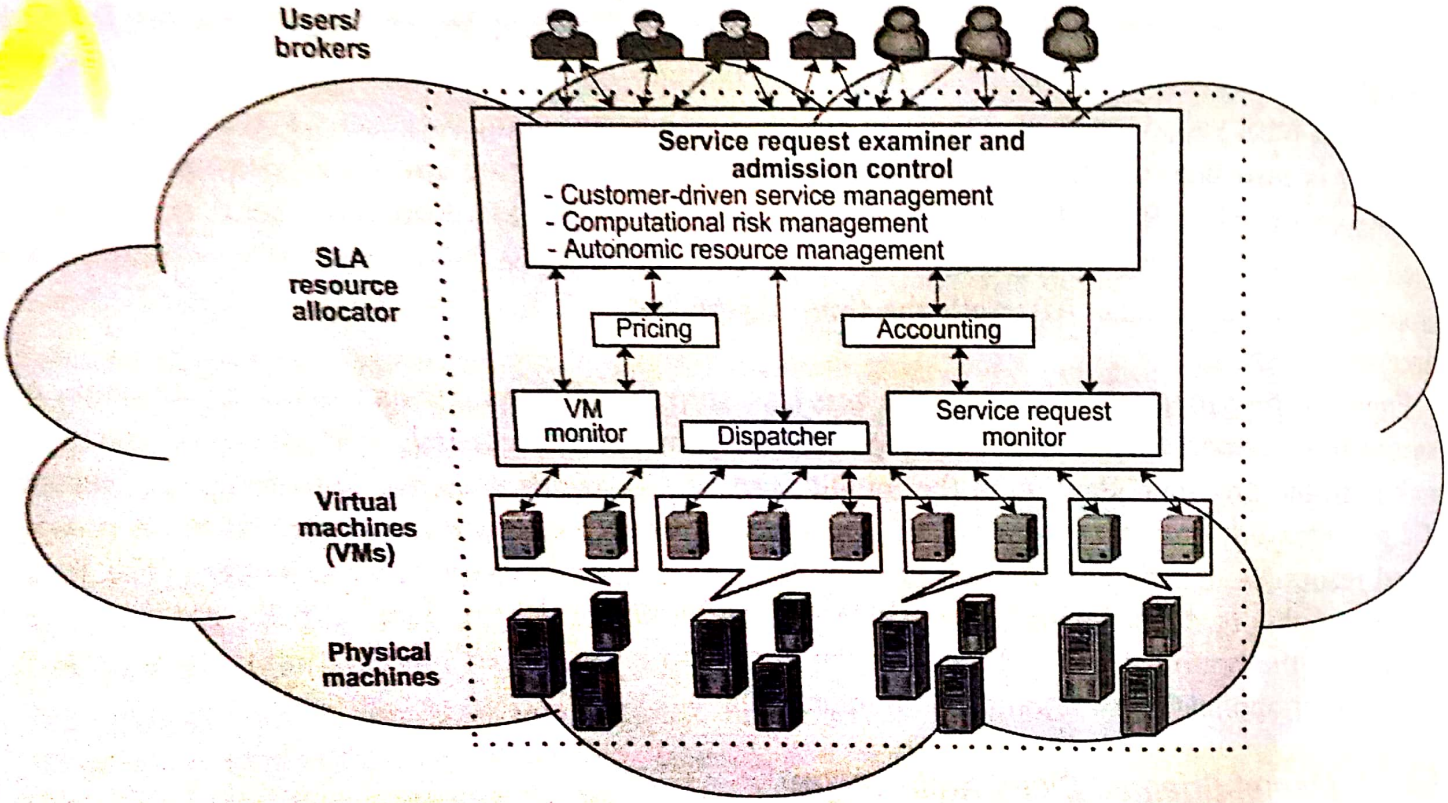


FIGURE 4.16

Market-oriented cloud architecture to expand/shrink leasing of resources with variation in QoS/demand from users.

(Courtesy of Rai Buyya, et al. [11])

Scanned with CamScanner

4.3.4 Architectural Design Challenges

In this section, we will identify six open challenges in cloud architecture development. Armbrust, et al. [4] have observed some of these topics as both obstacles and opportunities. Plausible solutions to meet these challenges are discussed shortly.

4.3.4.1 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.

SDU

4.3.4.2 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

Scanned with CamScanner

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. We will study all of these security and privacy problems on clouds in Section 4.5.

4.3.4.3 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. We will study performance issues in Chapter 8.

4.3.4.4 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.

4.3.4.5 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.

Scanned with CamScanner

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 hypervisor-agnostic 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.

4.3.4.6 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.

Scanned with CamScanner

4.5 INTER-CLOUD RESOURCE MANAGEMENT

This section characterizes the various cloud service models and their extensions. The cloud service trends are outlined. Cloud resource management and intercloud resource exchange schemes are reviewed. We will discuss the defense of cloud resources against network threats in Section 4.6.

Scanned with CamScanner

4.5.2 Resource Provisioning and Platform Deployment

The emergence of computing clouds suggests fundamental changes in software and hardware architecture. Cloud architecture puts more emphasis on the number of processor cores or VM instances. Parallelism is exploited at the cluster node level. In this section, we will discuss techniques to provision computer resources or VMs. Then we will talk about storage allocation schemes to interconnect distributed computing infrastructures by harnessing the VMs dynamically.

4.5.2.1 Provisioning of Compute Resources (VMs)

Providers supply cloud services by signing SLAs with end users. The SLAs must commit sufficient resources such as CPU, memory, and bandwidth that the user can use for a preset period. Underprovisioning of resources will lead to broken SLAs and penalties. Overprovisioning of resources will lead to resource underutilization, and consequently, a decrease in revenue for the provider. Deploying an autonomous system to efficiently provision resources to users is a challenging problem. The difficulty comes from the unpredictability of consumer demand, software and hardware failures, heterogeneity of services, power management, and conflicts in signed SLAs between consumers and service providers.

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.

Scanned with CamScanner

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. Power-efficient 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.

4.5.2.2 Resource Provisioning Methods

Figure 4.24 shows three cases of static cloud resource provisioning policies. In case (a), overprovisioning with the peak load causes heavy resource waste (shaded area). In case (b), underprovisioning (along the capacity line) of resources results in losses by both user and provider in that paid demand by the users (the shaded area above the capacity) is not served and wasted resources still exist for those demanded areas below the provisioned capacity. In case (c), the constant provisioning of resources with fixed capacity to a declining user demand could result in even worse resource waste. The user may give up the service by canceling the demand, resulting in reduced revenue for the provider. Both the user and provider may be losers in resource provisioning without elasticity.

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

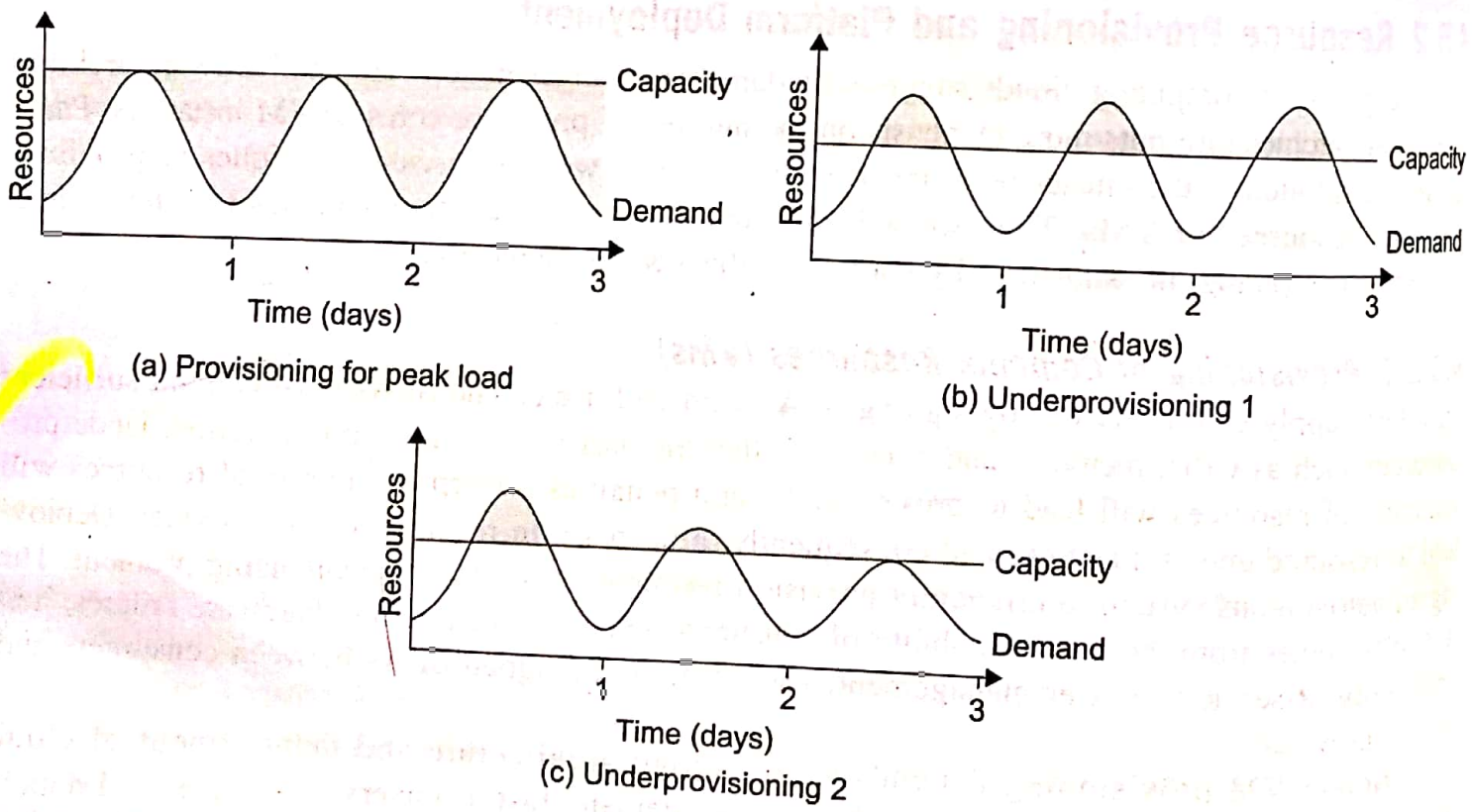


FIGURE 4.24

Three cases of cloud resource provisioning without elasticity: (a) heavy waste due to overprovisioning, (b) underprovisioning and (c) under- and then overprovisioning.

(Courtesy of Armbrust, et al., UC Berkeley, 2009 [4])

Scanned with CamScanner

on Internet traffic monitored. We characterize these resource provisioning methods as follows (see Figure 4.25).

4.5.2.3 Demand-Driven Resource Provisioning

This method adds or removes computing instances based on the current utilization level of the allocated resources. The demand-driven method automatically allocates two Xeon processors for the user application, when the user was using one Xeon processor more than 60 percent of the time for an extended period. In general, when a resource has surpassed a threshold for a certain amount of time, the scheme increases that resource based on demand. When a resource is below a threshold for a certain amount of time, that resource could be decreased accordingly. Amazon implements such an auto-scale feature in its EC2 platform. This method is easy to implement. The scheme does not work out right if the workload changes abruptly.

The x-axis in Figure 4.25 is the time scale in milliseconds. In the beginning, heavy fluctuations of CPU load are encountered. All three methods have demanded a few VM instances initially. Gradually, the utilization rate becomes more stabilized with a maximum of 20 VMs (100 percent utilization) provided for demand-driven provisioning in Figure 4.25(a). However, the event-driven method reaches a stable peak of 17 VMs toward the end of the event and drops quickly in Figure 4.25(b). The popularity provisioning shown in Figure 4.25(c) leads to a similar fluctuation with peak VM utilization in the middle of the plot.

4.5.2.4 Event-Driven Resource Provisioning

This scheme adds or removes machine instances based on a specific time event. The scheme works better for seasonal or predicted events such as Christmastime in the West and the Lunar New Year in the East. During these events, the number of users grows before the event period and then decreases during the event period. This scheme anticipates peak traffic before it happens. The method results in a minimal loss of QoS, if the event is predicted correctly. Otherwise, wasted resources are even greater due to events that do not follow a fixed pattern.

4.5.2.5 Popularity-Driven Resource Provisioning

In this method, the Internet searches for popularity of certain applications and creates the instances by popularity demand. The scheme anticipates increased traffic with popularity. Again, the scheme has a minimal loss of QoS, if the predicted popularity is correct. Resources may be wasted if traffic does not occur as expected. In Figure 4.25(c), EC2 performance by CPU utilization rate (the dark curve with the percentage scale shown on the left) is plotted against the number of VMs provisioned (the light curves with scale shown on the right, with a maximum of 20 VMs provisioned).

4.5.2.6 Dynamic Resource Deployment

The cloud uses VMs as building blocks to create an execution environment across multiple resource sites. The InterGrid-managed infrastructure was developed by a Melbourne University group [19]. Dynamic resource deployment can be implemented to achieve scalability in performance. The InterGrid is a Java-implemented software system that lets users create execution cloud environments on top of all participating grid resources. Peering arrangements established between gateways enable the allocation of resources from multiple grids to establish the execution environment. In Figure 4.26, a scenario is illustrated by which an *intergrid gateway (IGG)* allocates resources from a local cluster to

Scanned with CamScanner

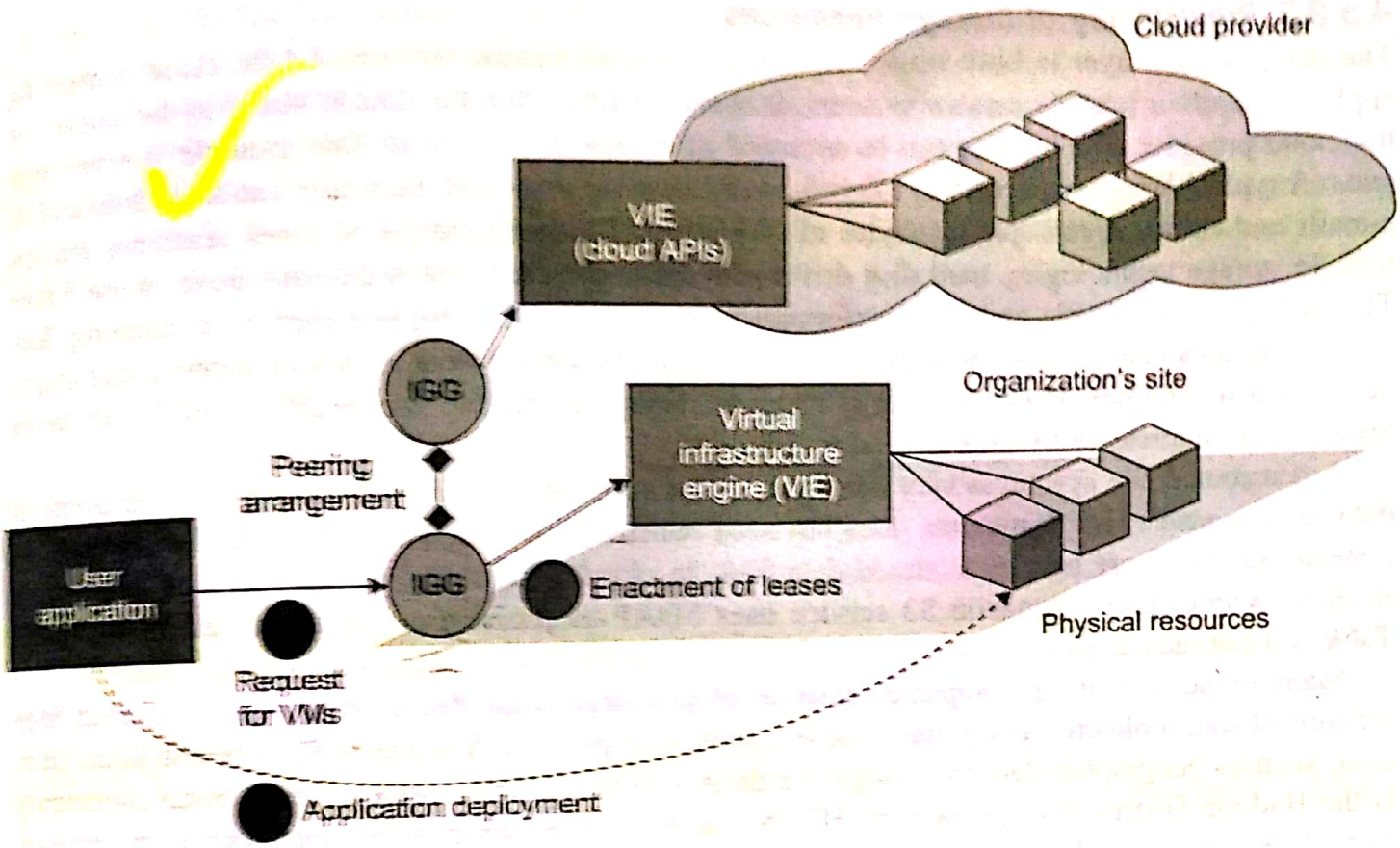


FIGURE 4.26

Cloud resource deployment using an IGG (intergrid gateway) to allocate the VMs from a Local cluster to interact with the IGG of a public cloud provider.

(*Courtesy of Constanzo, et al. [21]*)

deploy applications in three steps: (1) requesting the VMs, (2) enacting the leases, and (3) deploying the VMs as requested. Under peak demand, this IGG interacts with another IGG that can allocate resources from a cloud computing provider.

A grid has predefined peering arrangements with other grids, which the IGG manages. Through multiple IGGs, the system coordinates the use of InterGrid resources. An IGG is aware of the peering terms with other grids, selects suitable grids that can provide the required resources, and replies to requests from other IGGs. Request redirection policies determine which peering grid InterGrid selects to process a request and a price for which that grid will perform the task. An IGG can also allocate resources from a cloud provider. The cloud system creates a virtual environment to help users deploy their applications. These applications use the distributed grid resources.

The InterGrid allocates and provides a *distributed virtual environment* (DVE). This is a virtual cluster of VMs that runs isolated from other virtual clusters. A component called the *DVE manager* performs resource allocation and management on behalf of specific user applications. The core component of the IGG is a scheduler for implementing provisioning policies and peering with other gateways. The communication component provides an asynchronous message-passing mechanism. Received messages are handled in parallel by a thread pool.

Scanned with CamScanner

P R D EPD P

4.5.2.7 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 each 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. Table 4.8 outlines three cloud storage services provided by Google, Hadoop, and Amazon.

Many cloud computing companies have developed large-scale data storage systems to keep huge amount of data collected every day. For example, Google's GFS stores web data and some other data, such as geographic data for Google Earth. A similar system from the open source community is the Hadoop Distributed File System (HDFS) for Apache. Hadoop is the open source implementation of Google's cloud computing infrastructure. Similar systems include Microsoft's Cosmos file system for the cloud.

Despite the fact that the storage service or distributed file system can be accessed directly, similar to traditional databases, cloud computing does provide some forms of structure or semistructure database processing capability. For example, applications might want to process the information contained in a web page. Web pages are an example of semistructural data in HTML format. If some forms of database capability can be used, application developers will construct their application logic more easily. Another reason to build a database-like service in cloud computing is that it will be quite convenient for traditional application developers to code for the cloud platform. Databases are quite common as the underlying storage device for many applications.

Thus, such developers can think in the same way they do for traditional software development. Hence, in cloud computing, it is necessary to build databases like large-scale systems based on data

Table 4.8 Storage Services in Three Cloud Computing Systems

Storage System	Features
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.

Scanned with CamScanner

4) Global Exchange of cloud Resources :-

⇒ To support a large number of application service consumers from around the world, cloud Infrastructure providers (i.e) (Iaas providers) have established data centers in multiple geographical locations to provide redundancy and ensure reliability in case of site failures.

⇒ for eg :- Amazon data centers, however currently Amazon expects it cloud customers (i.e) Saas providers) to express a preference regarding where they want their application services to be hosted.

⇒ First, it is difficult for cloud customers to determine in advance the best location for hosting their services as they may not know the origin of consumers of their services.

Scanned with CamScanner

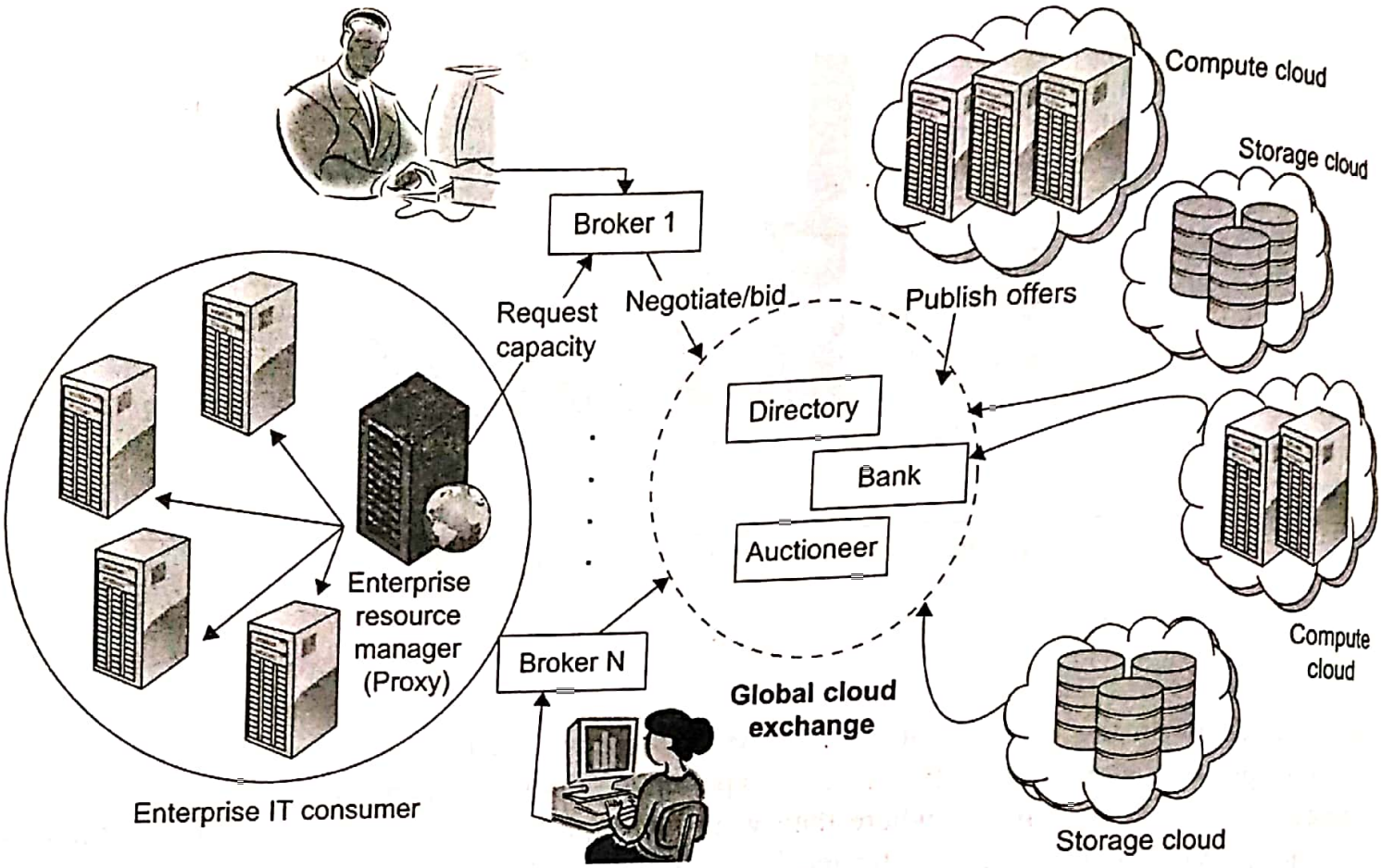


FIGURE 4.30

Inter-cloud exchange of cloud resources through brokering.

(Courtesy of R. Buyya, et al., University of Melbourne [12])

service providers; operate as part of a market-driven resource leasing federation, where application service providers such as Salesforce.com host their services based on negotiated SLA contracts driven by competitive market prices; and deliver on-demand, reliable, cost-effective, and QoS-aware services based on virtualization technologies while ensuring high QoS standards and minimizing service costs. They need to be able to utilize market-based utility models as the basis for provisioning of virtualized software services and federated hardware infrastructure among users with heterogeneous applications.

They consist of client brokering and coordinator services that support utility-driven federation of clouds: application scheduling, resource allocation, and migration of workloads. The architecture cohesively couples the administratively and topologically distributed storage and compute capabilities of clouds as part of a single resource leasing abstraction. The system will ease the cross-domain capability integration for on-demand, flexible, energy-efficient, and reliable access to the infrastructure based on virtualization technology [6,75].

The *Cloud Exchange (CEx)* acts as a market maker for bringing together service producers and consumers. It aggregates the infrastructure demands from application brokers and evaluates them against the available supply currently published by the cloud coordinators. It supports trading of cloud services based on competitive economic models such as commodity markets and auctions. CEx allows participants to locate providers and consumers with fitting offers. Such markets enable

Scanned with CamScanner

6.2.1 Parallel Computing and Programming Paradigms

Consider a distributed computing system consisting of a set of networked nodes or workers. The system issues for running a typical parallel program in either a parallel or a distributed manner would include the following [36–39]:

- 1 **Partitioning** This is applicable to both computation and data as follows:
- 2 **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.
- 3 **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.
- 4 **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.
- 5 **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.
- 6 **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.
- 7 **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.

Scanned with CamScanner

Motivations :

⇒ Motivations behind parallel & distributed programming models are :

(i) To improve productivity of programmers

(ii) To decrease programs time to market

(iii) to leverage underlying resources more efficiently

(iv) to increase system throughput

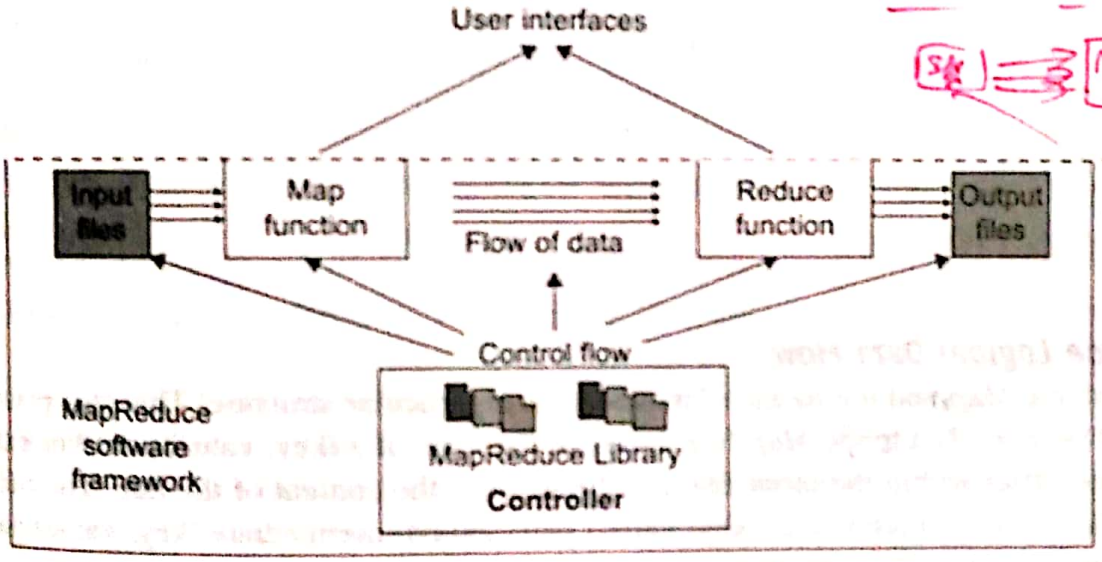
(v) to support higher levels of abstraction.

⇒ eg :- Mapreduce, Hadoop, Dryad.

Scanned with CamScanner

6.2.2 MapReduce, Twister, and Iterative MapReduce

MapReduce, as introduced in Section 6.1.4, is a software framework which supports parallel and distributed computing on large data sets [27,37,45,46]. This software framework abstracts the data flow of running a parallel program on a distributed computing system by providing users with two interfaces in the form of two functions: *Map* and *Reduce*. Users can override these two functions to interact with and manipulate the data flow of running their programs. Figure 6.1 illustrates the logical data flow from the *Map* to the *Reduce* function in MapReduce frameworks. In this framework,



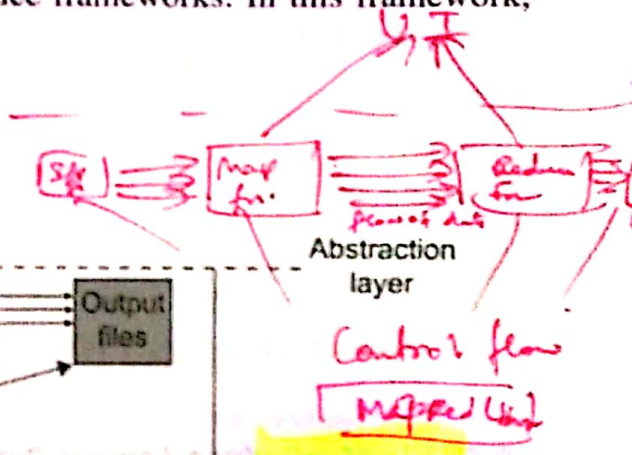


FIGURE 6.1

MapReduce framework: Input data flows through the Map and Reduce functions to generate the output result under the control flow using MapReduce software library. Special user interfaces are used to access the Map and Reduce resources.

Scanned with CamScanner

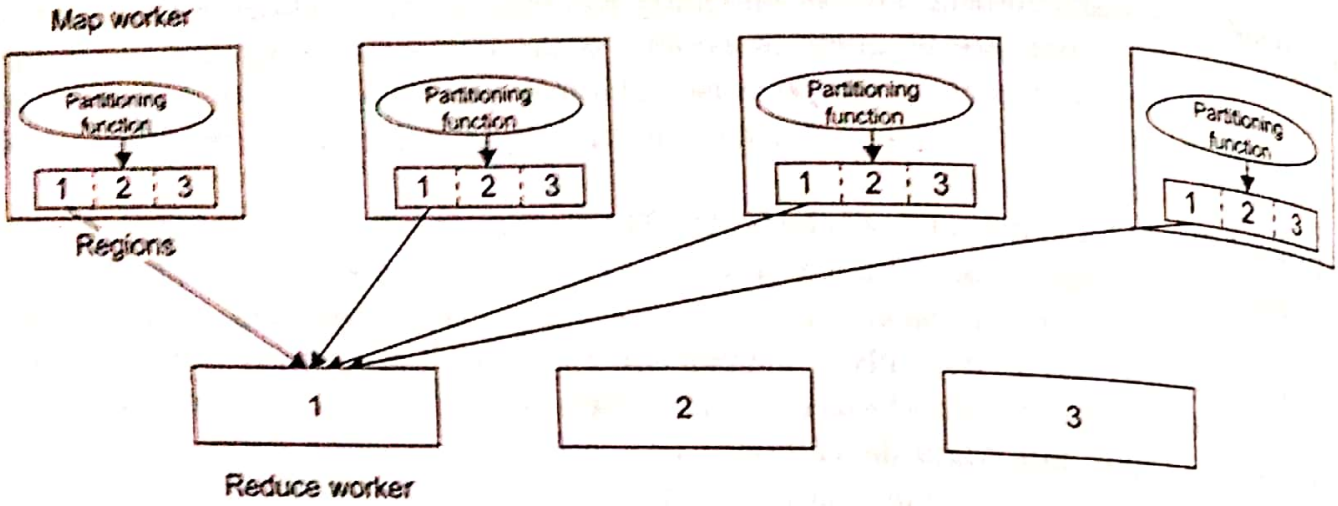


FIGURE 6.4

Use of MapReduce *partitioning* function to link the Map and Reduce workers.

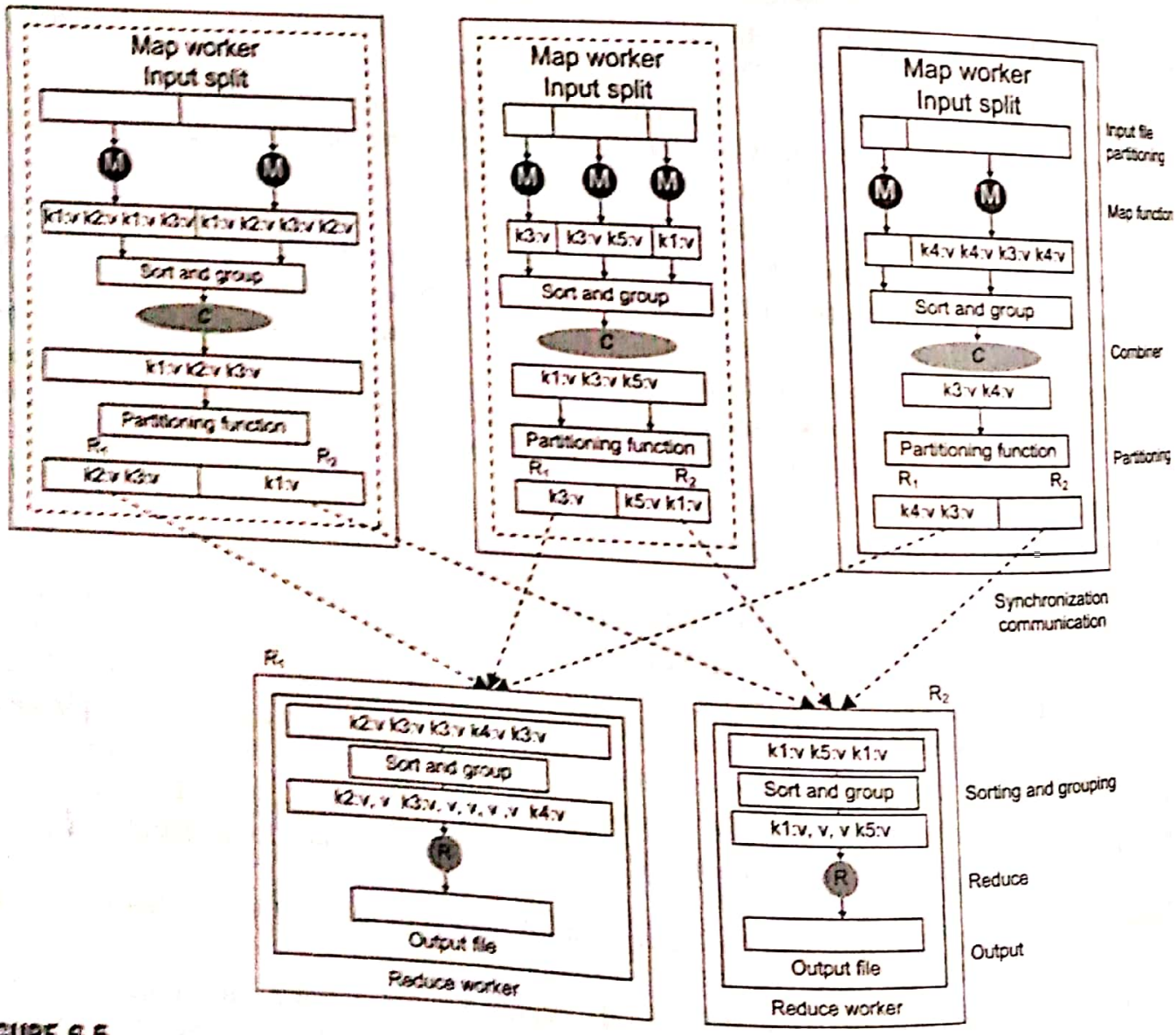


FIGURE 6.5

Data flow implementation of many functions in the Map workers and in the Reduce workers through multiple sequences of partitioning, combining, synchronization and communication, sorting and grouping, and reduce operations.

Scanned with CamScanner

6.2.2.7 Twister and Iterative MapReduce

It is important to understand the performance of different runtimes and, in particular, to compare MPI and MapReduce [43,44,55,56]. The two major sources of parallel overhead are load imbalance and communication (which is equivalent to synchronization overhead as communication synchronizes parallel units [threads or processes] in Categories 2 and 6 of Table 6.10). 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.

Scanned with CamScanner

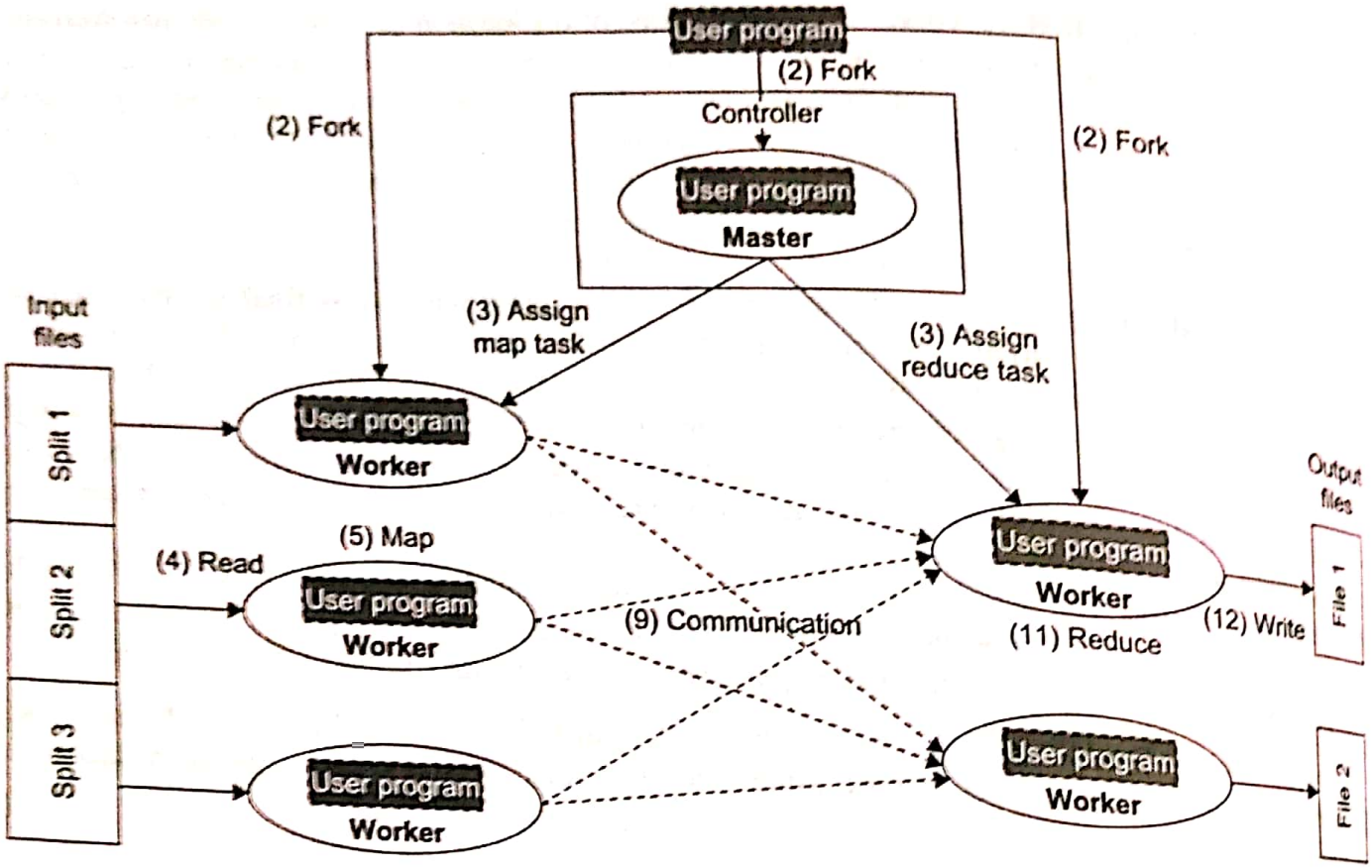


FIGURE 6.6

Control flow implementation of the MapReduce functionalities in Map workers and Reduce workers (running user programs) from input files to the output files under the control of the master user program.

(*Courtesy of Yahoo! Pig Tutorial [54]*)

- 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 [42] has been investigated in several projects [34,57–59] while the direct idea of using MPI for MapReduce applications is investigated in [44]. The Twister programming paradigm and its implementation architecture at run time are illustrated in Figure 6.7(a, b). In Example 6.1, we summarize Twister [60] whose performance results for K means are shown in Figure 6.8 [55,56], where Twister is much faster than traditional MapReduce. Twister distinguishes the static data which is never reloaded from the dynamic δ flow that is communicated.

Scanned with CamScanner

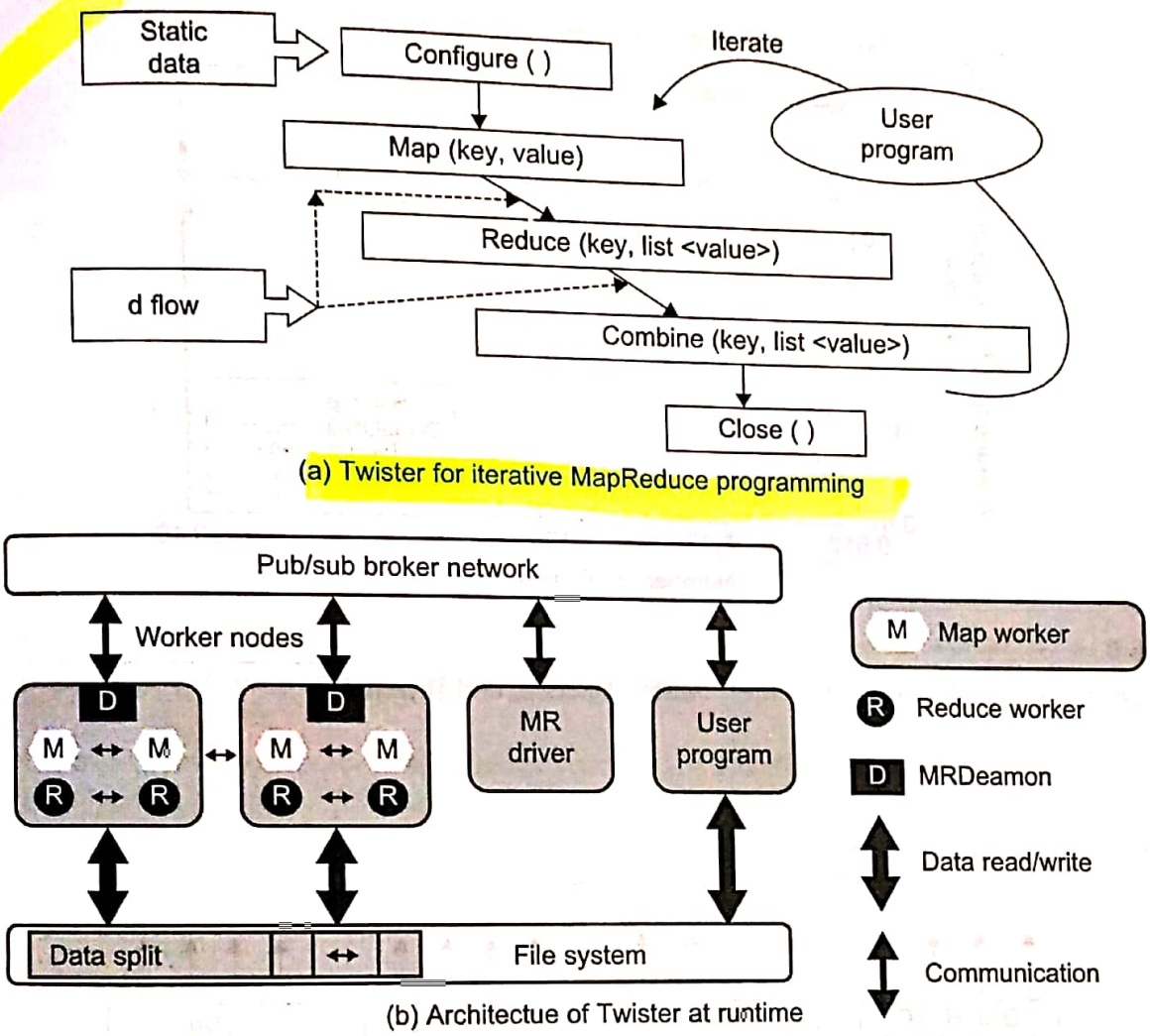


FIGURE 6.7

Twister: An iterative MapReduce programming paradigm for repeated MapReduce executions.

Example 6.1 Performance of K Means Clustering in MPI, Twister, Hadoop, and DryadLINQ

The MapReduce approach leads to fault tolerance and flexible scheduling, but for some applications the performance degradation compared to MPI is serious, as illustrated in Figure 6.8 for a simple parallel K means clustering algorithm. Hadoop and DryadLINQ are more than a factor of 10 slower than MPI for the largest data set, and perform even more poorly for smaller data sets. One could use many communication mechanisms in Iterative MapReduce, but Twister chose a publish-subscribe network using a distributed set of brokers, as described in Section 5.2 with similar performance achieved with ActiveMQ and NaradaBrokering.

The Map-Reduce pair is iteratively executed in long-running threads. We compare in Figure 6.9. the different thread and process structures of 4 parallel programming paradigms: namely Hadoop, Dryad, Twister (also called MapReduce++), and MPI. Note that Dryad can use pipes and avoids costly disk writing according to the original papers [26,27].

Scanned with CamScanner

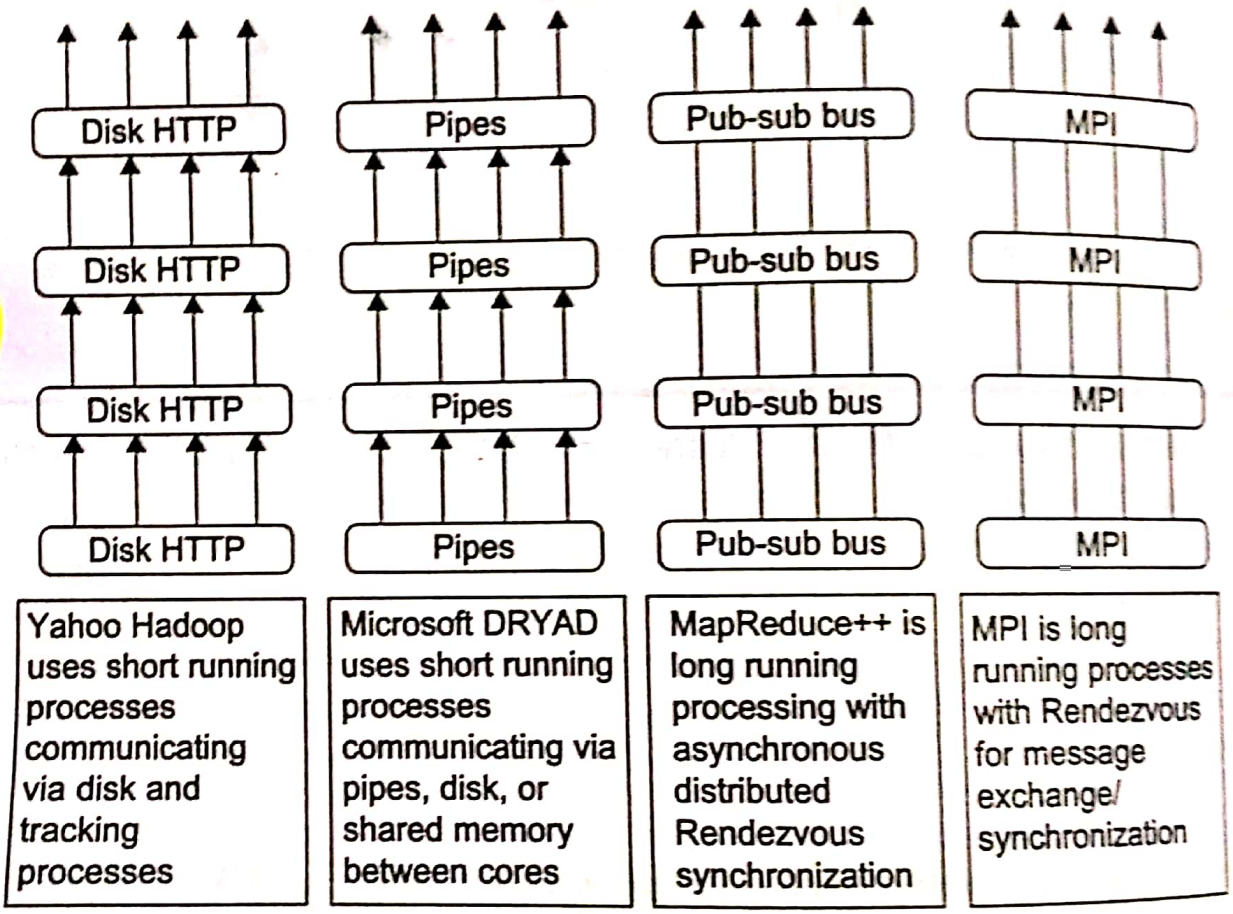


FIGURE 6.9

Thread and process structure of four parallel programming paradigms at runtimes.

Scanned with CamScanner

6.2.3 Hadoop Library from Apache

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

HDFS: HDFS is a distributed file system inspired by GFS that organizes files and stores their data on a distributed computing system.

HDFS Architecture: HDFS has a master/slave architecture containing a single NameNode as the master and a number of DataNodes as workers (slaves). To store a file in this architecture, HDFS splits the file into fixed-size blocks (e.g., 64 MB) and stores them on workers (DataNodes). The mapping of blocks to DataNodes is determined by the NameNode. The NameNode (master) also manages the file system's metadata and namespace. In such systems, the namespace is the area maintaining the metadata, and metadata refers to all the information stored by a file system that is needed for overall management of all files. For example, NameNode in the metadata stores all information regarding the location of input splits/blocks in

Scanned with CamScanner

all DataNodes. Each DataNode, usually one per node in a cluster, manages the storage attached to the node. Each DataNode is responsible for storing and retrieving its file blocks [61].

HDFS Features: Distributed file systems have special requirements, such as performance, scalability, concurrency control, fault tolerance, and security requirements [62], to operate efficiently. However, because HDFS is not a general-purpose file system, as it only executes specific types of applications, it does not need all the requirements of a general distributed file system. For example, security has never been supported for HDFS systems. The following discussion highlights two important characteristics of HDFS to distinguish it from other generic distributed file systems [63].

HDFS Fault Tolerance: One of the main aspects of HDFS is its fault tolerance characteristic. Since Hadoop is designed to be deployed on low-cost hardware by default, a hardware failure in this system is considered to be common rather than an exception. Therefore, Hadoop considers the following issues to fulfill reliability requirements of the file system [64]:

- **Block replication** To reliably store data in HDFS, file blocks are replicated in this system. In other words, HDFS stores a file as a set of blocks and each block is replicated and distributed across the whole cluster. The replication factor is set by the user and is three by default.
- **Replica placement** The placement of replicas is another factor to fulfill the desired fault tolerance in HDFS. Although storing replicas on different nodes (DataNodes) located in different racks across the whole cluster provides more reliability, it is sometimes ignored as the cost of communication between two nodes in different racks is relatively high in comparison with that of different nodes located in the same rack. Therefore, sometimes HDFS compromises its reliability to achieve lower communication costs. For example, for the default replication factor of three, HDFS stores one replica in the same node the original data is stored, one replica on a different node but in the same rack, and one replica on a different node in a different rack to provide three copies of the data [65].
- **Heartbeat and Blockreport messages** Heartbeats and Blockreports are periodic messages sent to the NameNode by each DataNode in a cluster. Receipt of a Heartbeat implies that the DataNode is functioning properly, while each Blockreport contains a list of all blocks on a DataNode [65]. The NameNode receives such messages because it is the sole decision maker of all replicas in the system.

HDFS High-Throughput Access to Large Data Sets (Files): Because HDFS is primarily designed for batch processing rather than interactive processing, data access throughput in HDFS is more important than latency. Also, because applications run on HDFS typically have large data sets, individual files are broken into large blocks (e.g., 64 MB) to allow HDFS to decrease the amount of metadata storage required per file. This provides two advantages: The list of blocks per file will shrink as the size of individual blocks increases, and by keeping large amounts of data sequentially within a block, HDFS provides fast streaming reads of data.

HDFS Operation: 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 [63].

Scanned with CamScanner

- **Reading a file** To read a file in HDFS, a user sends an "open" request to the NameNode to get the location of file blocks. For each file block, the NameNode returns the address of a set of DataNodes containing replica information for the requested file. The number of addresses depends on the number of block replicas. Upon receiving such information, the user calls the *read* function to connect to the closest DataNode containing the first block of the file. After the first block is streamed from the respective DataNode to the user, the established connection is terminated and the same process is repeated for all blocks of the requested file until the whole file is streamed to the user.
- **Writing to a file** To write a file in HDFS, a user sends a "create" request to the NameNode to create a new file in the file system namespace. If the file does not exist, the NameNode notifies the user and allows him to start writing data to the file by calling the *write* function. The first block of the file is written to an internal queue termed the data queue while a data streamer monitors its writing into a DataNode. Since each file block needs to be replicated by a predefined factor, the data streamer first sends a request to the NameNode to get a list of suitable DataNodes to store replicas of the first block.

The steamer then stores the block in the first allocated DataNode. Afterward, the block is forwarded to the second DataNode by the first DataNode. The process continues until all allocated DataNodes receive a replica of the first block from the previous DataNode. Once this replication process is finalized, the same process starts for the second block and continues until all blocks of the file are stored and replicated on the file system.

Scanned with CamScanner

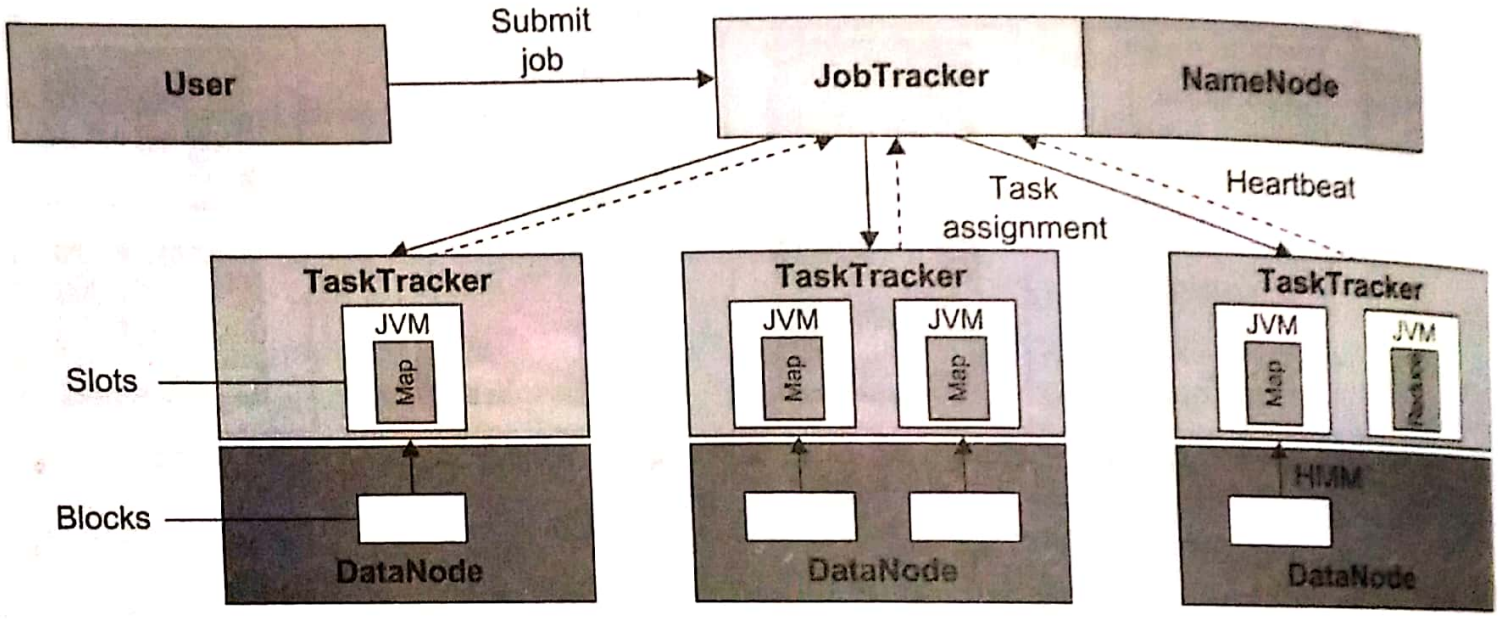


FIGURE 6.12

Data flow in running a MapReduce job at various task trackers using the Hadoop library.

Scanned with CamScanner

6.2.6 Mapping Applications to Parallel and Distributed Systems

In the past, Fox has discussed mapping applications to different hardware and software in terms of five application architectures [71]. These initial five categories are listed in Table 6.10, followed by a sixth emerging category to describe data-intensive computing [72,73]. The original classifications largely described simulations and were not aimed directly at data analysis. It is instructive to briefly summarize them and then explain the new category. Category 1 was popular 20 years ago, but is no longer significant. It describes applications that can be parallelized with lock-step operations controlled by hardware.

Such a configuration would run on SIMD (*single-instruction and multiple-data*) machines, whereas category 2 is now much more important and corresponds to a SPMD (*single-program and multiple-data*) model running on MIMD (*multiple instruction multiple data*) machines. Here, each decomposed

Scanned with CamScanner

Table 6.10 Application Classification for Parallel and Distributed Systems

Category	Class	Description	Machine Architecture
1	Synchronous	The problem class can be implemented with instruction-level lockstep operation as in SIMD architectures.	SIMD
2	Loosely synchronous (BSP or bulk synchronous processing)	These problems exhibit iterative compute-communication stages with independent compute (map) operations for each CPU that are synchronized with a communication step. This problem class covers many successful MPI applications including partial differential equation solutions and particle dynamics applications.	MIMD on MPP (massively parallel processor)
3	Asynchronous	Illustrated by Compute Chess and Integer Programming; combinatorial search is often supported by dynamic threads. This is rarely important in scientific computing, but it is at the heart of operating systems and concurrency in consumer applications such as Microsoft Word.	Shared memory
4	Pleasingly parallel	Each component is independent. In 1988, Fox estimated this at 20 percent of the total number of applications, but that percentage has grown with the use of grids and data analysis applications including, for example, the Large Hadron Collider analysis for particle physics.	Grids moving to clouds
5	Metaproblems	These are coarse-grained (asynchronous or data flow) combinations of categories 1-4 and 6. This area has also grown in importance and is well supported by grids and described by workflow in Section 3.5.	Grids of clusters
6	MapReduce++ (Twister)	This describes file (database) to file (database) operations which have three subcategories (see also Table 6.11): 6a) Pleasingly Parallel Map Only (similar to category 4) 6b) Map followed by reductions 6c) Iterative "Map followed by reductions" (extension of current technologies that supports linear algebra and data mining)	Data-intensive clouds a) Master-worker or MapReduce b) MapReduce c) Twister

unit executes the same program, but at any given time, there is no requirement that the same instruction be executed. Category 1 corresponds to regular problems, whereas category 2 includes dynamic irregular cases with complex geometries for solving partial differential equations or particle dynamics. Note that synchronous problems are still around, but they are run on MIMD machines with the SPMD model. Also note that category 2 consists of compute–communicate phases and the computations are synchronized by communication. No additional synchronization is needed.

Category 3 consists of asynchronously interacting objects and is often considered the people's view of a typical parallel problem. It probably does describe the concurrent threads in a modern operating system, as well as some important applications, such as event-driven simulations and areas such

Scanned with CamScanner

as search in computer games and graph algorithms. Shared memory is natural due to the low latency often needed to perform dynamic synchronization. It wasn't clear in the past, but now it appears that this category is not very common in large-scale parallel problems of importance.

Category 4 is the simplest algorithmically, with disconnected parallel components. However, the importance of this category has probably grown since the original 1988 analysis when it was estimated to account for 20 percent of all parallel computing. Both grids and clouds are very natural for this class, which does not need high-performance communication between different nodes.

Category 5 refers to the coarse-grained linkage of different "atomic" problems, and this was fully covered in Section 5.5. This area is clearly common and is expected to grow in importance. Remember the critical observation in Section 5.5 that we use a two-level programming model with the metaproblem (workflow) linkage specified in one fashion and the component problems with approaches such as those in this chapter. Grids or clouds are suitable for metaproblems as coarse-grained decomposition does not usually require stringent performance.

As noted earlier, we added a sixth category to cover data-intensive applications motivated by the clear importance of MapReduce as a new programming model. We call this category MapReduce++, and it has three subcategories: "map only" applications similar to pleasingly parallel category 4 [41,43,74,75]; the classic MapReduce with file-to-file operations consisting of parallel maps followed by parallel reduce operations; and a subcategory that captures the extended MapReduce version covered in Section 6.2.2. Note that category 6 is a subset of categories 2 and 4 with additional reading and writing of data and a specialized, loosely synchronous structure compared to that used in data analysis. This comparison is made clearer in Table 6.11.

6.3 PROGRAMMING SUPPORT OF GOOGLE APP ENGINE

Section 4.4.2 introduced the Google App Engine infrastructure in Figures 4.20. In this section, we describe the programming model supported by GAE. The access links to the GAE platform are provided in Chapter 4.

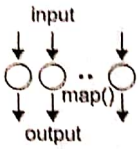
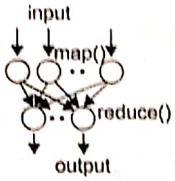
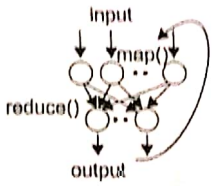
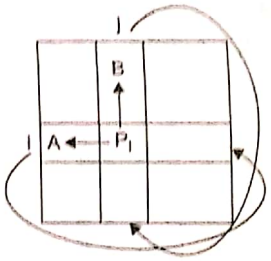
6.3.1 Programming the Google App Engine

Several web resources (e.g., http://code.google.com/appengine/) and specific books and articles (e.g., www.byteonic.com/2009/overview-of-java-support-in-google-app-engine/) discuss how to program GAE. Figure 6.17 summarizes some key features of 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.

There are several powerful constructs for storing and accessing data. 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. Queries can retrieve entities of a given kind filtered and sorted by the values of the properties. Java offers Java Data Object (JDO) and Java Persistence API (JPA) interfaces implemented by the open source Data Nucleus Access platform, while Python has a SQL-like query language called GQL. The data store is strongly consistent and uses optimistic concurrency control.

Scanned with CamScanner

Table 6.11 Comparison of MapReduce++ Subcategories along with the Loosely Synchronous Category Used in MPI

Map-Only	Classic MapReduce	Iterative MapReduce	Loosely Synchronous
			

• Document conversion (e.g., PDF->HTML)
• Brute force searches in cryptography
• Parametric sweeps
• Gene assembly
• PolarGrid Matlab data analysis (www.polargrid.org)
	• High-energy physics (HEP) histograms
	• Distributed search
	• Distributed sort
	• Information retrieval
	• Calculation of pairwise distances for sequences (BLAST)
		• Expectation maximization algorithms
		• Linear algebra
		• Data mining including
		  • Clustering
		  • K-means
		  • Deterministic annealing clustering
		  • Multidimensional scaling (MDS)
			• Many MPI scientific applications utilizing a wide variety of communication constructs including local interactions
			• Solving differential equations and particle dynamics with short-range forces

← **Domain of MapReduce and Iterative Extensions** → **MPI**

Scanned with CamScanner

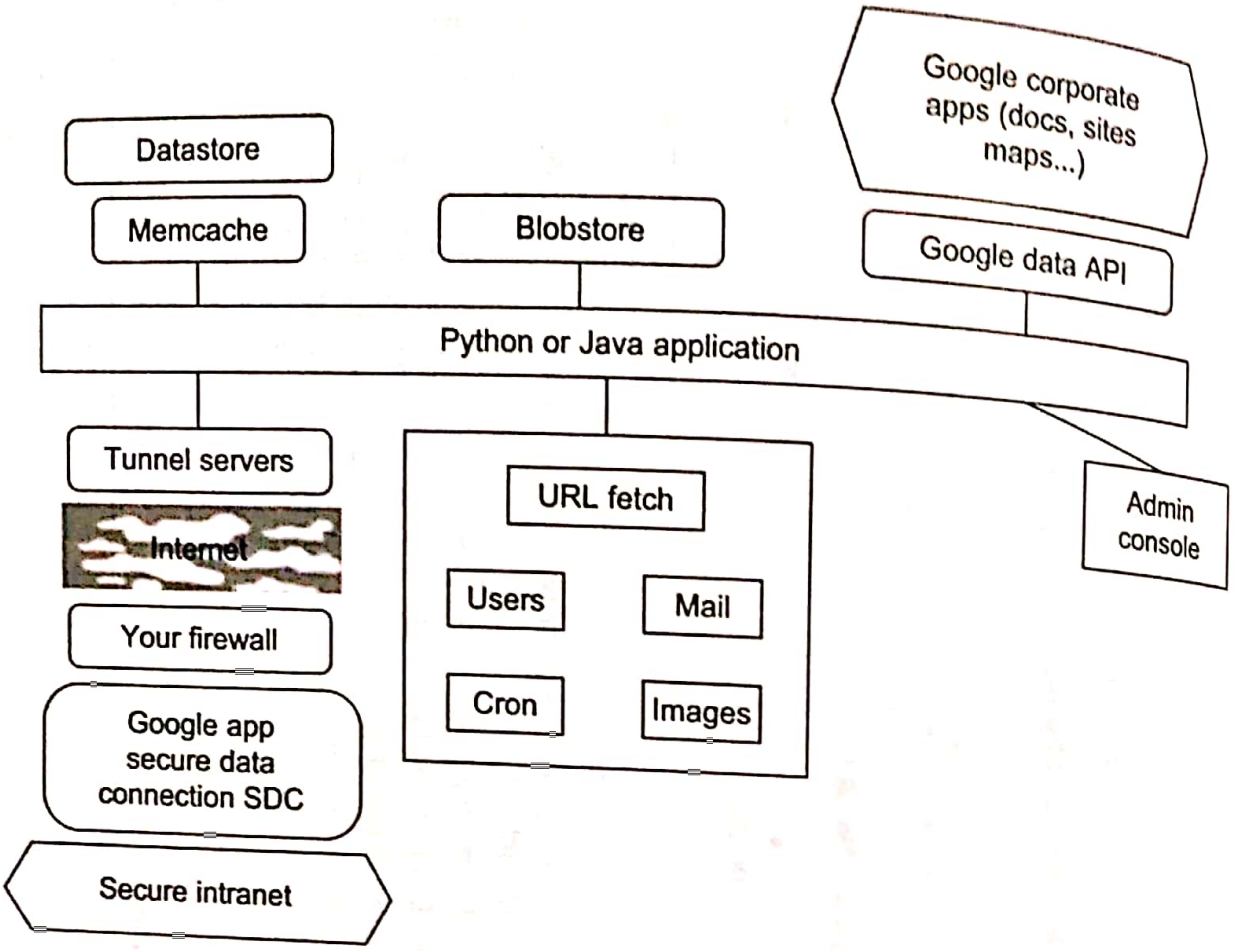


FIGURE 6.17

Programming environment for Google AppEngine.

An update of an entity occurs in a transaction that is retried a fixed number of times if other processes are trying to update the same entity simultaneously. Your application can execute multiple data store operations in a single transaction which either all succeed or all fail together. The data store implements transactions across its distributed network using "entity groups." A transaction manipulates entities within a single group. Entities of the same group are stored together for efficient execution of transactions. Your GAE application can assign entities to groups when the entities are created. 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.

Recently, 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.

Applications can access resources on the Internet, such as web services or other data, using GAE's URL fetch service. The URL fetch service retrieves web resources using the same high-speed Google infrastructure that retrieves web pages for many other Google products. There are dozens of Google "corporate" facilities including maps, sites, groups, calendar, docs, and YouTube, among others. These support the *Google Data API* which can be used inside GAE.

Scanned with CamScanner

6.4 PROGRAMMING ON AMAZON AWS AND MICROSOFT AZURE

In this section, we will consider the programming support in the AWS platform. First we will review the AWS platform and its updated service offerings. Then we will study the EC2, S3, and Simple DB services with programming examples. Returning to the programming environment features in Figures 4.22 and 4.23, Amazon (like Azure) offers a Relational Database Service (RDS) with a messaging interface sketched in Section 6.1.3. The Elastic MapReduce capability is equivalent to Hadoop running on the basic EC2 offering. Amazon has NOSQL support in SimpleDB introduced in Section 6.1.3 and discussed in Section 6.4.5. However, Amazon does not directly support BigTable as described in Section 6.3.4.

Now we will highlight a few more capabilities that are listed in Table 4.6. Amazon offers the Simple Queue Service (SQS) and Simple Notification Service (SNS), which are the cloud implementations of services discussed in Sections 5.2 and 5.4.5. Note that brokering systems run very efficiently in clouds and offer a striking model for controlling sensors and giving back-office support

Scanned with CamScanner

for a growing number of smartphones and tablets [77]. We further note the auto-scaling and elastic load balancing services which support related capabilities. Auto-scaling enables you to automatically scale your Amazon EC2 capacity up or down according to conditions that you define. With auto-scaling, you can ensure that the number of Amazon EC2 instances you're using scales up seamlessly during demand spikes to maintain performance, and scales down automatically during demand lulls to minimize cost.

Elastic load balancing automatically distributes incoming application traffic across multiple Amazon EC2 instances and allows you to avoid nonoperating nodes and to equalize load on functioning images. Both auto-scaling and elastic load balancing are enabled by CloudWatch which monitors running instances. CloudWatch is a web service that provides monitoring for AWS cloud resources, starting with Amazon EC2. It provides customers with visibility into resource utilization, operational performance, and overall demand patterns—including metrics such as CPU utilization, disk reads and writes, and network traffic.

6.4.1 Programming on Amazon EC2

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

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

$$\text{Create an AMI} \rightarrow \text{Create Key Pair} \rightarrow \text{Configure Firewall} \rightarrow \text{Launch} \tag{6.3}$$

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

Table 6.12 Three Types of AMI

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

Scanned with CamScanner

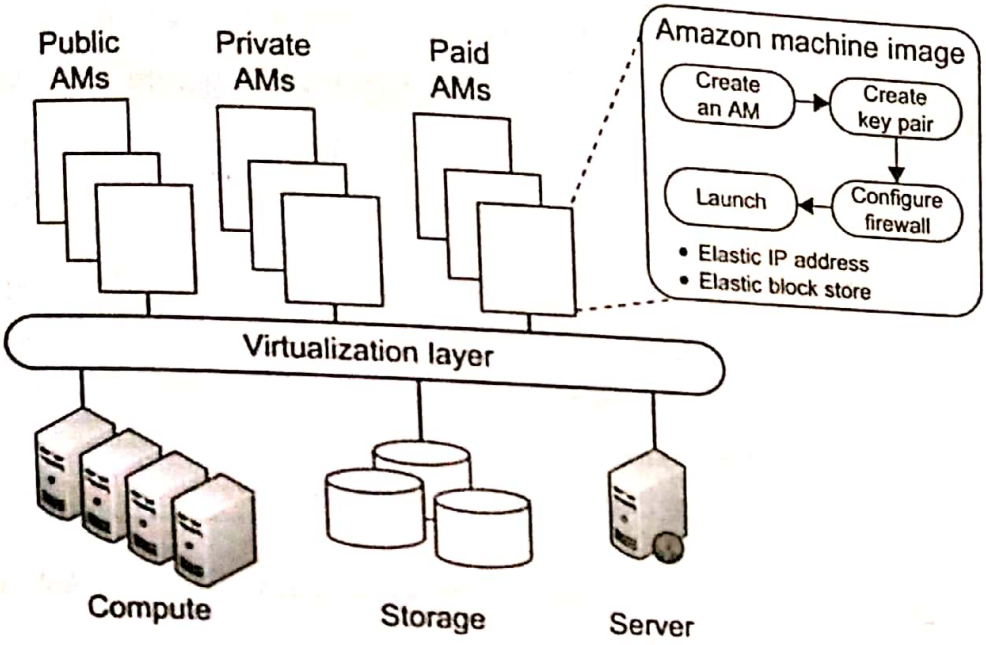


FIGURE 6.23

Amazon EC2 execution environment.

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

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

Example 6.9 Use of EC2 Services in the AWS Platform

Table 6.13 defines the IaaS instances available in October 2010 in five broad classes:

1. **Standard instances** are well suited for most applications.
2. **Micro instances** provide a small number of consistent CPU resources and allow you to burst CPU capacity when additional cycles are available. They are well suited for lower throughput applications and web sites that consume significant compute cycles periodically.
3. **High-memory instances** offer large memory sizes for high-throughput applications, including database and memory caching applications.

Scanned with CamScanner

Table 6.14 Cost of Amazon On-Demand VM Instant Types (October 6, 2010)

VM Instance Type	Size	Linux/UNIX Usage	Windows Usage
	Small (default)	$0.085 per hour	$0.12 per hour
Standard instances	Large	$0.34 per hour	$0.48 per hour
	Extra large	$0.68 per hour	$0.96 per hour
Micro instances	Micro	$0.02 per hour	$0.03 per hour
High-memory instances	Extra large	$0.50 per hour	$0.62 per hour
	Double extra large	$1.00 per hour	$1.24 per hour
	Quadruple extra large	$2.00 per hour	$2.48 per hour
Cluster compute instances	Quadruple extra large	$1.60 per hour	Not available

4. **High-CPU instances** have proportionally more CPU resources than memory (RAM) and are well suited for compute-intensive applications.
5. **Cluster compute instances** provide proportionally high CPU resources with increased network performance and are well suited for high-performance computing (HPC) applications and other demanding network-bound applications. They use 10 Gigabit Ethernet interconnections.

The cost in the third column is expressed in terms of EC2 Compute Units (ECUs) where one ECU provides the CPU capacity of a 1.0–1.2 GHz 2007 Opteron or 2007 Xeon processor. This leads to the cost per hour for CPUs shown in Table 6.14. Note that a real-world use of EC2 must pay for use of many different resources; the CPU charge in Table 6.14 is just one component and all charges (which naturally change often and so the reader should get the latest values online) are available on the AWS web site.

■

6.4.2 Amazon Simple Storage Service (S3)

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. SQS is responsible for ensuring a reliable message service between two processes, even if the receiver processes are not running. Figure 6.24 shows the S3 execution environment.

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).

Scanned with CamScanner

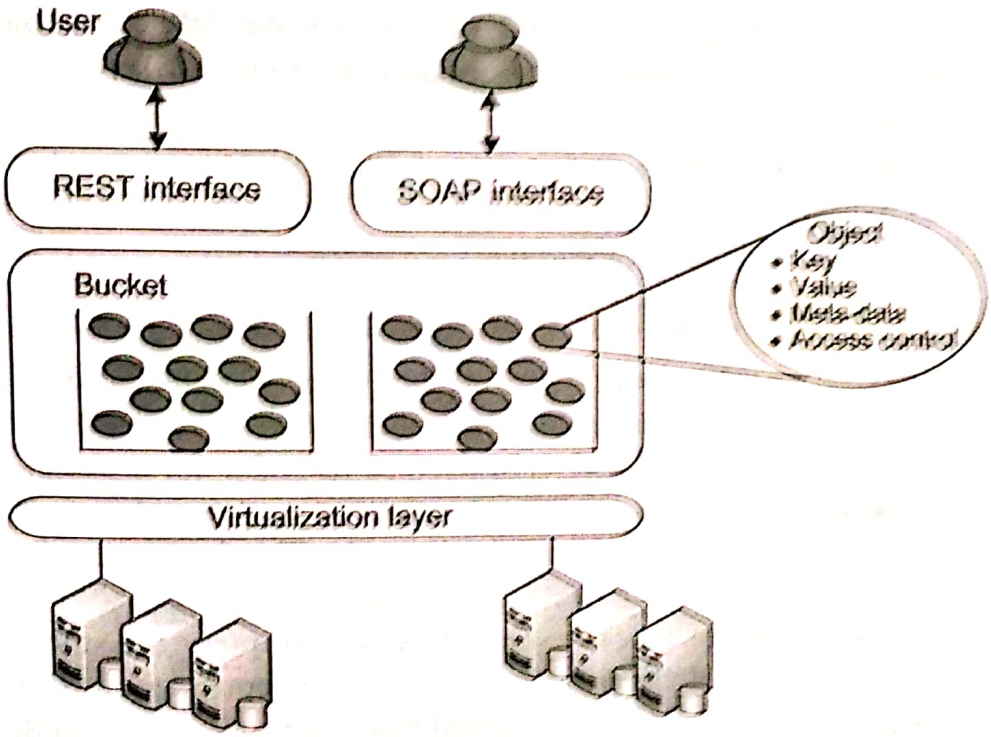


FIGURE 6.24

Amazon S3 execution environment.

- 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 or for data transferred between the Amazon EC2 Northern Virginia region and the Amazon S3 U.S. Standard region (as of October 6, 2010).

Scanned with CamScanner

6.5 EMERGING CLOUD SOFTWARE ENVIRONMENTS

In this section, we will assess popular cloud operating systems and emerging software environments. We cover the open source Eucalyptus and Nimbus, then examine OpenNebula, Sector/Sphere, and Open Stack. These environments were introduced in Chapter 3 from a virtualization perspective. Here, we provide details regarding programming requirements. We will also cover the Aneka cloud programming tools recently developed at the University of Melbourne.

6.5.1 Open Source Eucalyptus and Nimbus

Eucalyptus is a product from Eucalyptus Systems (www.eucalyptus.com) that was developed out of a research project at the University of California, Santa Barbara. Eucalyptus was initially aimed at bringing the cloud computing paradigm to academic supercomputers and clusters. Eucalyptus provides an AWS-compliant EC2-based web service interface for interacting with the cloud service. Additionally, Eucalyptus provides services, such as the AWS-compliant Walrus, and a user interface for managing users and images.

6.5.1.1 Eucalyptus Architecture

The Eucalyptus system is an open software environment. The architecture was presented in a Eucalyptus white paper [79,80]. Figure 3.31 introduced Eucalyptus from a virtual clustering point of view. Figure 6.26 shows the architecture based on the need to manage VM images. The system supports cloud programmers in VM image management as follows. Essentially, the system has been extended to support the development of both the computer cloud and storage cloud.

6.5.1.2 VM 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 appliances (http://en.wikipedia.org/wiki/Virtual_appliance) and deploy them within Eucalyptus with ease. The Eucalyptus system is available in a commercial proprietary version, as well as the open source version we just described.

6.5.1.3 Nimbus

Nimbus [81,82] is a set of open source tools that together provide an IaaS cloud computing solution. Figure 6.27 shows the architecture of Nimbus, which allows a client to lease remote resources by deploying VMs on those resources and configuring them to represent the environment desired

Scanned with CamScanner

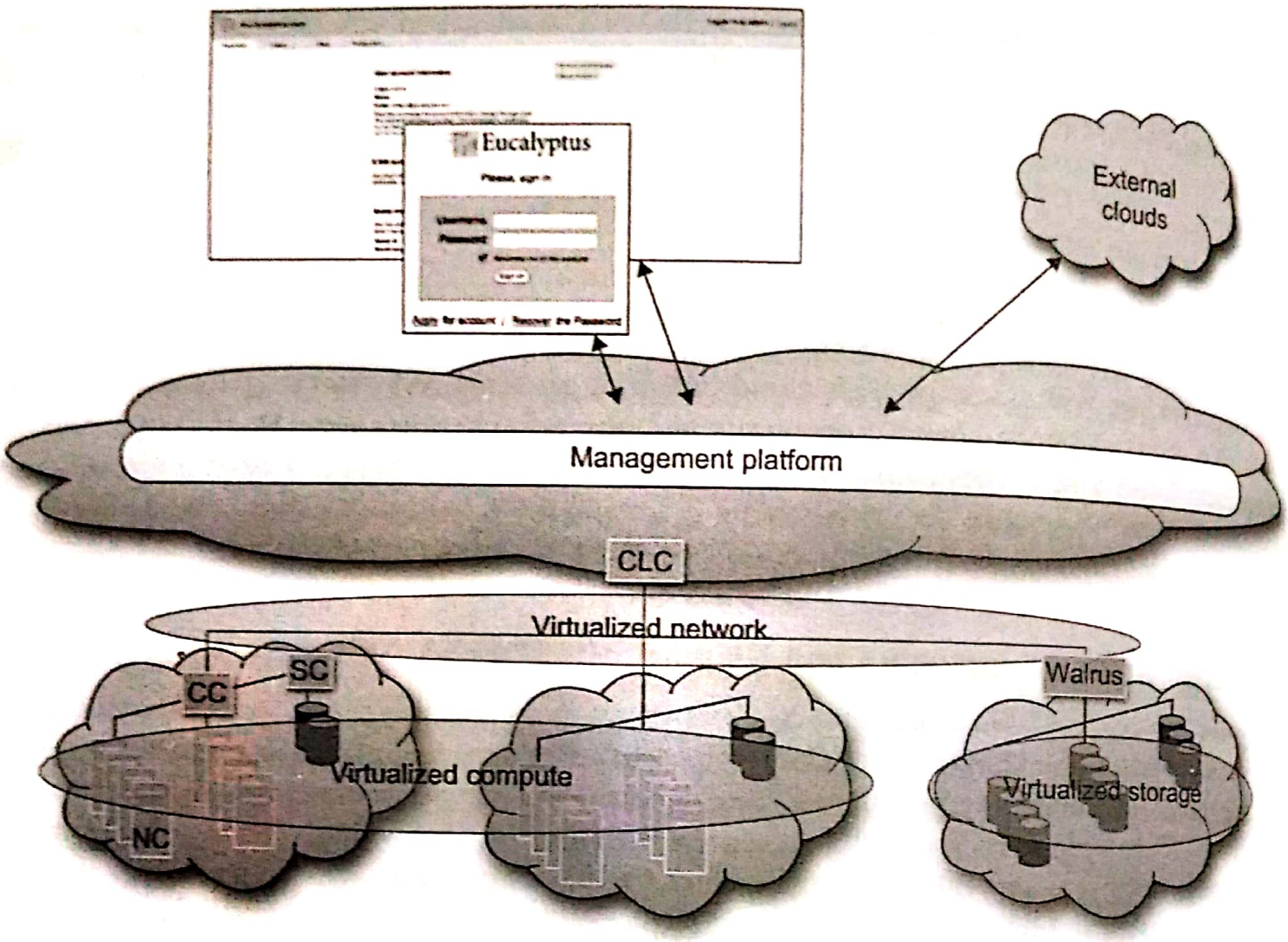


FIGURE 6.26

The Eucalyptus architecture for VM image management.

(Courtesy of Eucalyptus LLC [81])

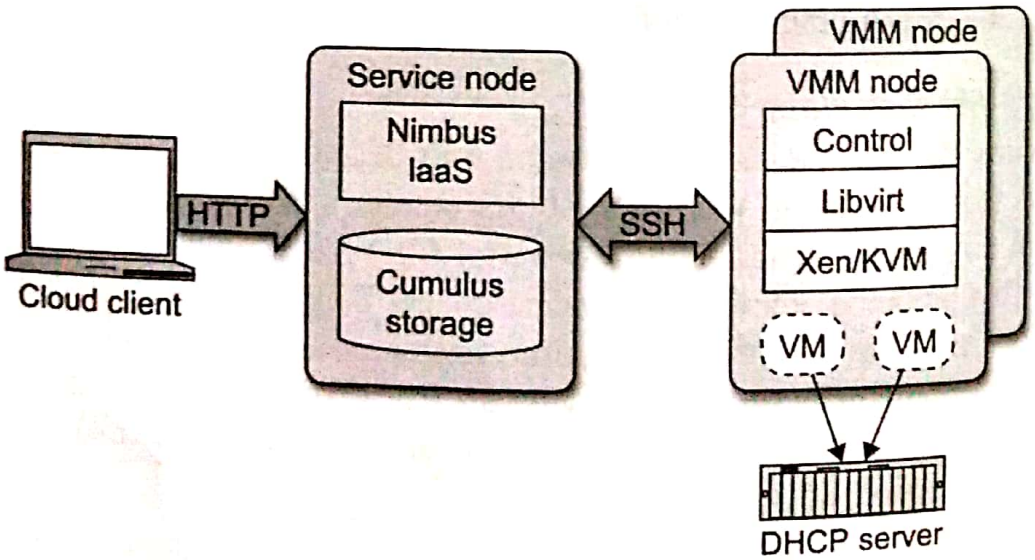


FIGURE 6.27

Nimbus cloud infrastructure.

(Courtesy of Nimbus Project [82])

Scanned with CamScanner

by the user. To this end, Nimbus provides a special web interface known as Nimbus Web [83]. Its aim is to provide administrative and user functions in a friendly interface. Nimbus Web is centered around a Python Django [84] web application that is intended to be deployable completely separate from the Nimbus service.

As shown in Figure 6.27, a storage cloud implementation called Cumulus [83] has been tightly integrated with the other central services, although it can also be used stand-alone. Cumulus is compatible with the Amazon S3 REST API [85], but extends its capabilities by including features such as quota management. Therefore, clients such as boto [86] and s2cmd [87], that work against the S3 REST API, work with Cumulus. On the other hand, the Nimbus cloud client uses the Java Jets3t library [88] to interact with Cumulus.

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 [89] that allows users to use clients developed for the real EC2 system against Nimbus-based clouds.

Scanned with CamScanner

6.5.2.2 OpenStack

OpenStack [103] 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 the tradition of other open source projects, the software is open source and limited to just open source APIs such as Amazon.

Scanned with CamScanner

Currently, OpenStack focuses on the development of two aspects of cloud computing to address compute and storage aspects with the OpenStack Compute and OpenStack Storage solutions. "OpenStack Compute is the internal fabric of the cloud creating and managing large groups of virtual private servers" and "OpenStack Object Storage is software for creating redundant, scalable object storage using clusters of commodity servers to store terabytes or even petabytes of data." Recently, an image repository was prototyped. The image repository contains an image registration and discovery service and an image delivery service. Together they deliver images to the compute service while obtaining them from the storage service. This development gives an indication that the project is striving to integrate more services into its portfolio.

6.5.2.3 OpenStack Compute

As part of its computing support efforts, OpenStack [103] is developing a cloud computing fabric controller, a component of an IaaS system, known as Nova. The architecture for Nova is built on the concepts of shared-nothing and messaging-based information exchange. Hence, most communication in Nova is facilitated by message queues. To prevent blocking components while waiting for a response from others, deferred objects are introduced. Such objects include callbacks that get triggered when a response is received. This is very similar to established concepts from parallel computing, such as "futures," which have been used in the grid community by projects such as the CoG Kit.

To achieve the shared-nothing paradigm, the overall system state is kept in a distributed data system. State updates are made consistent through atomic transactions. Nova it implemented in Python while utilizing a number of externally supported libraries and components. This includes boto, an Amazon API provided in Python, and Tornado, a fast HTTP server used to implement the S3 capabilities in OpenStack. Figure 6.30 shows the main architecture of Open Stack Compute. In this 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

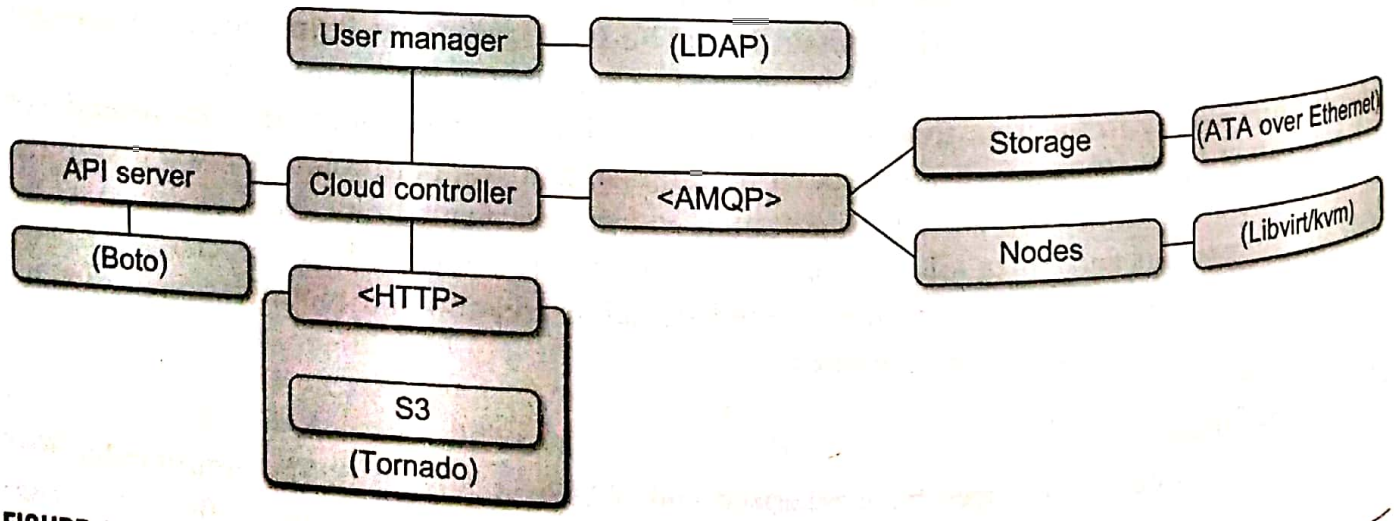


FIGURE 6.30

OpenStack Nova system architecture. The AMQP (Advanced Messaging Queuing Protocol) was described in Section 5.2.

Scanned with CamScanner

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 addressing, virtual private network (VPN) connectivity, and firewall rules. It includes the following types:

- *NetworkController* manages address and virtual LAN (VLAN) allocations
- *RoutingNode* governs the NAT (*network address translation*) conversion of public IPs to private IPs, and enforces firewall rules
- *AddressingNode* runs Dynamic Host Configuration Protocol (DHCP) services for private networks
- *TunnelingNode* provides VPN connectivity

The network state (managed in the distributed object store) consists of the following:

- VLAN assignment to a project
- Private subnet assignment to a security group in a VLAN
- Private IP assignments to running instances
- Public IP allocations to a project
- Public IP associations to a private IP/running instance

6.5.2.4 OpenStack Storage

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. Thus, any object is streamed to or from an object server directly through the proxy server to or from the user. A ring represents a mapping between the names of entities stored on disk and their physical locations.

Separate rings for accounts, containers, and objects exist. A ring includes the concept of using zones, devices, partitions, and replicas. Hence, it allows the system to deal with failures, and isolation of zones representing a drive, a server, a cabinet, a switch, or even a data center. Weights can be used to balance the distribution of partitions on drives across the cluster, allowing users to support heterogeneous storage resources. According to the documentation, "the Object Server is a very simple blob storage server that can store, retrieve and delete objects stored on local devices."

Objects are stored as binary files with metadata stored in the file's extended attributes. This requires that the underlying file system is built around object servers, which is often not the case for standard Linux installations. To list objects, a container server can be utilized. Listing of containers is handled by the account server. The first release of OpenStack "Austin" Compute and Object Storage was October 22, 2010. This system has a strong developer community.

6.5.3 Manjrasoft Aneka Cloud and Appliances

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

Scanned with CamScanner

accessible through the Internet to its subscribers, or a private cloud constituted by a set of nodes with restricted access as shown in Figure 6.31.

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

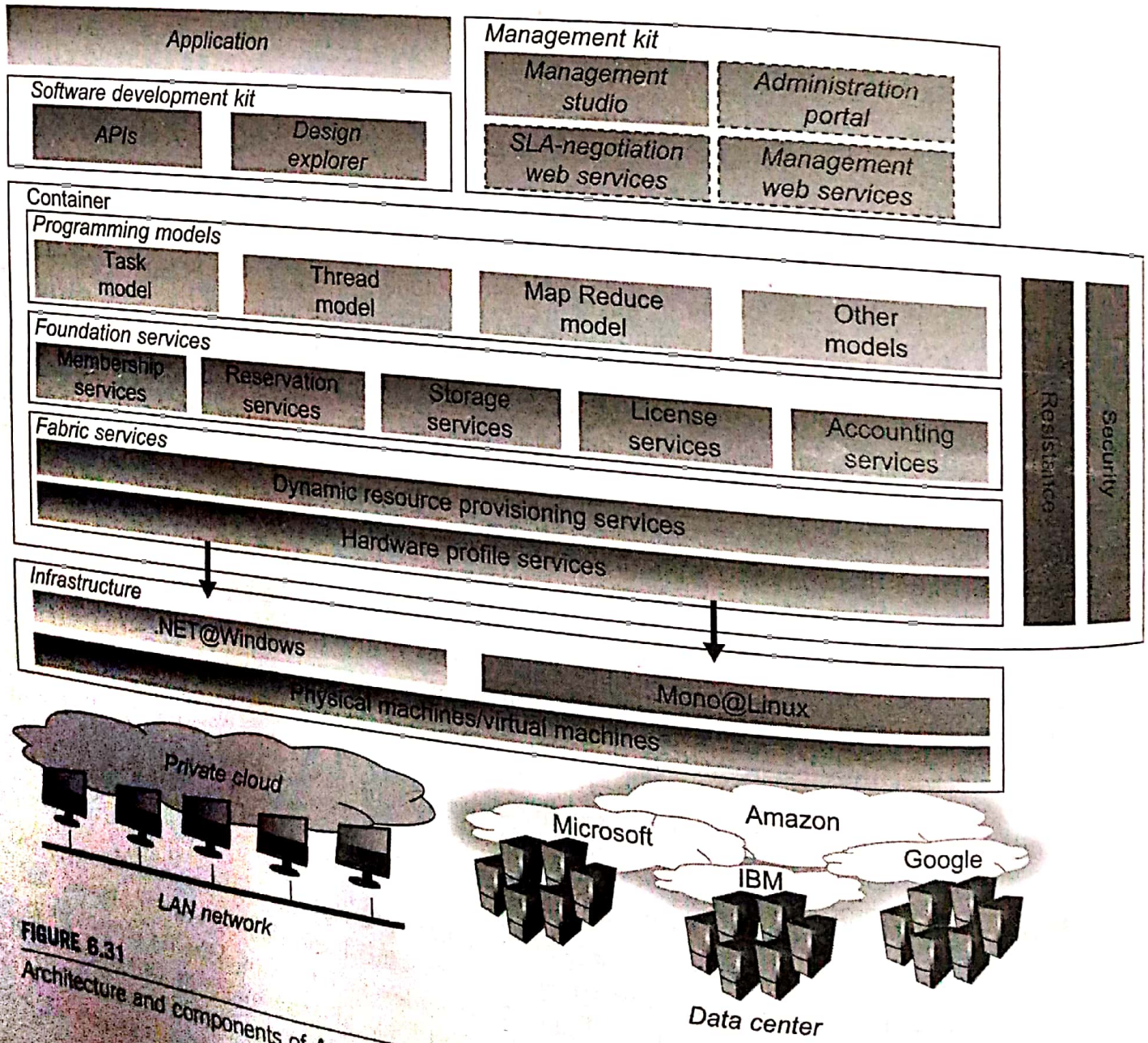


FIGURE 6.31
Architecture and components of Aneka.

Scanned with CamScanner

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 (see Figure 6.32).

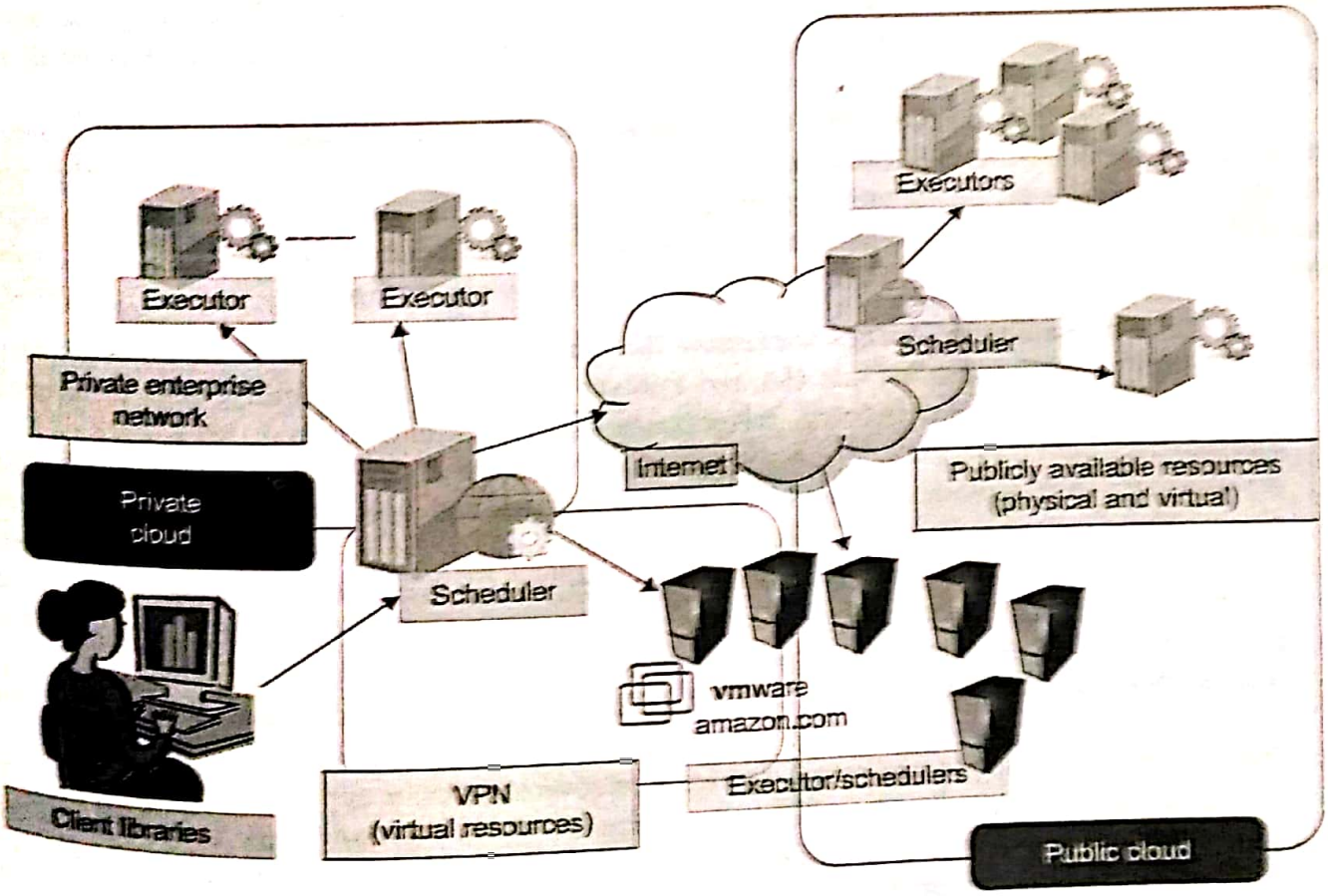


FIGURE 6.32

Aneka using private cloud resources along with dynamically leased public cloud resources.

(Courtesy of Raj Buyya, Manjrasoft, www.manjrasoft.com)

Scanned with CamScanner

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,as discussed in Section 6.2.2

6.5.3.1 Aneka Architecture

Aneka as a cloud application platform features a homogeneous distributed runtime environment for applications. This environment is built by aggregating together physical and virtual nodes hosting the Aneka container. The container is a lightweight layer that interfaces with the hosting environment and manages the services deployed on a node. The interaction with the hosting platform is mediated through the *Platform Abstraction Layer (PAL)*, which hides in its implementation all the heterogeneity of the different operating systems.

By means of the PAL it is possible to perform all the infrastructure-related tasks, such as performance and system monitoring. These activities are vital to ensure the desired QoS for applications. The PAL, together with the container, represents the hosting environment of services which implement the core capabilities of the middleware and make it a dynamically composable and extensible system. The available services can be aggregated into three major categories:

Fabric Services: Fabric services implement the fundamental operations of the infrastructure of the cloud. These services include HA and failover for improved reliability, node membership and directory, resource provisioning, performance monitoring, and hardware profiling.

Foundation Services: Foundation services constitute the core functionalities of the Aneka middleware. They provide a basic set of capabilities that enhance application execution in the cloud. These services provide added value to the infrastructure and are of use to system administrators and developers. Within this category we can list storage management, resource reservation, reporting, accounting, billing, services monitoring, and licensing. Services in this level operate across the range of supported application models.

Application Services: Application services deal directly with the execution of applications and are in charge of providing the appropriate runtime environment for each application model. They leverage foundation and fabric services for several tasks of application execution such as elastic scalability, data transfer, and performance monitoring, accounting, and billing. At this level, Aneka expresses its true potential in supporting different application models and distributed programming patterns.

Scanned with CamScanner

UNIT-V
CLOUD SECURITY

Security Overview-Cloud Security Challenges and Risks-Software-as-a-Service-Security Security Governance-Risk Management-Security Monitoring-Security Architecture Design-Data Security-Application Security-Virtual Machine Security-Identity Management and Access Control-Autonomic Security.

Security Overview

It is a set of control-based technologies & policies adapted to stick to regulatory compliances, rules & protect data application and cloud technology infrastructure. Because of cloud's nature of sharing resources, cloud security gives particular concern to identity management, privacy & access control. So the data in the cloud should have to be stored in an encrypted form. With the increase in the number of organizations using cloud technology for a data operation, proper security and other potentially vulnerable areas became a priority for organizations contracting with cloud providers. Cloud computing security processes the security control in cloud & provides customer data security, privacy & compliance with necessary regulations.

Cloud Security Challenges and Risks

These are:

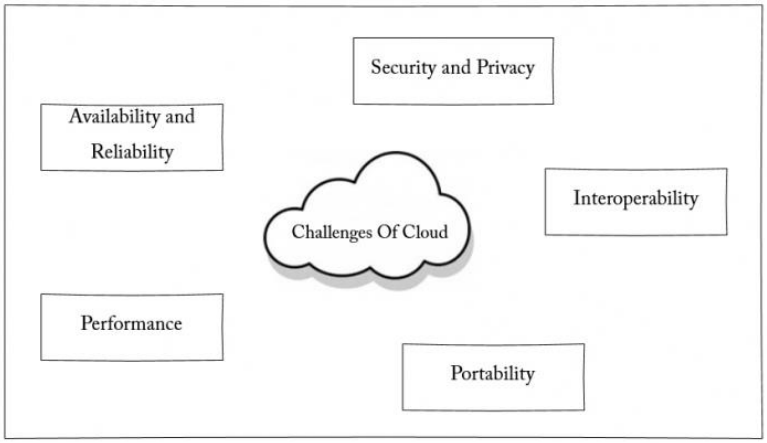
- Analyze the sensitivity to risks of user's resources.
- The cloud service models require the customer to be responsible for security at various levels of service.
- Understand the data storage and transfer mechanism provided by the cloud service provider.
- Consider proper cloud type to be used.

This emergent cloud technology is facing many technological challenges in different aspects of data & information handling & storage.

Some of the challenges are as follows:

- Availability & reliability
- Security & Privacy
- Interoperability
- Performance
- Portability

Figure - Challenges Of Cloud:



The challenges as mentioned above are the most important and concerned points that should be processed for the betterment.

Security Planning

Before deploying a particular resource to cloud, one should need to analyze several aspects of the resource such as:

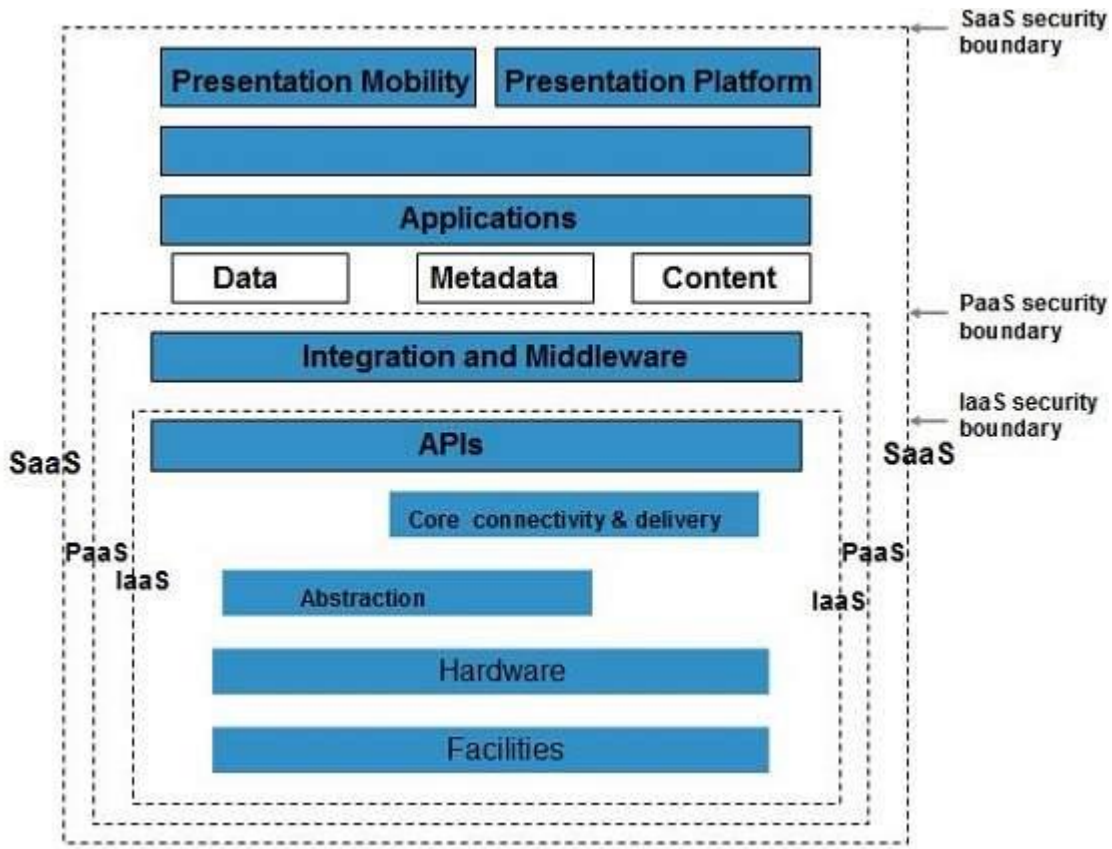
- Select resource that needs to move to the cloud and analyze its sensitivity to risk.
- Consider cloud service models such as **IaaS, PaaS,** and **SaaS.** These models require customer to be responsible for security at different levels of service.
- Consider the cloud type to be used such as **public, private, community** or **hybrid.**
- Understand the cloud service provider's system about data storage and its transfer into and out of the cloud.

The risk in cloud deployment mainly depends upon the service models and cloud types.

Understanding Security of Cloud

Security Boundaries

A particular service model defines the boundary between the responsibilities of service provider and customer. **Cloud Security Alliance (CSA)** stack model defines the boundaries between each service model and shows how different functional units relate to each other. The following diagram shows the **CSA stack model:**



Key Points to CSA Model

- IaaS is the most basic level of service with PaaS and SaaS next two above levels of services.
- Moving upwards, each of the service inherits capabilities and security concerns of the model beneath.
- IaaS provides the infrastructure, PaaS provides platform development environment, and SaaS provides operating environment.
- IaaS has the least level of integrated functionalities and integrated security while SaaS has the most.
- This model describes the security boundaries at which cloud service provider's responsibilities end and the customer's responsibilities begin.
- Any security mechanism below the security boundary must be built into the system and should be maintained by the customer.

Although each service model has security mechanism, the security needs also depend upon where these services are located, in private, public, hybrid or community cloud.

The main security risks of cloud computing are:

- Compliance violations
- Identity theft
- Malware infections and data breaches
- Diminished customer trust and potential revenue loss

While the concern is understandable, today's reality is that—when implemented correctly—cloud computing security is just as reliable as traditional on-premise IT.

Software-as-a-Service-Security

Software-as-a-service (SaaS) is an on-demand, cloud-based software delivery model that enables organizations to subscribe to the applications they need without hosting them in house. SaaS is one of several categories of cloud subscription services, including platform-as-a-service and infrastructure-as-a-service. SaaS has become increasingly popular because it saves organizations from needing to purchase servers and other infrastructure or maintain an in-house support staff. Instead, a SaaS provider hosts and provides SaaS security and maintenance to their software. Some well-known SaaS applications include Microsoft Office 365, Salesforce.com, Cisco Webex, Box, and Adobe Creative Cloud. Most enterprise software vendors also offer cloud versions of their applications, such as Oracle Financials Cloud.

SaaS security

SaaS providers handle much of the security for a cloud application. The SaaS provider is responsible for securing the platform, network, applications, operating system, and physical infrastructure. However, providers are not responsible for securing customer data or user access to it. Some providers offer a bare minimum of security, while others offer a wide range of SaaS security options.

- **Apply identity and access management (IAM).** A role-based identity and access management solution can ensure that end users do not gain access to more resources than they require for their jobs. IAM solutions use processes and user access policies to determine what files and applications a particular user can access. An organization can apply role-based permissions to data so that end users will see only the data they're authorized to view.
- **Encrypt cloud data.** Data encryption protects both data at rest (in storage) and data in transit between the end user and the cloud or between cloud applications. Government regulations usually require encryption of sensitive data. Sensitive data includes financial information, healthcare data, and personally identifiable information (PII). While a SaaS vendor may provide some type of encryption, an organization can enhance data security by applying its own encryption, such as by implementing a cloud access security broker (CASB).
- **Enforce data loss prevention (DLP).** DLP software monitors for sensitive data within SaaS applications or outgoing transmissions of sensitive data and blocks the transmission. DLP software detects and prevents sensitive data from being downloaded to personal devices and blocks malware or hackers from attempting to access and download data.
- **Monitor collaborative sharing of data.** Collaboration controls can detect granular permissions on files that are shared with other users, including users outside the organization who access the file through a web link. Employees may inadvertently or intentionally share confidential documents through email, team spaces, and cloud storage sites such as Dropbox.

SaaS security solutions

- **Data loss prevention (DLP)**) safeguards intellectual property and protects sensitive data in cloud applications, as well as at endpoints such as laptops. Organizations can define data access policies that DLP enforces.
- **Compliance solutions** provide controls and reporting capabilities to ensure compliance with government and industry regulations.
- **Advanced malware prevention** includes technologies such as behavioral analytics and real-time threat intelligence that can help detect and block zero-day attacks and malicious files that may be spread through cloud email and file sharing applications.
- **Cloud access security brokers (CASBs)** protect enterprise data and users across all cloud services, including SaaS, PaaS, and IaaS. According to Gartner's Magic Quadrant for Cloud Access Security Brokers, CASBs detect threats and provide IT departments with greater visibility into data usage and user behavior for cloud services, end users, and devices. CASBs also act immediately to remediate security threats by eliminating security misconfigurations and correcting high-risk user activities applications. CASBs provide a variety of security services, including:
    - Monitoring for unauthorized cloud services
    - Enforcing data security policies including encryption
    - Collecting details about users who access data in cloud services from any device or location
    - Restricting access to cloud services based on the user, device, and application
    - Providing compliance reporting

Security Governance

Cloud security governance refers to the management model that facilitates effective and efficient security management and operations in the cloud environment so that an enterprise's business targets are achieved. This model incorporates a hierarchy of executive mandates, performance expectations, operational practices, structures, and metrics that, when implemented, result in the optimization of business value for an enterprise. Cloud security governance helps answer leadership questions such as:

- Are our security investments yielding the desired returns?
- Do we know our security risks and their business impact?
- Are we progressively reducing security risks to acceptable levels?
- Have we established a security-conscious culture within the enterprise?

Strategic alignment, value delivery, risk mitigation, effective use of resources, and performance measurement are key objectives of any IT-related governance model, security included. To successfully pursue and achieve these objectives, it is important to understand the operational culture and business and customer profiles of an enterprise, so that an effective security

Key Objectives for Cloud Security Governance

The following represents key objectives to pursue in establishing a governance model for security in the cloud. These objectives assume that appropriate security standards and a governance framework have been chosen based on the enterprise's business targets, customer profile, and obligations for protecting data and other information assets in the cloud environment.

1. **Strategic Alignment**
   Enterprises should mandate that security investments, services, and projects in the cloud are executed to achieve established business goals (e.g., market competitiveness, financial, or operational performance).

2. **Value Delivery**
   Enterprises should define, operationalize, and maintain an appropriate security function/organization with appropriate strategic and tactical representation, and charged with the responsibility to maximize the business value (Key Goal Indicators, ROI) from the pursuit of security initiatives in the cloud.

3. **Risk Mitigation**
   Security initiatives in the cloud should be subject to measurements that gauge effectiveness in mitigating risk to the enterprise (Key Risk Indicators). These initiatives should also yield results that progressively demonstrate a reduction in these risks over time.

4. **Effective Use of Resources**
   It is important for enterprises to establish a practical operating model for managing and performing security operations in the cloud, including the proper definition and operationalization of due processes, the institution of appropriate roles and responsibilities, and use of relevant tools for overall efficiency and effectiveness.

5. **Sustained Performance**
   Security initiatives in the cloud should be measurable in terms of performance, value and risk to the enterprise (Key Performance Indicators, Key Risk Indicators), and yield results that demonstrate attainment of desired targets (Key Goal Indicators) over time.

Risk Management

It is a

- Set of activities and methods to control risks.
- Architecture to manage risks.
- Identifying the security risks
- Occurrence for these risks
- Safeguards against these risks
- Controls for reducing or eliminating those risks

Risk assessment steps

- Threat Identification
- Vulnerability Identification(list)
- Risk Determination(level)
- Control Recommendation

To Evaluate

- Data Location
- Recovery
- Viability
- Support in Reducing Risk

Data Location

- Every customer need to know where his data are hosted, in which country the data is stored

Recovery

- How cloud offerings will recover from total disaster?
- Know what will happen if one of the offered sites went down?
- Can it completely restore everything?
- How much time does it need to complete restoration?

Viability

- What would happen to your service if the provider goes broke?
- How would I get my data back?
- Can I use the data in a replacement application?

Support in Reducing Risk

- How to use the product safely?
- To whom the instructions for setting and monitoring policies provided to ?
- How to avoid phishing or malware attacks?

Security Monitoring

Monitoring is a critical component of cloud security and management. Typically relying on automated solutions, cloud security monitoring supervises virtual and physical servers to continuously assess and measure data, application, or infrastructure behaviors for potential security threats. This assures that the cloud infrastructure and platform function optimally while minimizing the risk of costly data breaches.

BENEFITS OF CLOUD SECURITY MONITORING

Cloud monitoring provides an easier way to identify patterns and pinpoint potential security vulnerabilities in cloud infrastructure. As there's a general perception of a loss of control when valuable data is stored in the cloud, effective cloud monitoring can put companies more at ease with making use of the cloud for transferring and storing data.

When customer data is stored in the cloud, cloud monitoring can prevent loss of business and frustrations for customers by ensuring that their personal data is safe. The use of web services can increase security risks, yet cloud computing offers many benefits for businesses, from accessibility to a better customer experience. Cloud monitoring is one initiative that enables companies to find the balance between the ability to mitigate risks and taking advantage of the benefits of the cloud – and it should do so without hindering business processes.

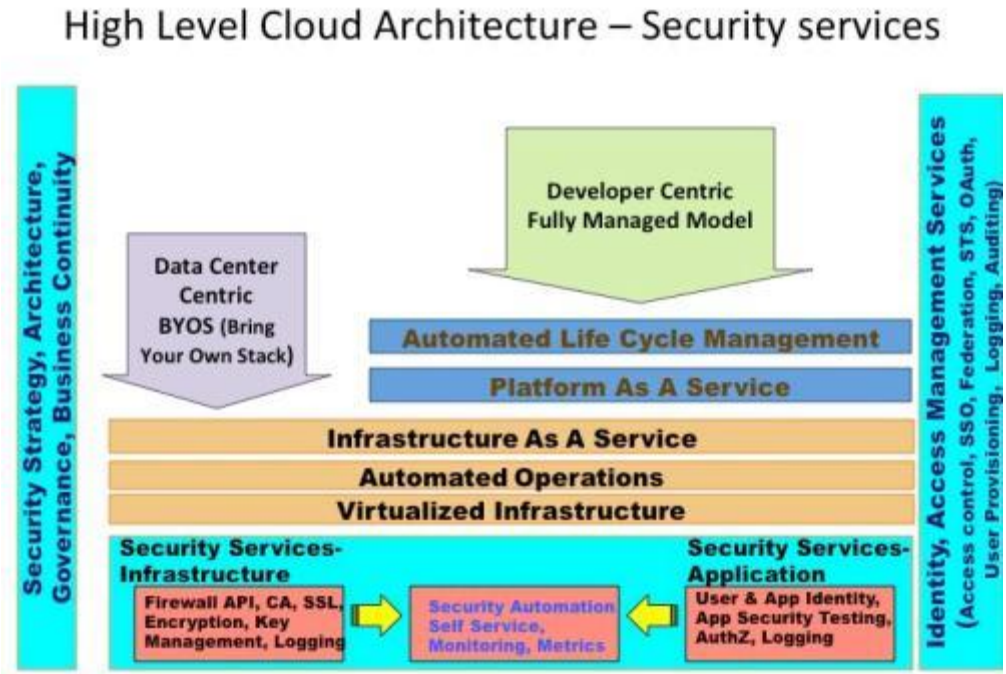
HOW CLOUD SECURITY MONITORING WORKS

There are several approaches to cloud security monitoring. Cloud monitoring can be done in the cloud platform itself, on premises using an enterprise's existing security management tools, or via a third party service provider. Some of the key capabilities of cloud security monitoring software include:

- **Scalability:** tools must be able to monitor large volumes of data across many distributed locations
- **Visibility:** the more visibility into application, user, and file behavior that a cloud monitoring solution provides, the better it can identify potential attacks or compromises
- **Timeliness:** the best cloud security monitoring solutions will provide constant monitoring, ensuring that new or modified files are scanned in real time
- **Integration:** monitoring tools must integrate with a wide range of cloud storage providers to ensure full monitoring of an organization's cloud usage
- **Auditing and Reporting:** cloud monitoring software should provide auditing and reporting capabilities to manage compliance requirements for cloud security

Security Architecture Design

As a first step, architects need to understand what security capabilities are offered by cloud platforms (PaaS, IaaS). The figure below illustrates the architecture for building security into cloud services.



The following are cloud security best practices to mitigate risks to cloud services:

- **Architect for security-as-a-service** – Application deployments in the cloud involve orchestration of multiple services including automation of DNS, load balancer, network QoS, etc. Security automation falls in the same category which includes automation of firewall policies between cloud security zones, provisioning of certificates (for SSL), virtual machine system configuration, privileged accounts and log configuration. Application deployment processes that depend on security processes such as firewall policy creation, certificate provisioning, key distribution and application pen testing should be migrated to a self-service model. This approach will eliminate human touch points and will enable a security as a service scenario. Ultimately this will mitigate threats due to human errors, improve operational efficiency and embed security controls into the cloud applications.

- **Implement sound identity, access management architecture and practice** – Scalable cloud bursting and elastic architecture will rely less on network based access controls and warrant strong user access management architecture. Cloud access control architecture should address all aspects of user and access management lifecycles for both end users and privileged users – user provisioning & deprovisioning, authentication, federation, authorization and auditing. A sound architecture will enable reusability of identity and access services for all use cases in

public, private and hybrid cloud models. It is good practice to employ secure token services along with proper user and entitlement provisioning with audit trails. Federation architecture is the first step to extending enterprise SSO to cloud services. Refer to cloud security alliance, Domain 12 for detailed guidance here.

- **Leverage APIs to automate safeguards** – Any new security services should be deployed with an API (REST/SOAP) to enable automation. APIs can help automate firewall policies, configuration hardening, and access control at the time of application deployment. This can be implemented using open source tools such as puppet in conjunction with the API supplied by cloud service provider.

- **Always encrypt or mask sensitive data** – Today's private cloud applications are candidates for tomorrow's public cloud deployment. Hence architect applications to encrypt all sensitive data irrespective of the future operational model.

- **Do not rely on an IP address for authentication services** – IP addresses in clouds are ephemeral in nature so you cannot solely rely on them for enforcing network access control. Employ certificates (self-signed or from a trusted CA) to enable SSL between services deployed on cloud.

- **Log, Log, Log** – Applications should centrally log all security events that will help create an end-to-end transaction view with non-repudiation characteristics. In the event of a security incident, logs and audit trails are the only reliable data leveraged by forensic engineers to investigate and understand how an application was exploited. Clouds are elastic and logs are ephemeral hence it is critical to periodically migrate log files to a different cloud or to the enterprise data center.

- **Continuously monitor cloud services** – Monitoring is an important function given that prevention controls may not meet all the enterprise standards. Security monitoring should leverage logs produced by cloud services, APIs and hosted cloud applications to perform security event correlation. Cloud audit (cloudaudit.org) from CSA can be leveraged towards this mission

In general, patterns should highlight the following attributes (but not limited to) for each of the security services consumed by the cloud application:

- Logical location – Native to cloud service, in-house, third party cloud. The location may have an implication on the performance, availability, firewall policy as well as governance of the service.
- Protocol – What protocol(s) are used to invoke the service? For example REST with X.509 certificates for service requests.
- Service function – What is the function of the service? For example encryption of the artifact, logging, authentication and machine finger printing.
- Input/Output – What are the inputs, including methods to the controls, and outputs from the security service? For example, Input = XML doc and Output =XML doc with encrypted attributes.

- Control description – What security control does the security service offer? For example, protection of information confidentiality at rest, authentication of user and authentication of application.
- Actor – Who are the users of this service? For example, End point, End user, Enterprise administrator, IT auditor and Architect.

Data Security

Data security has been incumbent on the cloud service providers, and they have risen to the occasion. No matter which platform you select in the debate between AWS vs. Azure vs. Google, all sport various compliances to standards like HIPAA, ISO, PCI DSS, and SOC.

Privacy Protection

Your data should be protected from unauthorized access regardless of your cloud decisions, which includes data encryption and controlling who sees and can access what. There may also situations where you want to make data available to certain personnel under certain circumstances. For example, developers need live data for testing apps but they don't necessarily need to see the data, so you would use a redaction solution. Oracle, for example, has a Data Redact tool for its databases.
The first step is something you should have done already: identify the sensitive data types and define them. Discover where the sensitive data resides, classify and define the data types, and create policies based on where the data is and which data types can go into the cloud and which cannot. Too many early adopters of the cloud rushed to move all their data there, only to realize it needed to be kept on-premises in a private cloud.

There are automated tools to help discover and identify an organization's sensitive data and where it resides. Amazon Web Services has Macie while Microsoft Azure has Azure Information Protection (AIP) to classify data by applying labels. Third party tools include Tableau, Fivetran, Logikcull, and Looker.

Preserve Data Integrity

Data integrity can be defined as protecting data from unauthorized modification or deletion. This is easy in a single database, because there is only one way in or out of the database, which you can control. But in the cloud, especially a multicloud environment, it gets tricky.

Because of the large number of data sources and means to access, authorization becomes crucial in assuring that only authorized entities can interact with data. This means stricter means of access, like two-factor authorization, and logging to see who accessed what. Another potential means of security is a trusted platform module (TPM) for remote data checks.

Data Availability

Downtime is a fact of life and all you can do is minimize the impact. That's of considerable importance with cloud storage providers because your data is on

someone else's servers. This is where the service-level agreement (SLA) is vital and paying a close eye to details really matters.

For example, Microsoft offers 99.9% availability for major Azure storage options but AWS offers 99.99% availability for stored objects. This difference is not trivial. Also, make sure your SLA allows you to specify where the data is stored. Some providers, like AWS, allow you to dictate in what region data is stored. This can be important for issues of compliance and response time/latency.

Every cloud storage service has a particular strength: Amazon's Glacier is ideal for mass cold storage of rarely accessed data, Microsoft's Azure blob storage is ideal for most unstructured data, while Google Cloud's SQL is tuned for MySQL databases.

Data Privacy

A huge raft of privacy laws, national and international, have forced more than a few companies to say no to the cloud because they can't make heads or tails of the law or it's too burdensome. And it's not hard to see why.

All of these protect the interest of the data owner, so it is in your best interest to know them and know how well your provider complies with them.

Encryption

Encryption is the means for which data privacy is protected and insured, and encryption technologies are fairly mature. Encryption is done via key-based algorithms and the keys are stored by the cloud provider, although some business-related apps, like Salesforce and Dynamix, use tokenization instead of keys. This involves substituting specific token fields for anonymous data tokens.

Virtually every cloud storage provider encrypts the data while it is in transfer. Most do it through browser interfaces, although there are some cloud storage providers like Mega and SpiderOak that use a dedicated client to perform the encryption. This should all be spelled out in the SLA.

Many cloud services offer key management solutions that allow you to control access because the encryption keys are in your hands. This may prove to be a better or at least more reassuring risk because you are in control over who has the keys. Again, this should be spelled out in the SLA.

Threats

If you are online you are under threat of attack, that is a fact of life. The old style of attacks, like DDoS attacks, SQL injection, and cross-site scripting, have faded into new fears. Cloud service providers have a variety of security tools and policies in place but problems still happen, usually originating in human error.

- **Data breaches:** This can happen any number of ways, from the usual means – a hacked account or a lost password/laptop – to means unique to the cloud. For example, it is possible for a user on one virtual machine to listen for the signal that an encryption key has arrived on another VM on the same host. It's called the "side channel timing exposure," and it means the victim's security credentials in the hands of someone else.

- **Data loss:** While the chance of data loss is minimal short of someone logging in and erasing everything, it is possible. You can mitigate this by insuring your applications and data are distributed across several zones and you backup your data using off-site storage.
- **Hijacked accounts:** All it takes is one lost notebook for someone to get into your cloud provider. Secure, tough passwords and two-factor authentication can prevent this. It also helps to have policies that look for and alert to unusual activity, like copying mass amounts of data or deleting it.
- **Cryptojacking:** Cryptojacking is the act of surreptitiously taking over a computer to farm cryptocurrency, which is a very compute-intensive process. Cryptojacking spiked in 2017 and 2018 and the cloud was a popular target because there is more compute resources available. Monitoring for unusual compute activity is the key way to stop this.

Application Security

Cloud application security is a series of defined policies, processes, controls, and technology governing all information exchanges that happen in collaborative cloud environments like Microsoft Office 365, Google G Suite, Slack, and Box (to name a few).
So, if you or your employees frequently store and share data in cloud applications like the ones listed above (or any of the tens of thousands available), it is absolutely necessary to add a cloud application "safety net" to your zero trust security infrastructure.

Top Cloud Application Security Threats
It is no secret that there are security issues in cloud computing that IT teams must be aware of. According to the 2018 Cybersecurity Insider Report, the four most common cloud application security threats that IT teams are facing include:

1. **Misconfiguration of application setup** is the single biggest threat to cloud security because data breaches tend to happen when services are accidentally exposed to the public internet.
2. **Unauthorized access** to a website, server, service, or other system is also an area for great concern because once they're in, there's no telling what unauthorized users will do to create chaos.
3. **Insecure APIs and interfaces** present easy opportunities for attackers to breach systems because they are the only asset(s) outside of the organizational boundary with a public IP address.
4. **Account hijacking** is feared because so much sensitive data and resources is stored and accessed on devices shared by many different users—and because keeping tabs on rogue employees is difficult.

Cloud providers, however, have no visibility into what happens at the application layer and have no way to help the application owners in this area. The application security layer is the responsibility of application owners.

Before we can evaluate a solution for application security, we need to understand the following challenges:

- Security monitoring – there are numerous questions about the solution's capability, but monitoring the security should not be one of them. Security monitoring is imperative; it's a must-have
- Application vulnerabilities – these are susceptible to attackers looking to exploit and attack an application, either to gain complete control over it, deform it or steal data. OWASP analyses such vulnerabilities and exploits, and regularly publishes a list of its top 10 identified vulnerabilities
- Malware and ransomware – another well-known security problem that impacts a lot of users and should be addressed prior to deploying an application in the cloud
- Bots – approximately 30 percent of traffic comes from non-useful bots (i.e. bad bots). While some people don't consider them a security issue, yet, bad bots can waste 30 percent in server resources, resulting in a huge loss of productivity
- Application layer DDoS attacks (volumetric or protocol exploits) – are also a concern as DDoS attacks evolve in size, scope and sophistication. DDoS protection is a serious consideration for both application owners and cloud infrastructure providers

Solving these challenges

- Fortunately, there are solutions available to overcome the security challenges associated with cloud applications.
- Web Application Firewalls (WAFs), for example, can handle the common vulnerabilities listed by OWASP. And IP reputation and other signature databases have been created to combat malware and bad BOTs.
- Many Application Delivery Controllers (ADCs) bundle application security solutions with load balancing and other key application services. Having a complete set of application delivery tools along with security and visibility in a DDoS resilient architecture can create a complicated deployment architecture.

Virtual Machine Security

Bringing VMs (virtual machines) onto the cloud. Recent years have seen great advancements in both cloud computing and virtualization On one hand there is the ability to pool various resources to provide software-as-a-service, infrastructure-as-a-service and platform-as-a-service. At its most basic, this is what describes cloud computing. On the other hand, we have virtual machines that provide agility, flexibility, and scalability to the cloud resources by allowing the vendors to copy, move, and manipulate their VMs at will. The term *virtual machine* essentially describes sharing the resources of one single physical computer into various computers within itself. *VMware* and *virtual box* are very commonly used virtual systems on desktops. Cloud computing effectively stands for many computers

pretending to be one computing environment. Obviously, cloud computing would have many virtualized systems to maximize resources.
One issue is that the cloud-computing scenario is not as transparent as it claims to be. The service user has no clue about how his information is processed and stored. In addition, the service user cannot directly control the flow of data/information storage and processing. The service provider usually is not aware of the details of the service running on his or her environment. Thus, possible attacks on the cloud-computing environment can be classified in to:

1. **Resource attacks:**

   These kinds of attacks include manipulating the available resources into mounting a large-scale botnet attack. These kinds of attacks target either cloud providers or service providers.

2. Data attacks: These kinds of attacks include unauthorized modification of sensitive data at nodes, or performing configuration changes to enable a sniffing attack via a specific device etc. These attacks are focused on cloud providers, service providers, and also on service users.

3. Denial of Service attacks: The creation of a new virtual machine is not a difficult task, and thus, creating rogue VMs and allocating huge spaces for them can lead to a Denial of Service attack for service providers when they opt to create a new VM on the cloud. This kind of attack is generally called virtual machine sprawling.

4. Backdoor: Another threat on a virtual environment empowered by cloud computing is the use of backdoor VMs that leak sensitive information and can destroy data privacy.

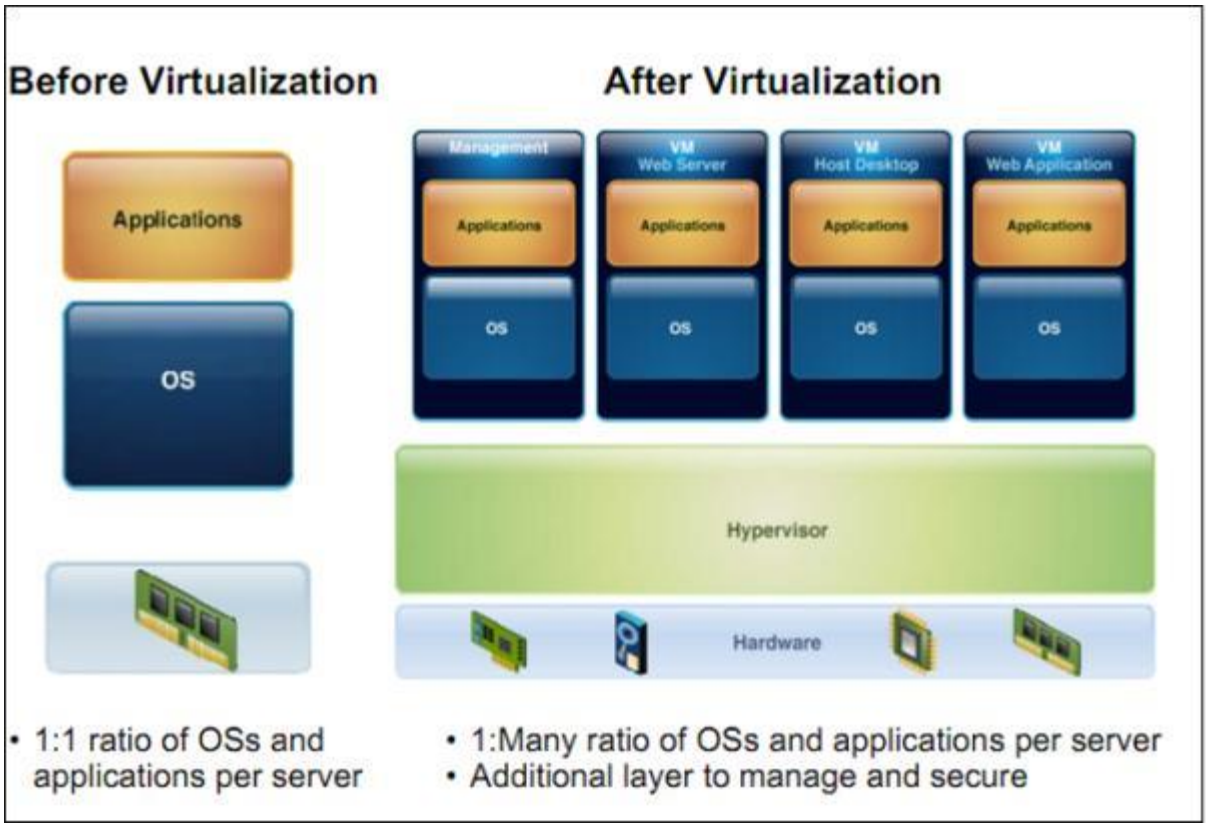
5. Having virtual machines would indirectly allow anyone with access to the host disk files of the VM to take a snapshot or illegal copy of the whole System. This can lead to corporate espionage and piracy of legitimate products.

In addition to the challenges introduced by cloud, there are challenges introduced by the dynamism of virtualization in cloud:

- Dynamic relocation of virtual machines (VMs): Hypervisors today move workloads based on the service level agreement (SLA), energy policy, resiliency policy, and a host of other reasons. IT administrators of today can no longer be sure where the workload resides in the data center.
- Increased infrastructure layers to manage and protect: Depending on the type of cloud model in use, there are a large number of additional infrastructure layers such as gateways, firewalls, access routers, and others, that need to be managed and protected, at the same time allowing access to the authorized users to perform their tasks.

- Multiple operating systems and applications per server: On virtualized commodity hardware, multiple workloads on a physical server run concurrently, with multiple operating systems and even with same operating systems but at different patch levels.
- Elimination of physical boundaries between systems: As virtualization adoption increases, workloads are co-located sharing the same physical infrastructure.
- Tracking software and configuration of VMs: As IT infrastructure becomes virtualized, it is increasingly complex to manage software configuration including patch levels, security patches, security audits, and others, not just for guest operating systems but also for other virtualized infrastructure such as virtual distributed switches.

  The figure shows before and after virtualization.



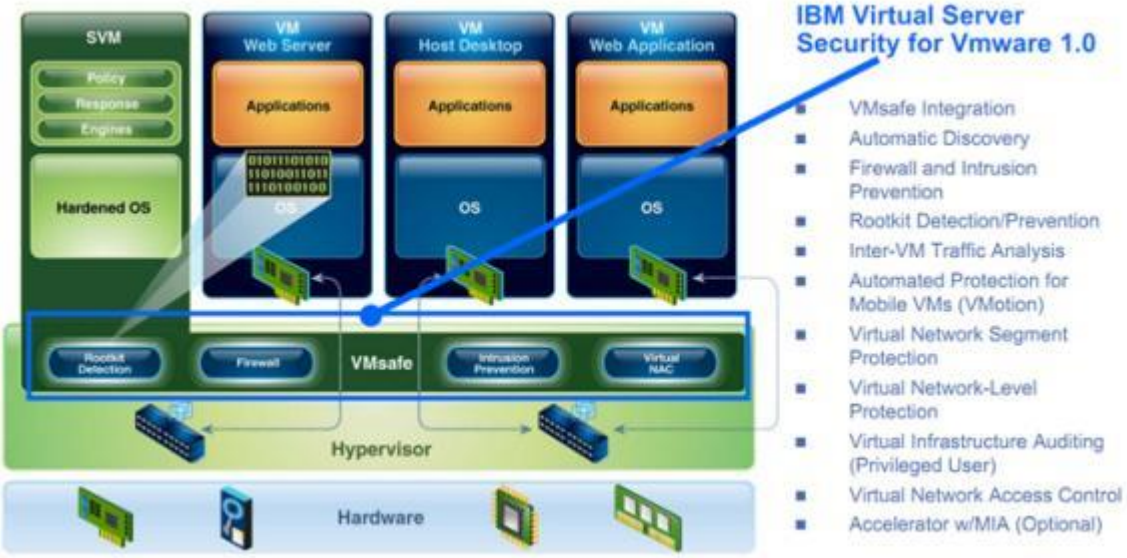
Challenges with existing security products

Traditional security products encounter new challenges in the virtualized world:

- Intrusiveness of existing solutions
- Reconfiguration of virtual network: Some existing solutions might require reconfiguration of the virtual network to allow for packet sniffing and protocol examination.
- Presence in the guest OS: For monitoring purposes, agents are required to be installed on the guest OS.
- Visibility and control gaps
- Virtual servers not connected to the physical network are invisible and unprotected.
- Lacks automation and transparency.

- Static security controls are too rigid. As VMs are moved around by the hypervisor, static controls need to be reapplied.
- No ability to deal with workload mobility exists.
- Resource overhead.
- Network traffic analysis in each guest OS is redundant, consuming more CPU cycles.

Virtual Server Protection for VMware is shown in the next figure.



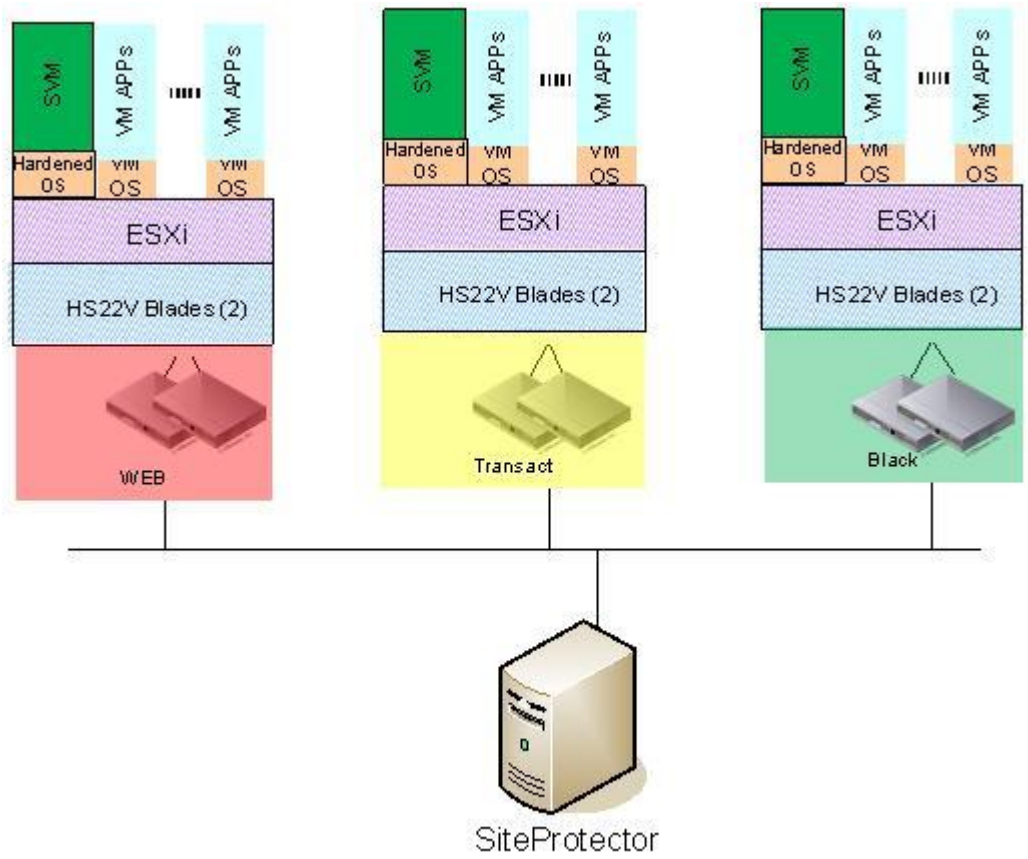
Virtual Server Protection for VMware provides the following benefits:

- Dynamic protection of every layer of infrastructure, mitigating the risks introduced by virtualization.
- Meets regulatory and compliance requirements.
- Increases ROI of virtual infrastructure because it is easy to deploy and maintain security.

Integrated security benefits of Virtual Server Protection for VMware are as follows:

- **Transparency**
  - No reconfiguration of virtual network required
  - No presence in guest OS
- **Security consolidation**
  - Only *one* Security Virtual Machine (SVM) required per physical server
  - 1:many protection ratio
  - **Automation**
  - Privileged presence gives SVM holistic view of the virtual network
  - Protection applied automatically as each new VM comes online
- **Efficiency**
  - Eliminates redundant processing tasks
- **Protection for any guest OS**

An example of Virtual Server Protection for VMware architecture is shown in the next figure:



In the figure example, three ESX clusters are in three separate network zones (WEB, Transact, and Black), separated physically. Virtual Server Protection for VMware is deployed on each ESX host in the cluster (HS22V blades in this example), which monitors all VMs as they are brought online. All policy events data is forwarded by the SVM to the SiteProtector appliance.

Identity Management and Access Control

IAM products provide IT managers with tools and technologies for controlling user access to critical information within an organization.

Identity and access management (IAM) is a framework of business processes, policies and technologies that facilitates the management of electronic or digital identities. With an IAM framework in place, information technology (IT) managers can control user access to critical information within their organizations. Identity and access management products offer role-based access control, which lets system administrators regulate access to systems or networks based on the roles of individual users within the enterprise.

In this context, access is the ability of an individual user to perform a specific task, such as view, create or modify a file. Roles are defined according to job competency, authority and responsibility within the enterprise.

Systems used for identity and access management include single sign-on systems, multi-factor authentication and privileged access management (PAM).

These technologies also provide the ability to securely store identity and profile data as well as data governance functions to ensure that only data that is necessary and relevant is shared. IAM systems can be deployed on premises, provided by a third-party vendor through a cloud-based subscription model or deployed in a hybrid cloud.

Basic components of IAM

On a fundamental level, IAM encompasses the following components:

- How individuals are identified in a system.
- How roles are identified in a system and how they are assigned to individuals.
- Adding, removing and updating individuals and their roles in a system.
- Assigning levels of access to individuals or groups of individuals.
- Protecting the sensitive data within the system and securing the system itself.

What IAM systems should include

Identity access management systems should consist of all the necessary controls and tools to capture and record user login information, manage the enterprise database of user identities and orchestrate the assignment and removal of access privileges. That means that systems used for IAM should provide a centralized directory service with oversight as well as visibility into all aspects of the company user base.

Technologies for identity access and management should simplify the user provisioning and account setup process. These systems should reduce the time it takes to complete these processes with a controlled workflow that decreases errors as well as the potential for abuse while allowing automated account fulfillment. An identity and access management system should also allow administrators to instantly view and change access rights.

These systems also need to balance the speed and automation of their processes with the control that administrators need to monitor and modify access rights. Consequently, to manage access requests, the central directory needs an access rights system that automatically matches employee job titles, business unit identifiers and locations to their relevant privilege levels.

Multiple review levels can be included as workflows to enable the proper checking of individual requests. This simplifies setting up appropriate review processes for higher-level access as well as easing reviews of existing rights to prevent privilege creep, the gradual accumulation of access rights beyond what users need to do their jobs.

IAM systems should be used to provide flexibility to establish groups with specific privileges for specific roles so that access rights based on employee job functions can be uniformly assigned. The system should also provide request and approval

processes for modifying privileges because employees with the same title and job location may need customized, or slightly different, access.

Benefits of identity and access management

IAM technologies can be used to initiate, capture, record and manage user identities and their related access permissions in an automated manner. This brings an organization the following benefits:

- Access privileges are granted according to one interpretation of policy and all individuals and services are properly authenticated, authorized and audited.
- Companies that properly manage identities have greater control of user access, reducing the risk of internal and external data breaches.
- Automating IAM systems allows businesses to operate more efficiently by decreasing the effort, time and money that would be required to manage access to their networks manually.
- In terms of security, the use of an IAM framework can make it easier to enforce policies around user authentication, validation and privileges and address issues regarding privilege creep.
- IAM systems help companies better comply with government regulations by allowing them to show that corporate information is not being misused. Companies can also demonstrate that any data needed for auditing can be made available on-demand.

Additionally, by implementing identity access management tools and following related best practices, a company can gain a competitive edge. For example, IAM technologies allow the business to give users outside the organization, like customers, partners, contractors and suppliers, access to its network across mobile applications, on-premises apps and software-as-a-service apps without compromising security. This enables better collaboration, enhanced productivity, increased efficiency and reduced operating costs.

IAM in the enterprise

It can be challenging to get funding for IAM projects because they do not directly increase an organization's profitability or functionality. However, a lack of effective identity and access management poses significant risks not only to compliance, but also overall security. These mismanagement issues increase the risk of greater damages from both external and internal threats.

Keeping the required flow of business data going while simultaneously managing its access has always required administrative attention. The business IT environment is ever evolving and the difficulties have only become greater with recent disruptive trends like bring your own device, cloud computing, mobile apps and an increasingly mobile workforce. There are more devices and services to be managed than ever before, with diverse requirements for associated access privileges.

Risks associated with IAM

Implementing proper identity and access management tools or platforms means storing all authorizations and credentials in one, unified place. When not secured

correctly, this can be a huge risk because if an attacker gains access to the system, all digital identities can be compromised. Similarly, if a specific employee that is authorized to the system does not follow security or password best practices, all of the information could be easily leaked.

Another concern for adopting IAM are challenges in implementation. Legacy systems will typically already have an identity management functionality in place, therefore, converting resources to a new system could be challenging, expensive and time-consuming. However, solutions for minimizing the need of technical support, such as cloud services, are becoming more viable.

IAM vendors and tools

Rather than developing internal tools, most companies decide to purchase or subscribe to third-party IAM tools. These products can take on multiple forms, such as an identity as a service (IDaaS) cloud model, a hybrid cloud model, a traditional on-premise model or a microservices model. IAM microservices may cover only one aspect of IAM like privileged account management, account compliance management or user authorization management.

Vendors with products in the IAM space include:

- Microsoft Azure Active Directory.
- IBM Security Identity and Access Assurance.
- Oracle Identity Cloud Service.
- Okta.
- Centrify.
- RSA SecurID Access.

Autonomic Security.

Autonomic computing is a computing model in which the system is self-healing, self-configured, self-protected and self-managed. Designed to mimic the human body's nervous system-in that the autonomic nervous system acts and reacts to stimuli independent of the individual's conscious input-an autonomic computing environment functions with a high level of artificial intelligence while remaining invisible to the users.

The autonomic computing initiative (ACI), which was developed by IBM, demonstrates and advocates networking **computer systems that do not involve a lot of human intervention other than defining input rules.** The ACI is derived from the autonomic nervous system of the human body.

IBM has defined the four areas of automatic computing:

- Self-Configuration.
- Self-Healing (error correction).
- Self-Optimization (automatic resource control for optimal functioning).

Self-Protection (identification and AC was **designed to mimic the human body's nervous system**-in that the autonomic nervous system acts and reacts to stimuli independent of the individual's conscious input-an autonomic computing

environment functions with a high level of **artificial intelligence** while remaining invisible to the users. Just as the human body acts and responds without the individual controlling functions (e.g., internal temperature rises and falls, breathing rate fluctuates, glands secrete hormones in response to stimulus), the autonomic computing environment operates organically in response to the input it collects.

IBM has set forth eight conditions that define an autonomic system:

1. The system must know itself in terms of what resources it has access to, what its capabilities and limitations are and how and why it is connected to other systems.
2. The system must be able to automatically configure and reconfigure itself depending on the changing computing environment.
3. The system must be able to optimize its performance to ensure the most efficient computing process.
4. The system must be able to work around encountered problems by either repairing itself or routing functions away from the trouble.
5. The system must detect, identify and protect itself against various types of attacks to maintain overall system security and integrity.
6. The system must be able to adapt to its environment as it changes, interacting with neighboring systems and establishing communication protocols.
7. The system must rely on open standards and cannot exist in a proprietary environment.
8. The system must anticipate the demand on its resources while keeping transparent to users.

Autonomic computing is one of the building blocks of ***pervasive computing***, an anticipated future computing model in which tiny – even invisible – computers will be all around us, communicating through increasingly interconnected networks leading to the concept of The Internet of Everything (IoE). Many industry leaders are researching various components of autonomic computing.