Unit 1

Chapter 1

Unit Structure

1.0 Objective

This chapter would make your under the concept of following concepts

- What is a cloud computing?
- What are characteristics and benefits of cloud computing?
- It's Challenges.
- Historical development of technologies toward the growth of cloud computing
- Types of Cloud Computing Models.
- Different types of Services in the Cloud Computing.
- Application development and Infrastructure and system development technologies about the Cloud Computing.
- Overview of different sets of Cloud Service Providers.

1.1 Introduction

Historically, computing power was a scarce, costly tool. Today, with the emergence of cloud computing, it is plentiful and inexpensive, causing a profound paradigm shift — a transition from scarcity computing to abundance computing. This computing revolution accelerates the commoditization of products, services and business models and disrupts current information and communications technology (ICT) Industry .It supplied the services in the same way to water, electricity, gas, telephony and other appliances. Cloud Computing offers on-demand computing, storage, software and other IT services with usage-based metered payment. Cloud Computing helps re-invent and transform technological partnerships to improve marketing, simplify and increase security and increasing stakeholder interest and consumer experience while reducing costs. With cloud computing, you don't have to over-provision resources to manage potential peak levels of business operation. Then, you have the resources you really required. You can scale these resources to expand and shrink capability instantly as the business needs evolve. This chapter offers a brief summary of the trend of cloud computing by describing its vision, addressing its key features, and analyzing technical advances that made it possible. The chapter also introduces some key cloud computing technologies and some insights into cloud computing environments.

1.2 Cloud computing at a glance

The notion of computing in the "cloud" goes back to the beginnings of utility computing, a term suggested publicly in 1961 by computer scientist John McCarthy:

"If computers of the kind I have advocated become the computers of the future, then computing may someday be organized as a public utility just as the telephone system is a public utility… The computer utility could become the basis of a new and important industry."

The chief scientist of the Advanced Research Projects Agency Network (ARPANET), Leonard Kleinrock, said in 1969:

"as of now, computer networks are still in their infancy, but as they grow up and become sophisticated, we will probably see the spread of 'computer utilities' which, like present electric and telephone utilities, will service individual homes and offices across the country."

This vision of the computing utility takes form with cloud computing industry in the 21st century. The delivery of computing services is easily available on demand just like other utilities services such as water, electricity, telephone and gas in today's society are available. Likewise, users (consumers) only have to pay service providers if they have access to computing resources. Instead of maintaining their own computing systems or data centers, customer can lease access from cloud service providers to applications and storage. The advantage of using cloud computing services is that organizations can avoid the upfront cost and difficulty of running and managing their own IT infrastructure and pay for when they use it. Cloud providers can benefit from large economies of scale by offering the same services to a wide variety of customers.

In the case, consumers can access the services according to their requirement with the knowing where all their services are hosted. These model can called as utility computing as cloud computing. As cloud computing called as utility computing because users can access the

infrastructure as a “cloud” as application as services from anywhere part in the world. Hence Cloud computing can be defined as a new dynamic provisioning model of computing services that improves the use of physical resources and data centers is growing uses virtualization and convergence to support multiple different systems that operate on server platforms simultaneously. The output achieved with different placement schemes of virtual machines will differ a lot. .

By observing advancement in several technologies , we can track of cloud computing that is (virtualization, multi-core chips), especially in hardware; Internet (Web services, service-oriented architectures, Web 2.0), Distributed computing (clusters, grids), and autonomous Computing, automation of the data center). The convergence of Figure 1.1 reveals the areas of technology that have evolved and led to the advent Cloud computing. Any of these technologies were considered speculation at an early stage of development; however, they received considerable attention later Academia and big business companies have been prohibited. Therefore, a Process of specification and standardization followed which resulted in maturity and wide adoption. . The rise of cloud computing is closely associated with the maturity of these technologies.

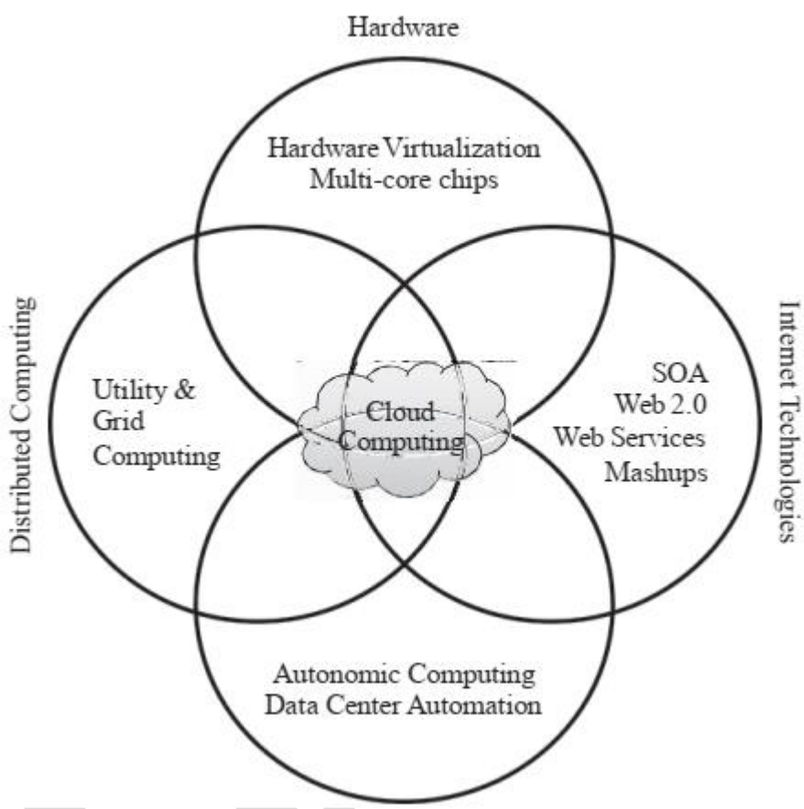


FIGURE 1.1. Convergence of various advances leading to the advent of cloud computing

1.2.1 The vision of cloud computing

The virtual provision of cloud computing is hardware, runtime environment and resources for a user by paying money. As of these items can be used as long as the User, no upfront commitment requirement. The whole computer device collection is turned into a Utilities set that can be supplied and composed in hours rather than days together, to deploy devices without Costs for maintenance. A cloud computer's long-term vision is that IT services are traded without technology and as utilities on an open market as barriers to the rules.

We can hope in the near future that it can be identified the solution that clearly satisfies our needs entering our application on a global digital market services for cloud computing. This market will make it possible to automate the process of discovery and integration with its existing software systems. A digital cloud trading platform is available services will also enable service providers to boost their revenue. A cloud service may also be a competitor's customer service to meet its consumer commitments.

Company and personal data is accessible in structured formats everywhere, which helps us to access and communicate easily on an even larger level. Cloud computing's security and stability will continue to improve, making it even safer with a wide variety of techniques. Instead of concentrating on what services and applications they allow, we do not consider "cloud" to be the

most relevant technology. The combination of the wearable and the bringing your own device (BYOD) with cloud technology with the Internet of Things (IOT) would become a common necessity in person and working life such that cloud technology is overlooked as an enabler.

Figure 1.2. Cloud computing vision.
(Reference from "Mastering Cloud Computing Foundations and Applications Programming" by Rajkumar Buyya)

1.2.2 Defining a cloud

The fairly recent motto in the IT industry "cloud computing," which came into being after many decades of innovation in virtualization, utility computing, distributed computing, networking and software services. A cloud establishes an IT environment invented to provide measured and scalable resources remotely. It has evolved as a modern model for information exchange and internet services. This provides more secure, flexible and scalable services for consumers. It is used as a service-oriented architecture that reduces end-user overhead information.

Figure 1.3 illustrates the variety of terms used in current cloud computing definitions.

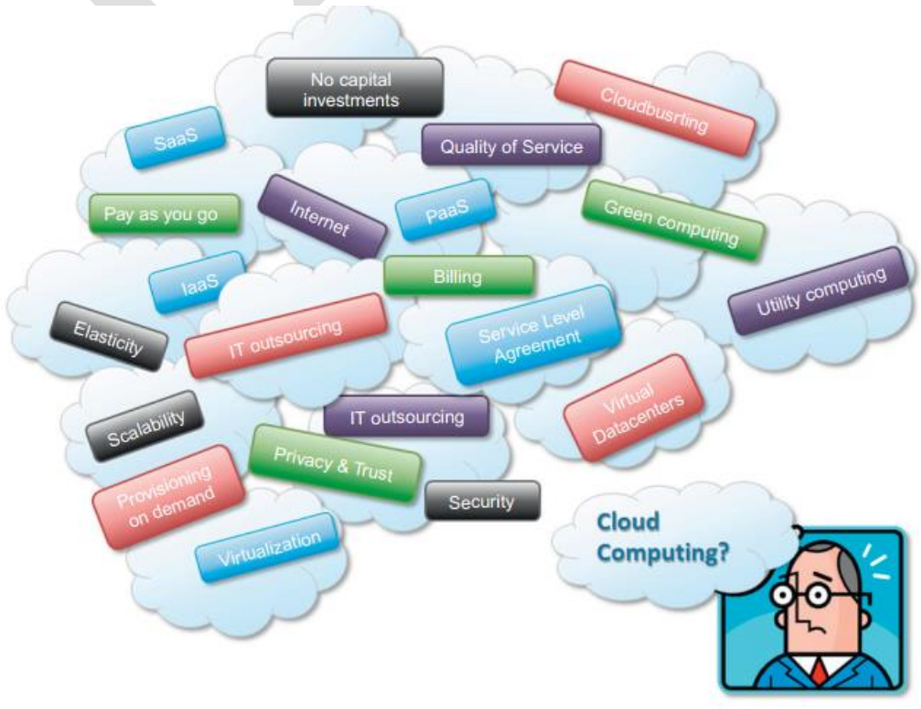


FIGURE 1.3 Cloud computing technologies, concepts, and ideas.

(Reference from "Mastering Cloud Computing Foundations and Applications Programming" by Rajkumar Buyya)

Internet plays a significant role in cloud computing for representing a transportation medium of cloud services which can deliver and accessible to cloud consumer. According to the definition given by Armbrust

Cloud computing refers to both the applications delivered as services over the Internet and the hardware and system software in the datacenters that provide those services

Above definition indicated about the cloud computing which touching upon entire stack from underlying hardware to high level software as service. It introduced with the concept of *everything as service* called as *Xaas* where different part of the system like IT Infrastructure , development platform for an application ,storage ,databases and so on can be delivered as services to the cloud consumers and consumers has to paid for the services what they want. This new paradigms of the technologies not only for the development of the software but also how the user can deploy the application ,make the application accessible and design of IT infrastructure and how this companies allocate the costs for IT needs. This approach encourage the cloud computing form global point of views that one single user can upload the documents in the cloud and on the others side Company owner want to deploy the entire infrastructure in the public cloud. According to the definition proposed by the U.S. National Institute of Standards and Technology (NIST):

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

Another approach of cloud computing is "utility computing" where could computing mainly focus on delivering services based upon the pricing model it called as "*pay-per-use*" strategy. Cloud computing make all the resources online mode such as storage, you can lease virtual hardware or you can use the resource for the application development and users has to pay according to their usage their will no or minimal amount of upfront cost. All this above operations are performed and user have to pay the bill by simply entering the credit card details and accesses this services through the web browsers. According to George Reese

He have defined three criteria on whether a particular service is a cloud service:

- The service is accessible via a web browser (nonproprietary) or web services API.
- Zero capital expenditure is necessary to get started.
- You pay only for what you use as you use it.

Many cloud service providers provides the cloud services freely to the users but some enterprise class services can be provided by the cloud service providers based upon specific pricing schemes where users have to subscribe with the service provider on which a service level agreement (SLA) is defined based on the quality parameters between the cloud service providers and user and cloud service providers has to delivered the services according the service level agreement (SLA)

RajKumar Buyya defined cloud computing based on the nature of utility computing

A cloud is a type of parallel and distributed system consisting of a collection of interconnected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on service-level agreements established through negotiation between the service provider and consumers.

1.2.3 A closer look

Cloud computing is useful in governments, enterprises, public and private institutions and research organizations which make more effective and demand-driven computing services systems. There seem to be a number of specific examples demonstrating emerging applications of cloud computing in both established companies and startups. Such cases are intended to illustrate the value proposition of viable cloud computing solutions and the benefits businesses have gained from these services.

NewYork Times : One of the most widely known examples of cloud computing commitment comes from New York Times . The New York Times has collected a large number of high-resolution scanned images of historical newspapers, ranging from 1851-1922. They want to process this set of images into separate articles in PDF format. Using 100 EC2 instances, they can complete the processing within 24 hours at a total cost of $ 890 (EC2 calculation time is $ 240, S3 data transfer and storage use is $ 650, storage and transfer of 4.0TB source image and 1.5TB Output

PDF). Derek Gottfrid pointed out: "Actually, it worked so well that we ran it twice, because after the completion we found an error in the PDF."

The New York Times had the option to utilize 100 servers for 24 hours at the low standard cost of ten cent an hour for every server. In the event that the New York times had bought even a solitary server for this errand, the probable expense would have surpassed the $890 for simply the hardware, and they likewise need to think about the expense of administration, power and cooling Likewise, the handling would have assumed control more than a quarter of a year with one server. On the off chance that the New York Times had bought four servers, as Derek Gottfrid had considered, it would have still taken almost a month of calculation time. The quick turnaround time (sufficiently quick to run the activity twice) and endlessly lower cost emphatically represents the prevalent estimation of cloud services.

Washington Post : In a related but more latest event, the Washington Post were able to transform 17,481 pages of scanned document images into a searchable database in just a day using Amazon EC2. On March 19th at 10am, Hillary Clinton's official White House schedule from 1993-2001 was published to the public as a large array of scanned photographs (in PDF format, but non-searchable). Washington Post programmer Peter Harkins utilized 200 Amazon EC2 instances to conduct OCR (Optical Character Recognition) on the scanned files to create searchable text – " I used 1,407 hours of virtual machine time with a total cost of $144.62. We find it a positive proof of concept.

DISA : Federal Computer Week mentioned that the Defense Information Systems Agency (DISA) as compared the cost of the usage of Amazon EC2 versus internally maintained servers : "In a latest take a look at, the Defense Information Systems Agency in comparison the price of growing a simple application known as the Tech Early Bird on $30,000 well worth of in-house servers and software program with the costs of growing the equal application using the Amazon Elastic Compute Cloud from Amazon Web Services. Amazon charged 10 cents an hour for the provider, and DISA paid a total of $5 to expand a software that matched the overall performance of the in-house application.

SmugMug : SmugMug, an image posting and hosting web site like Flickr , stores a substantial level of its photo information in Amazon's S3 cloud storage service . In 2006, they ended up saving "$500,000 in prepared disk drive expenses in 2006 and reduce its disk storage space array costs in half" through the use of Amazon S3. Based on the CEO of SmugMug, they might "easily save a lot more than $1 million" in the next year through the use of S3. The CEO known that their present growth rate during the article necessitates about $80,000 worth of new hardware, and the regular costs boost even more considerably after putting "power, cooling, the info center space, along with the manpower had a need to manage them." On the other hand, Amazon S3 costs around $23,000 per month for equivalent storage which is all-inclusive (power, maintenance, cooling, etc. are figured into the expense of the storage.

Eli Lily : Eli Lily, among the largest pharmaceutical companies, is needs to utilize Amazon's storage and compute clouds to supply on-demand high-performance processing for research reasons . John Foley highlights, "it accustomed to acquire Eli Lilly seven and a half weeks to deploy a server internally" whereas Amazon can provision a virtual server in 3 minutes. Furthermore "a 64-node Linux cluster could be online in 5 minutes (compared against 90 days internally)." Amazon's cloud providers not only deliver on-demand scaling and usage-based billing, they enable Eli Lily to respond with considerably amplified agility in addition, eliminating time-consuming products deployment and acquisition functions.

Best Buy's Giftag: Best Buy's Giftag is a new online wish-list service hosted by Google's App Engine. In a video interview, the developers suggested that they were beginning to build a platform with a different technology and moved to Google App Engine with its superior speed of development and scaling advantages. As one developer eloquently stated it, "a lot of the work that none of us even needs to do is [already] completed for us." The developers also lauded App Engine 's design to allow effortless scaling; App Engine-based web apps inherit Google's best-in - class technologies and expertise in running large-scale websites. By the end of the day, App Engine helps developers to focus on building site-specific separated features: "Not worried with the operational aspects of an application going away always frees you to create excellent code or evaluate your code better.

TC3 : TC3 (Total Claims Capture & Control) is a healthcare services company imparting claims management solution. TC3 now makes use of Amazon's cloud services to allow on-demand scaling of resource and lower infrastructure costs . TC3's CTO notes, "we're making use of Amazon S3, EC2, and SQS to permit our claim processing capacity to growth and reduce as required to satisfy our service level agreements (SLAs). There are times we require massive quantities of computing resource that a long way exceed our machine capacities and when these

conditions took place inside the past our natural response became to name our hardware vendor for a quote. Now, by using the usage of AWS products, we can dramatically reduce our processing time from weeks or months right down to days or hours and pay much less than shopping, housing and maintaining the servers ourselves" Another particular feature of TC3 's activities is that, because they provide US health-related services, they are obligated to abide with the HIPPA (Health Insurance Portability and Accountability Act). Regulatory compliance is one of the main obstacles facing corporate adoption of cloud infrastructure – the fact that TC3 is capable of complying with HIPPA on Amazon's platform is significant.

How all of the computing made possible? in same IT services on demand like computing power , storage and providing an runtime environments for development of an applications on pay-as-you go basis .cloud computing not only provides an opportunity for easily accessing of IT services as per demand , but also provides newly ideas regarding IT Services and resources as am utilities .Figure 1.4 provides a bird's-eye view of cloud computing

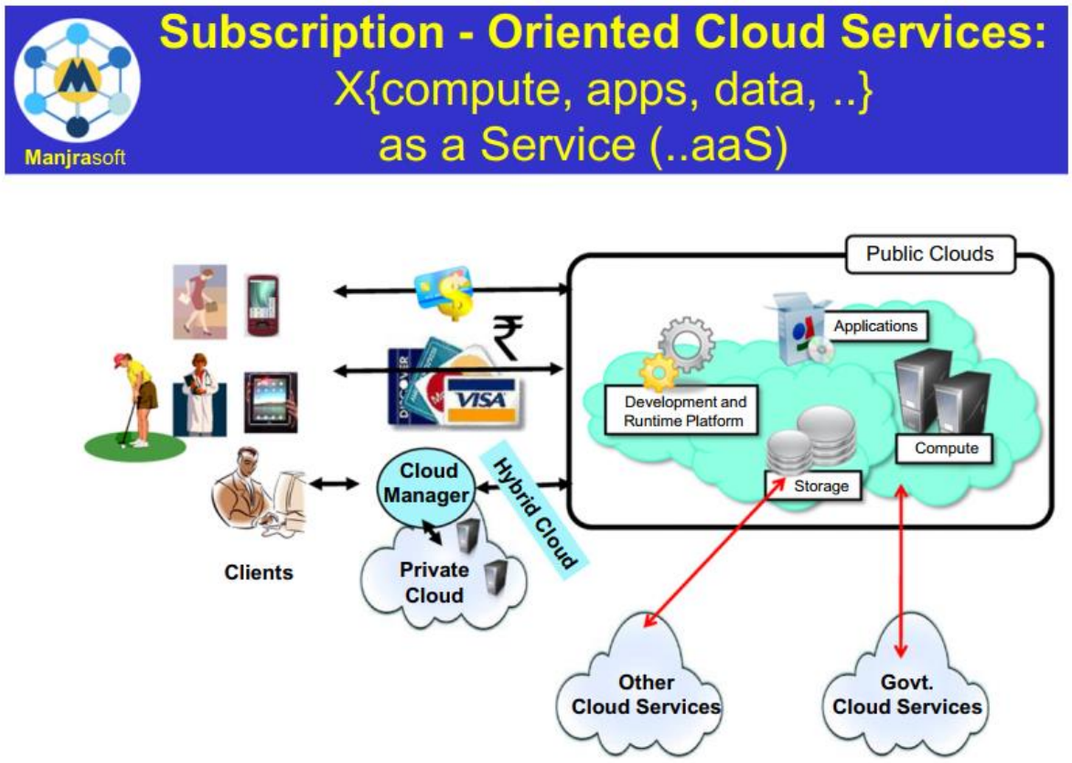


FIGURE 1.4 A bird's-eye view of cloud computing
(Reference from "Mastering Cloud Computing Foundations and Applications Programming" by Rajkumar Buyya)

There are three deployment models for accessing the services of cloud computing environment are public, private and hybrids clouds (see Figure 1.5). The *public cloud* is one of the most common deployment models in which computing services is offered by third-party vendors that the consumer are able to access and purchase the resource from the public cloud via the public internet. These can be free or on-demand, meaning that consumers pay for their CPU cycles, storage or bandwidth per use. Public clouds will save companies from the expensive procurement, management and on-site maintenance of hardware and application infrastructure — all management and maintenance of the system is held to responsibility of the cloud service provider. *Public clouds* can also be deployed faster than on-site infrastructures with a platform almost constantly scalable. Although security issues have been posed by public cloud implementations, the public cloud could be as secure as the most efficiently operated private cloud deployment when it is implemented correctly. A *private cloud* is an essentially one organization's cloud service. In using a *private cloud*, the advantages of cloud computing are experienced without sharing resources with other organizations. There can be a private cloud within an organization, or be controlled from a third party remotely, and accessed via the Internet (but it is not shared with others, unlike a public cloud). Private cloud incorporates several of the advantages of cloud computing — including elasticity, scalability and easy service delivery — with the on-site control, security, and resource customization .Many companies select private cloud over public cloud (cloud computing services delivered through multi-customer infrastructure) because private cloud is a simpler (or the only way) way to satisfy their regulatory compliance requirements. Others prefer private cloud because their workloads deal with confidential information, intellectual property, and personally identifiable information (PII), medical records, financial data and other sensitive data. *Hybrid cloud* is an infrastructure that contains links between a user's cloud (typically referred to as "*private cloud*") and a third-party cloud (typically referred to as "*public cloud*").

Whilst the private and public areas of the hybrid cloud are linked, they remain unique. This allows a hybrid cloud to simultaneously offer the advantages of several implementation models. The sophistication of hybrid clouds is very different. Some hybrid clouds, for example, only connect the on-site to public clouds. The operations and application teams are responsible for all the difficulties inherent in the two different infrastructures.

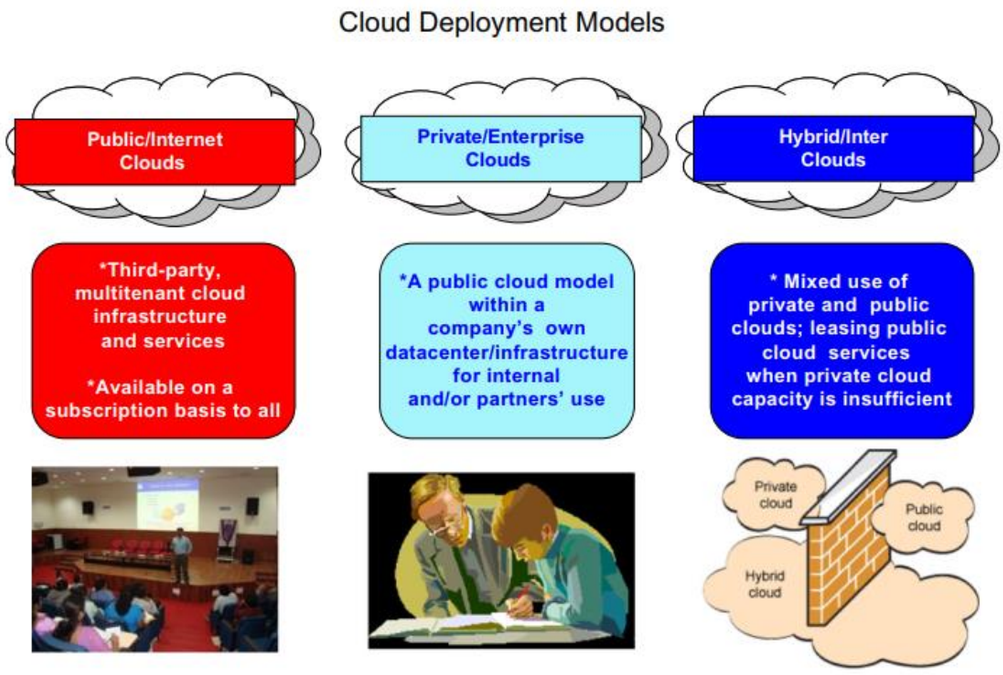


FIGURE 1.5 Major deployment models for cloud computing.
Reference from "Mastering Cloud Computing Foundations and Applications Programming" by Rajkumar Buyya)

1.2.4 The cloud computing reference model

A model that characterizes and standardizes the functions of a cloud computing environment, is the cloud reference model. This is a basic benchmark for cloud computing development. The growing popularity of cloud computing has expanded the definitions of different cloud computing architectures. The cloud environment has a wide range of vendors and multiple offer definitions which make the evaluation of their services very hard. The way the cloud functions and interacts with other technology can be a little confusing with such complexity in its implementation.

A standard cloud reference model for architects, software engineers, security experts and businesses is required to achieve the potential of cloud computing. This cloud landscape is controlled by the Cloud Reference Model. Figure 1.6 displays various cloud providers and their innovations in the cloud services models available on the market.

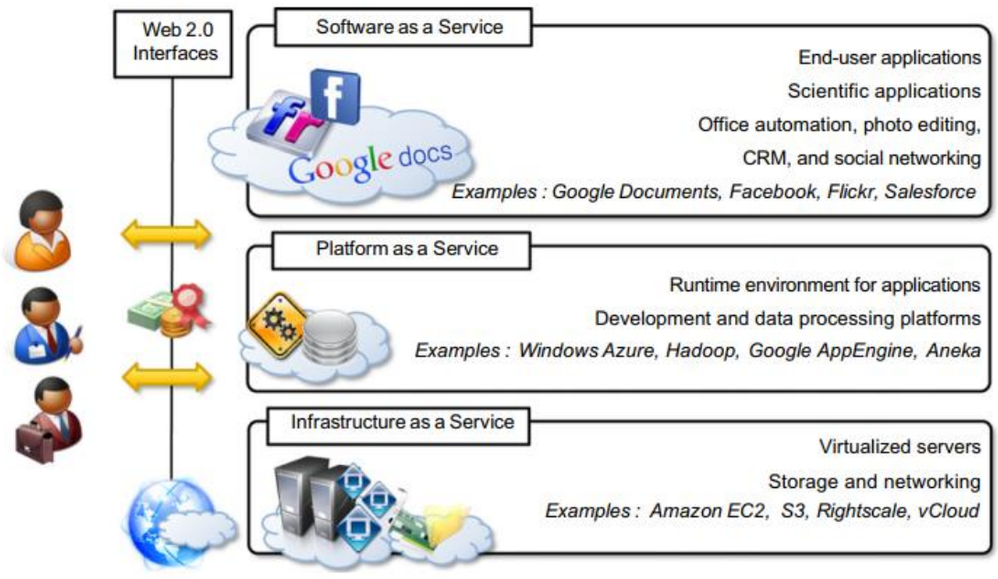


FIGURE 1.6 The Cloud Computing Reference Model.

Reference from "Mastering Cloud Computing Foundations and Applications Programming" by Rajkumar Buyya)

Cloud computing is an all-encompassing term for all resources that are hosted on the Internet. These services are classified under three main categories: infrastructure as a service (IaaS), platform as a service (PaaS) and software as a service (SaaS). These categories are mutually related as outlined in Figure 1.6 which gives an organic view of cloud computing. The model structures the broad variety of cloud computing services in a layered view from the base to the top of the computing stack.

At the stack foundation, *Infrastructure as Service* (IaaS) is the most common cloud computing service model, offering the basic infrastructure of virtual servers, networks, operating systems and storage drives. This provides the flexibility, reliability and scalability many companies seek with the cloud and eliminates the need for the office hardware. This makes it a perfect way to promote business growth for SMEs looking for a cost-effective IT way. IaaS is a completely outsourced pay-for-use service that can be run in a public, private or hybrid infrastructure.

The next step in the stack is *platform-as-a-service* (PaaS) solutions. Cloud providers deploy the software and infrastructure framework, but companies can develop and run their own apps. Web applications can easily and quickly be created via PaaS with the flexibility and robustness of the service to support it. PaaS solutions are scalable and suitable if multiple developers work on a single project. It is also useful when using an established data source (such as a CRM tool).

Top of the stack, *Software as a Service* (SaaS) This cloud computing solution includes deploying Internet-based software to different companies paying via subscription or a paid-per-use model. It is an important tool for CRM and applications which require a great deal of Web or mobile access – such as software for mobile sales management. SaaS is managed from a centralized location so that companies need not be worried about its own maintenance and is ideal for short-term projects.

The big difference in control between PaaS and IaaS is users got. Essentially, PaaS makes it possible for suppliers to manage everything IaaS calls for more customer management. In general, companies with a software package or application already have specific purpose and you should choose to install and run it in the cloud IaaS rather than PaaS.

1.2.5 Characteristics and benefits

As both commercially and technologically mature cloud computing services, companies will be easier to maximize their potential benefits. However, it is equally important to know what cloud computing is and what it does.

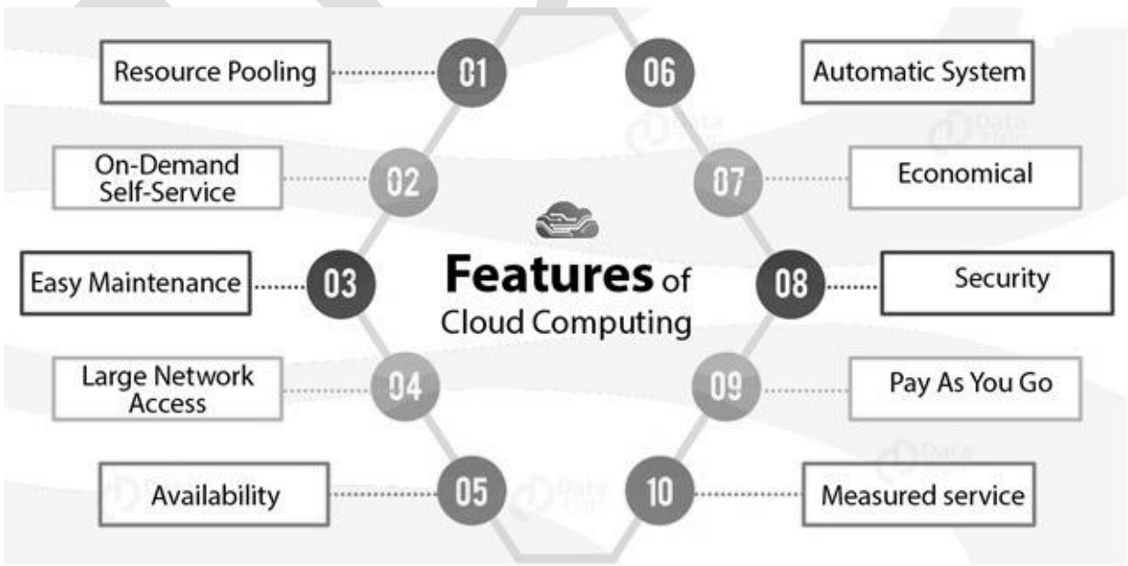


FIGURE 1. 7 Features of Cloud Computing

Following are the characteristics of Cloud Computing:

1. **Resources Pooling**

   This means that the Cloud provider used a multi-leaner model to deliver the computing resources to various customers. There are various allocated and reassigned physical and virtual resources, which rely on customer demand. In general, the customer

has no control or information about the location of the resources provided, but can choose location on a higher level of abstraction.

2. On-Demand Self-Service

This is one of the main and useful advantages of Cloud Computing as the user can track server uptimes, capability and network storage on an ongoing basis. The user can also monitor computing functionalities with this feature.

3. Easy Maintenance

The servers are managed easily and the downtime is small and there are no downtime except in some cases. Cloud Computing offers an update every time that increasingly enhances it. The updates are more system friendly and operate with patched bugs faster than the older ones.

4. Large Network Access

The user may use a device and an Internet connection to access the cloud data or upload it to the cloud from anywhere. Such capabilities can be accessed across the network and through the internet.

5. Availability

The cloud capabilities can be changed and expanded according to the usage. This review helps the consumer to buy additional cloud storage for a very small price, if necessary.

6. Automatic System

Cloud computing analyzes the data required automatically and supports a certain service level of measuring capabilities. It is possible to track, manage and report the usage. It provides both the host and the customer with accountability**.**

7. Economical

It is a one-off investment since the company (host) is required to buy the storage, which can be made available to many companies, which save the host from monthly or annual costs. Only the amount spent on the basic maintenance and some additional costs are much smaller.

8. Security

Cloud Security is one of cloud computing's best features. It provides a snapshot of the data stored so that even if one of the servers is damaged, the data cannot get lost. The information is stored on the storage devices, which no other person can hack or use. The service of storage is fast and reliable**.**

9. Pay as you go

Users only have to pay for the service or the space in cloud computing. No hidden or additional charge to be paid is liable to pay. The service is economical and space is often allocated free of charge.

10. Measured Service

Cloud Computing resources that the company uses to monitor and record. This use of resources is analyzed by charge-per-use capabilities. This means that resource use can be measured and reported by the service provider, either on the virtual server instances running through the cloud. You will receive a models pay depending on the manufacturing company's actual consumption.

1.2.6 Challenges ahead

All has advantages and challenges. We saw many Cloud features and it’s time to identify the Cloud computing challenges with tips and techniques you can identify all your own. Let's therefore start to explore cloud computing risk and challenges. Nearly all companies are using cloud computing because companies need to store the data. The companies generate and store a tremendous amount of data. Thus, they face many security issues. Companies would include establishments to streamline and optimize the process and to improve cloud computing management.

This is a list of all cloud computing threats and challenges:

1. Security & Privacy
2. Interoperability & Portability
3. Reliable and flexible
4. Cost
5. Downtime
6. Lack of resources
7. Dealing with Multi-Cloud Environments
8. Cloud Migration
9. Vendor Lock-In
10. Privacy and Legal issues

1. **Security and Privacy of Cloud**

   The cloud data store must be secure and confidential. The clients are so dependent on the cloud provider. In other words, the cloud provider must take security measures necessary to secure customer data. Securities are also the customer's liability because they must have a good password, don't share the password with others, and update our password on a regular basis. If the data are outside of the firewall, certain problems may occur that the cloud provider can eliminate. Hacking and malware are also one of the biggest problems because they can affect many customers. Data loss can result; the encrypted file system and several other issues can be disrupted.

2. **Interoperability and Portability**

   Migration services into and out of the cloud shall be provided to the Customer. No bond period should be allowed, as the customers can be hampered. The cloud will be capable of supplying premises facilities. Remote access is one of the cloud obstacles, removing the ability for the cloud provider to access the cloud from anywhere.

3. **Reliable and Flexible**

   Reliability and flexibility are indeed a difficult task for cloud customers, which can eliminate leakage of the data provided to the cloud and provide customer trustworthiness. To overcome this challenge, third-party services should be monitored and the performance, robustness, and dependence of companies supervised.

4. **Cost**

   Cloud computing is affordable, but it can be sometimes expensive to change the cloud to customer demand. In addition, it can hinder the small business by altering the cloud as demand can sometimes cost more. Furthermore, it is sometimes costly to transfer data from the Cloud to the premises.

5. **Downtime**

   Downtime is the most popular cloud computing challenge as a platform free from downtime is guaranteed by no cloud provider. Internet connection also plays an important role, as it can be a problem if a company has a nontrustworthy internet connection, because it faces downtime.

6. **Lack of resources**

   The cloud industry also faces a lack of resources and expertise, with many businesses hoping to overcome it by hiring new, more experienced employees. These employees will not only help solve the challenges of the business but will also train existing employees to benefit the company. Currently, many IT employees work to enhance cloud computing skills and it is difficult for the chief executive because the employees are little qualified. It claims that employees with exposure of the latest innovations and associated technology would be more important in businesses.

7. Dealing with Multi-Cloud Environments

Today not even a single cloud is operating with full businesses. According to the RightScale report revelation, almost 84 percent of enterprises adopt a multi-cloud approach and 58 percent have their hybrid cloud approaches mixed with the public and private clouds. In addition, five different public and private clouds are used by organizations.

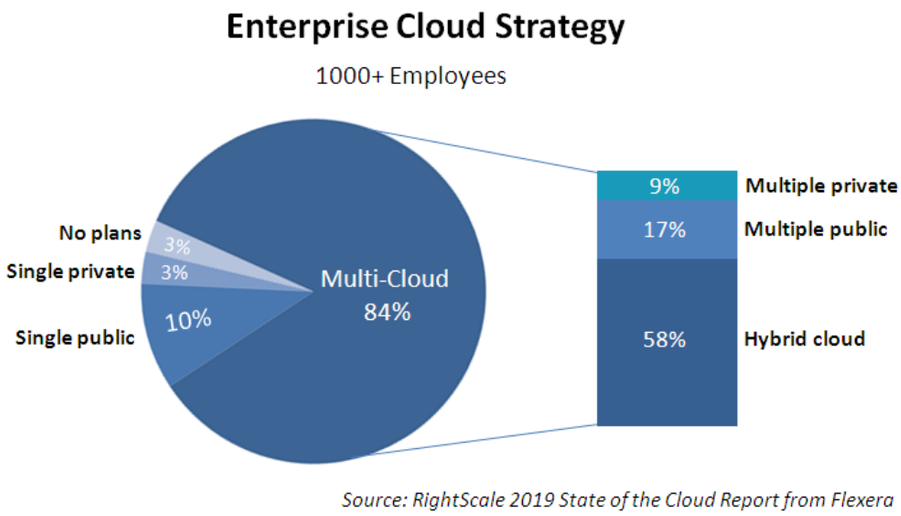


FIGURE 1. 8 RightScale 2019 report revelation

The teams of the IT infrastructure have more difficulty with a long-term prediction about the future of cloud computing technology. Professionals have also suggested top strategies to address this problem, such as rethinking processes, training personnel, tools, active vendor relations management, and the studies.

8. Cloud Migration

While it is very simple to release a new app in the cloud, transferring an existing app to a cloud computing environment is harder. 62% said their cloud migration projects are harder than they expected, according to the report. In addition, 64% of migration projects took longer than expected and 55% surpassed their budgets. In particular, organizations that migrate their applications to the cloud reported migration downtime (37%), data before cutbacks synchronization issues (40%), migration tooling problems that work well (40%), slow migration of data (44%), security configuration issues (40%), and time-consuming troubleshooting (47%). And to solve these problems, close to 42% of the IT experts said that they wanted to see their budget increases and that around 45% of them wanted to work at an in-house professional, 50% wanted to set the project longer, 56% wanted more pre-migration tests.

9. Vendor lock-in

The problem with vendor lock-in cloud computing includes clients being reliant (i.e. locked in) on the implementation of a single Cloud provider and not switching to another vendor without any significant costs, regulatory restrictions or technological incompatibilities in the future. The lock-up situation can be seen in apps for specific cloud platforms, such as Amazon EC2, Microsoft Azure, that are not easily transferred to any other cloud platform and that users are vulnerable to changes made by their providers to further confirm the lenses of a software developer. In fact, the issue of lock-in arises when, for example, a company decide to modify cloud providers (or perhaps integrate services from different providers), but cannot move applications or data across different cloud services, as the semantics of cloud providers' resources and services do not correspond. This heterogeneity of cloud semantics and APIs creates technological incompatibility which in turn leads to challenge interoperability and portability. This makes it very complicated and difficult to interoperate, cooperate, portability, handle and maintain data and services. For these reasons, from the point of view of the company it is important to maintain flexibility in changing providers according to business needs or even to maintain in-house certain components which are less critical to safety due to risks. The issue of supplier lock-in will prevent interoperability and portability between cloud providers. It is the way for cloud providers and clients to become more competitive.

10. Privacy and Legal issues

Apparently, the main problem regarding cloud privacy/data security is 'data breach.'

Infringement of data can be generically defined as loss of electronically encrypted personal information. An infringement of the information could lead to a multitude of losses both for the provider and for the customer; identity theft, debit/credit card fraud for the customer, loss of credibility, future prosecutions and so on. In the event of data infringement, American law requires notification of data infringements by affected persons. Nearly every State in the USA now needs to report data breaches to the affected persons. Problems arise when data are subject to several jurisdictions, and the laws on data privacy differ. For example, the Data Privacy Directive of the European Union explicitly states that 'data can only leave the EU if it goes to a 'additional level of security' country.' This rule, while simple to implement, limits movement of data and thus decreases data capacity. The EU's regulations can be enforced.

1.3 Historical developments

No state-of-the-art technology is cloud computing. The development of Cloud Computing through various phases, including Grid Computing, Utility Computing, Application Service Provision and Software as a Service, etc., has taken place. But the overall (whole) concept of the provision of computing resources via a global network began in the 1960s. By 2020, it is projected that the cloud computing market will exceed 241 billion dollars. But the history of cloud computing is how we got there and where all that started. Cloud computing has a history that is not that old, the first business and consumer cloud computing website was launched in 1999 (Salesforce.com and Google). Cloud computing is directly connected to Internet development and the development of corporate technology as cloud computing is the answer to the problem of how the Internet can improve corporate technology. Business technology has a rich and interesting background, almost as long as businesses themselves, but the development that has influenced Cloud computing most directly begins with the emergence of computers as suppliers of real business solutions.

History of Cloud Computing

Cloud computing is one of today's most breakthrough technology. Then there's a brief cloud-computing history.

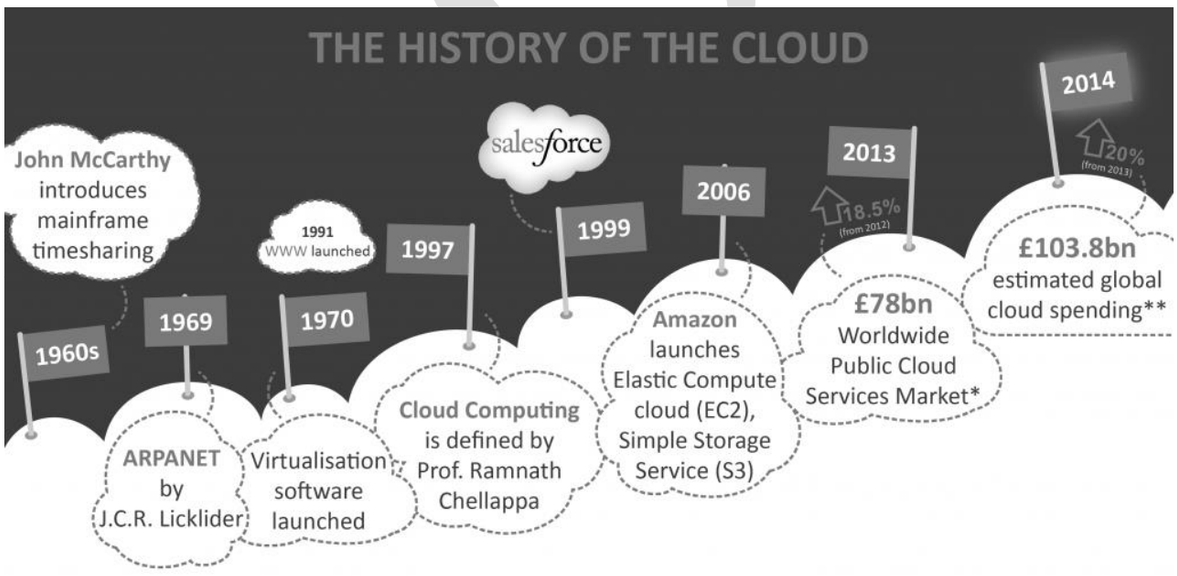


FIGURE 1. 9 History of Cloud Computing [*Gartner, **Constellation Research]

EARLY 1960S
Computer scientist John McCarthy has a time-sharing concept that allows the organization to use an expensive mainframe at the same time. This machine is described as a major contribution to Internet development, and as a leader in cloud computing.

IN 1969
J.C.R. Licklider, responsible for the creation of the Advanced Research Projects Agency (ARPANET), proposed the idea of an "Intergalactic Computer Network" or "Galactic Network" (a computer networking term similar to today's Internet). His vision was to connect everyone around the world and access programs and data from anywhere.

IN 1970
Usage of tools such as VMware for virtualization. More than one operating system can be run in a separate environment simultaneously. In a different operating system it was possible to

operate a completely different computer (virtual machine).

IN 1997
Prof Ramnath Chellappa in Dallas in 1997 seems to be the first known definition of "cloud computing," "a paradigm in which computing boundaries are defined solely on economic rather than technical limits alone."

IN 1999
Salesforce.com was launched in 1999 as the pioneer of delivering client applications through its simple website. The services firm has been able to provide applications via the Internet for both the specialist and mainstream software companies.

IN 2003
This first public release of Xen ,is a software system that enables multiple virtual guest operating systems to be run simultaneous on a single machine, which also known as the Virtual Machine Monitor (VMM) as a hypervisor.

IN 2006
The Amazon cloud service was launched in 2006. First, its Elastic Compute Cloud (EC2) allowed people to use their own cloud applications and to access computers. Simple Storage Service (S3) was then released. This incorporated the user-as-you-go model and has become the standard procedure for both users and the industry as a whole.

IN 2013
A total of £ 78 billion in the world 's market for public cloud services was increased by 18.5% in 2012, with IaaS as one of the fastest growing services on the market.

IN 2014
Global business spending for cloud-related technology and services is estimated to be £ 103.8 billion in 2014, up 20% from 2013 (Constellation Research).

Figure gives an analysis of the development of cloud computing distributed technologies. When we track the historic developments, we review briefly five key technologies that have played a significant role in cloud computing. They are distributed systems, virtualization, Web 2.0, service orientation and utility computing.

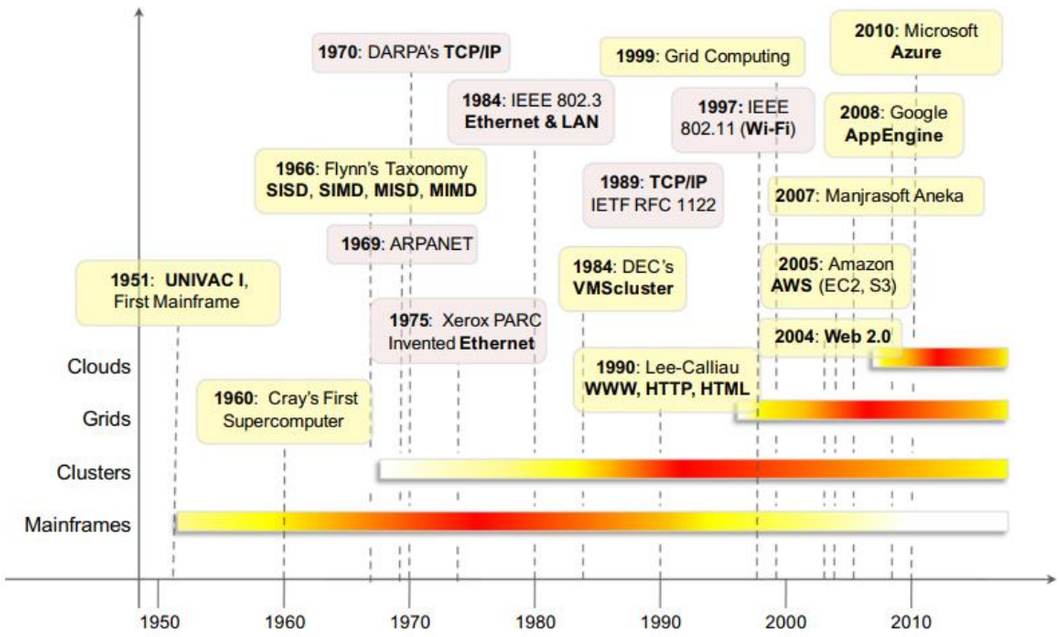


FIGURE 1.10: The evolution of distributed computing technologies, 1950s- 2010s.

Reference from "Mastering Cloud Computing Foundations and Applications Programming" by Rajkumar Buyya)

Distributed computing is a computer concept that refers most of the time to multiple computer systems that work on a single problem. A single problem in distributed computing is broken

down into many parts, and different computers solve each part. While the computers are interconnected, they can communicate to each other to resolve the problem. The computer functions as a single entity if done properly.

The ultimate goal of distributed computing is to improve the overall performance through cost-effective, transparent and secure connections between users and IT resources. It also ensures defect tolerance and provides access to resources in the event of failure of one component.

There really is nothing special about distributing resources in a computer network. This began with the use of mainframe terminals, then moved to minicomputers and is now possible in personal computers and client server architecture with several tiers.

A distributed computer architecture consists of a number of very lightweight client machines installed with one or several dedicated servers for computer management. Client agents normally recognize when the machine is idle, so that the management server is notified that the machine is not in use or that it is available. The agent then asks for a package. When this application package is delivered from the management server to the client, when it has free CPU cycles, the software runs the application software and returns the results to the management server. When the user returns, the management server will return the resources used to perform a number of tasks in the absence of the user.
Distributed systems show heterogeneity, openness, scalability, transparency, concurrency, continuous availability and independent failures. These characterize clouds to some extent, especially with regard to scalability, concurrency and continuous availability.
Cloud computing has contributed to three major milestones: mainframe, cluster computing and grid computing.

Mainframes: A mainframe is a powerful computer which often serves as the main data repository for an IT infrastructure of an organization. It is connected with users via less powerful devices like workstations or terminals. It is easier to manage, update and protect the integrity of data by centralizing data into a single mainframe repository. Mainframes are generally used for large-scale processes which require greater availability and safety than smaller machines. Mainframes computers or mainframes are primarily machines for essential purposes used by large organizations; bulk data processing, for example census, industry and consumer statistics, enterprise resource planning and transaction processing. During the late 1950s, mainframes only had a basic interactive interface, using punched cards, paper tape or magnetic tape for data transmission and programs. They worked in batch mode to support back office functions, like payroll and customer billing, mainly based on repetitive tape and merging operations followed by a line printing to continuous stationary pre-printed. Introducing digital user interfaces almost solely used to execute applications (e.g. airline booking) rather than to build the software. Typewriter and Teletype machines were standard network operators' control consoles in the early ' 70s, although largely replaced with keypads.

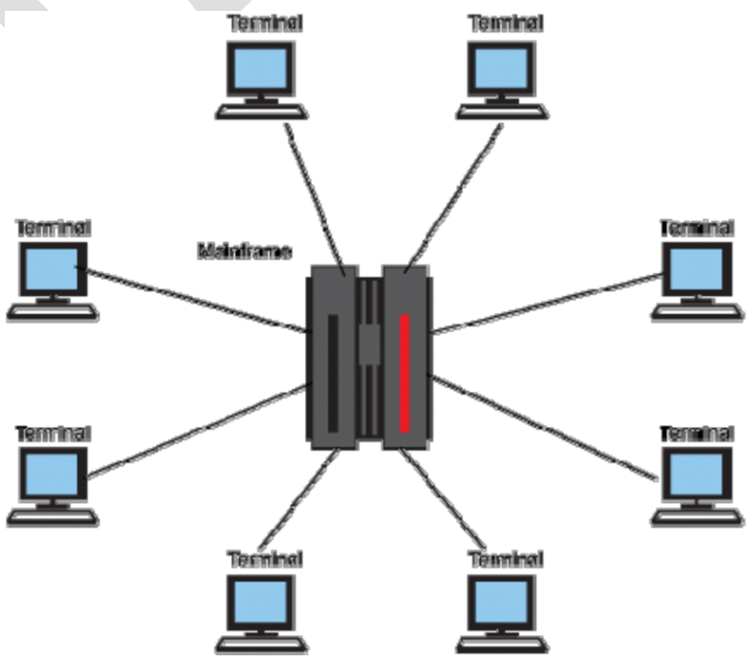


FIGURE 1.11 Mainframes

Cluster computing: The approach to computer clustering typically connects some computer

nodes (personal computer used as a server) ready for download via a fast local zone (LAN) network. Computing node activity coordinated by the software "clustering middleware," a software layer situated in front of nodes that enables the users to access the cluster as a whole by means of a Single system image concept. A cluster is a type of computer system that is parallel or distributed and which consists of a collection of interconnected independent computers, working together as a highly centralized computing tool that integrates software and networking with independent computers in a single system. Clusters are usually used to provide greater computational power than can be provided by a single computer for high availability for greater reliability or high performance computing. In comparison with other technology, the cluster technique is economical with respect to power and processing speeds, since it uses shelf hardware and software components in comparison with mainframe computers which use own hardware and software components custom built. Multiple computers in a cluster work together to deliver unified processing and faster processing. A Cluster can be upgraded to a higher specification, or extended by adding additional nodes as opposed to a mainframe computer. Redundant machines which take over the processing continuously minimize the single component failure. For mainframe applications, this kind of redundancy is absent.

PVM and MPI are the two methods most widely used in cluster communication.

PVM is the parallel virtual machine. The Oak Ridge National Laboratory was developed the PVM around 1989. It is installed directly on each node and provides a set of libraries that transform the node into a "parallel virtual machine." It offers a runtime environment for control of resources and tasks management , error reporting and message passing . C, C++ or Fortran may use PVM for user programs.

MPI is the message passing interface. In the 1990s PVM was created and replaced. Different commercially available systems of the time are the basis for MPI design. It typically uses TCP / IP and socket connections for implementation. The communication system currently used most widely allows for parallel scheduling in C, Fortran, Python etc.

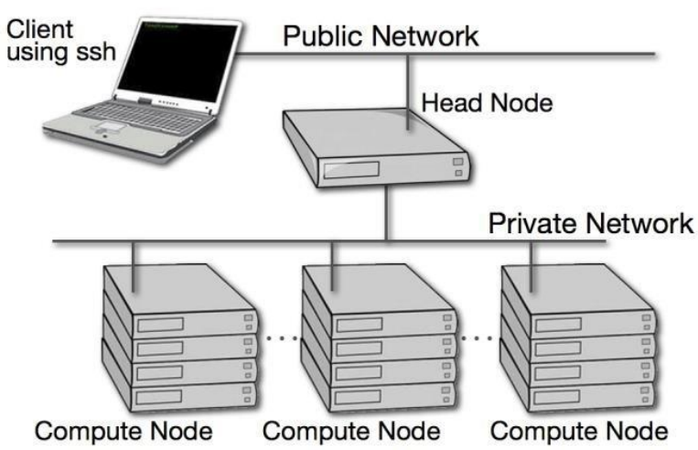


FIGURE 1.12 Cluster computing

Grid computing: It is a processor architecture that combines computer resources from different fields to achieve the main purpose. The network computers can work together on a work task in grid computing and therefore work as a super-computer. In general, a grid works on several tasks in a network, but can also work on specific applications. In general, a grid operates on different network tasks, but it can also operate on specific applications. It is intended to solve problems that are too large for a supercomputer and to retain the ability to handle several small problems. Computing grids have a multi-user network that meets discontinuous information processing requirements.

A grid is connected to a computer cluster, which runs on an operating system, on Linux or free software, using parallel nodes. The cluster can vary in size from small to multiple networks. The technology is used through several computing resources in a broad variety of applications, such as mathematical, research or educational tasks. It is often used in structural analysis as well as in web services such as ATM banking, back office infrastructure, and research in sciences or marketing. Grid computing consists of applications which are used in a parallel networking environment to solve computational problems. It connects every PC and combines information into a computational application.

Grids have a range of resources, whether through a network, or through open standards with clear guidelines to achieve common goals and objectives, based on different software and hardware structures, computer languages and framework.

Generally, grid operations are divided into two categories:
Data Grid: a system that handles large distributed sets of data used to control data and to share users. It builds virtual environments that facilitate scattered and organized research. A data grid example is Southern California's Earthquake Center, which uses a middle software framework to construct a digital library, a distributed filesystem and a continuous archive.

CPU Scavenging Grids: A cycle-scavenging system that moves projects as necessary from one PC to another. The search for extraterrestrial intelligence computing, including more than 3 million computers, represents a familiar CPU scavenging grid. The detection of radio signals in Searches for Extra Terrestrial Intelligence (SETI), is one of radio astronomy's most exciting applications. A radio astronomy dish was used by the first SETI team in the late 1950s. A few years later, the privately funded SETI Institute was established to perform more searches with several American radio telescopes. Today, in cooperation with the radio astronomy engineers and researchers of various observatories and universities, the SETi Institute creates its own collection of private funds again. SETI 's vast computing capacity has led to a unique grid computing concept which has now been expanded into many applications.

SETI@home is a scientific experiment using Internet connected computers for downloading and analyzing SETI program radio telescope data. A free software program utilizes the power of millions of computers and uses idle computer capacity to run in the background. Over two million years of combined processing time have taken place over 5.2 million participants.

Grid computing for biology, medicine, Earth sciences, physics, astronomy, chemistry and mathematics are being used. The Berkeley Open Infrastructure for Network Computing (BOINC) is free open source computer and desktop grid computing software. By using the BOINC platform, users can divide jobs between several grid computing projects and decide to only give them one percentage of CPU time.

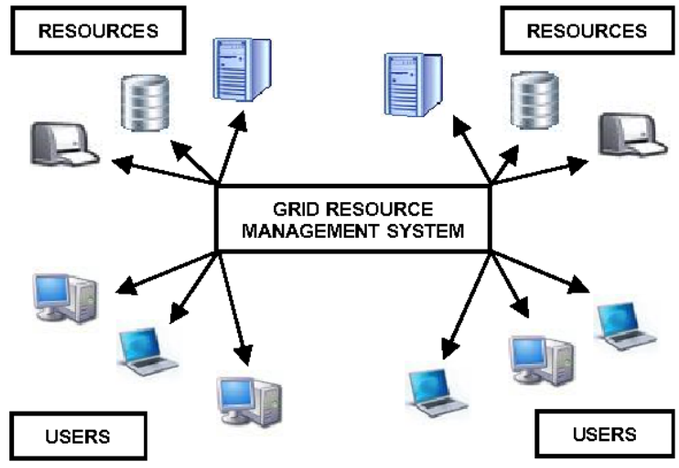


FIGURE 1.13 Grid computing Environment

1.3.2 Virtualization

Virtualization is a process that makes the use of physical computer hardware more effective and forms the basis for cloud computing. Virtualization uses software to create a layer of abstraction over computer hardware, enabling multiple virtual computers, usually referred to as VMs, to split the hardware elements from a single computer — processors, memory, storage and more. Every VM performs its own OS and acts like an autonomous computer given the fact that it runs on only a portion of the underlying computer hardware.

The virtualization therefore facilitates a much more effective use of physical computer hardware, thus allowing a larger return on the hardware investment of an organization.

Virtualization is today a common practice in IT architecture for companies. It is also the technology that drives the business of cloud computing. Virtualization allows cloud providers to service consumers with their own physical computing hardware and allows cloud users to purchase only the computer resources they need when they need it and scale them cost-effectively as their workloads increase.

Virtualization involves the creation of something's virtual platform, including virtual computer hardware, virtual storage devices and virtual computer networks.

Software called hypervisor is used for hardware virtualization. With the help of a virtual machine hypervisor, software is incorporated into the server hardware component. The role of hypervisor is to control the physical hardware that is shared between the client and the provider. Hardware virtualization can be done using the Virtual Machine Monitor (VVM) to remove physical hardware. There are several extensions to the processes which help to speed up virtualization activities and increase hypervisor performance. When this virtualization is done for the server platform, it is called server socialization.
Hypervisor creates an abstract layer from the software to the hardware in use. After a hypervisor is installed, virtual representations such as virtual processors take place. After installation, we cannot use physical processors. There are several popular hypervisors including ESXi-based VMware vSphere and Hyper-V.

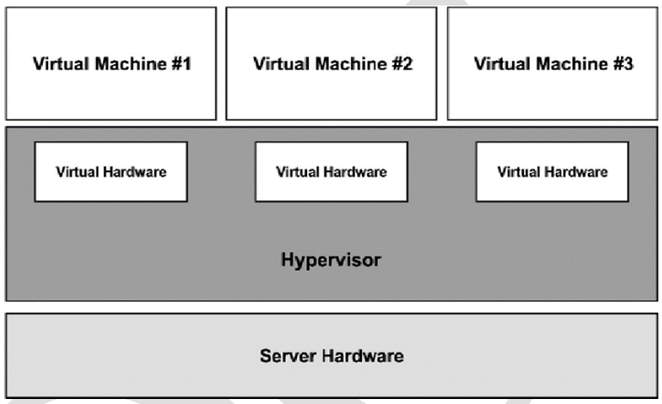


FIGURE 1.14 Hardware Virtualization

Instances in virtual machines are typically represented by one or more data, which can be easily transported in physical structures. In addition, they are also autonomous since they do not have other dependencies for their use other than the virtual machine manager.

A Process virtual machine, sometimes known as an application virtual machine, runs inside a host OS as a common application, supporting a single process. It is created at the beginning and at the end of the process. Its aim is to provide a platform-independent programming environment which abstracts the information about the hardware or operating system underlying the program and allows it to run on any platform in the same way. For example, Linux wine software helps you run Windows.

A high level abstraction of a VM process is the high level programming language (compared with the low-level ISA abstraction of the VM system). Process VMs are implemented by means of an interpreter; just-in-time compilation achieves performance comparable to compiled programming languages.

The Java programming language introduced with the Java virtual machine has become popular with this form of VM. The .NET System, which runs on a VM called the Common Language Runtime, is another example.

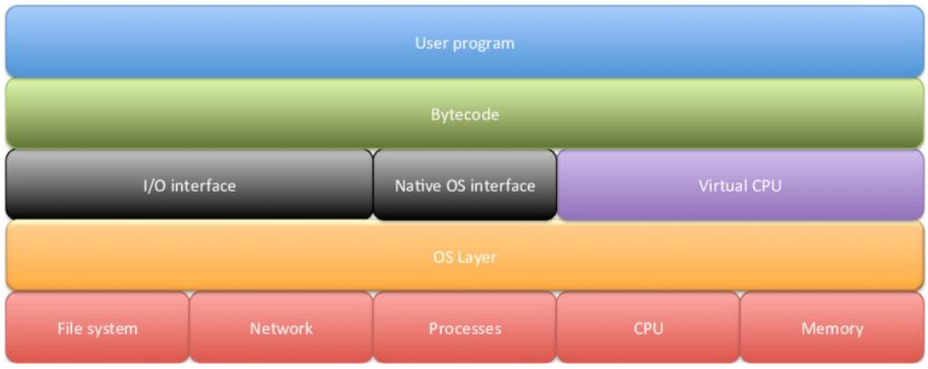


FIGURE 1.15 process virtual machine design

Reference from "Mastering Cloud Computing Foundations and Applications Programming" by Rajkumar Buyya)

Web 2.0

"Websites which emphasize user-generated content, user-friendliness, participatory culture, and interoperability for end users" or participatory, or participative / activist and social websites. Web 2.0 is a new concept that was first used in common usage in 1999 about 20 years ago. It was first coined by Darcy DiNucci and later popularized during a conference held in 2004 by Tim O'Reilly and Dale Doughtery. It is necessary to remember that Web 2.0 frameworks deal only with website design and use without placing the designers with technical requirements.

Web 2.0 is the term used to represent a range of websites and applications that permit anyone to create or share information or material created online. One key feature of the technology is the ability to people to create, share and communicate. Web 2.0 is different from other kinds of sites because it does not require the participation of any Web design or publishing skills and makes the creation, publication or communication of work in the world easy for people. The design allows it to be simple and popular for a small community or a much wider audience to share knowledge. The University will use these tools to communicate with students, staff and the university community in general. It can also be a good way for students and colleagues to communicate and interact.

It represents the evolution of the World Wide Web; the web apps, which enable interactive data sharing, user-centered design and worldwide collaboration. Web 2.0 is a collective concept of Web-based technologies that include blogging and wikis, online networking platforms, podcasting, social networks, social bookmaking websites, Really Simple Syndication (RSS) feeds. The main concept behind Web 2.0 is to enhance Web applications' connectivity and enable users to easily and efficiently access the Web. Cloud computing services are essentially Web applications that provide computing services on the Internet on demand. As a consequence, Cloud Computing uses a Web 2.0 methodology, Cloud Computing is considered to provide a main Web 2.0 infrastructure; it facilitates and is improved by the Web 2.0 Framework Beneath Web 2.0 is a set of web technologies. Recently appeared or shifted to a new production stage RIAs (Rich Internet Applications).One of them Web's most prominent technology and quasi-standard AJAX (Asynchronous JavaScript and XML). Other technologies like RSS (Really Simple Syndication), Widgets (plug-in modular components) and Web services (e.g. SOAP, REST).

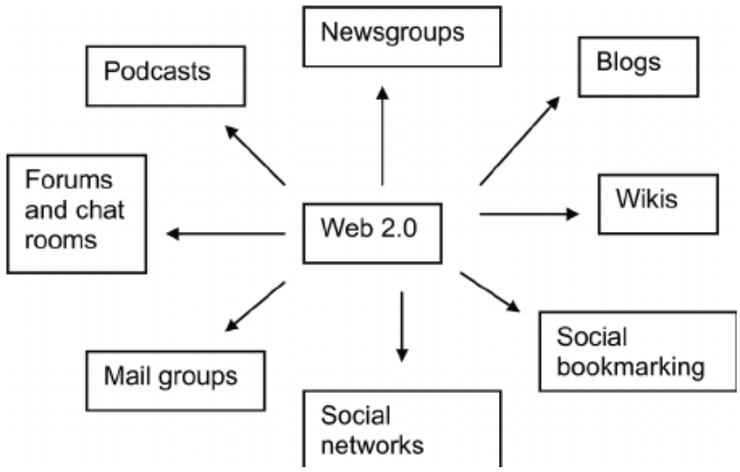


FIGURE 1.15 Components of the social web (Web 2.0)

1.3.3 Service-oriented computing

The computing paradigm that uses services as a fundamental component in the creation of applications / solutions is service oriented computing (SOC). Services are computer platform-specific self-description components that enable the easy and cost-effective composition of distributed applications. Services perform functions, from simple requests to complex business processes. Services permit organisations, using common XML languages and protocols, to display their core skills programming over the Internet or intra-network, and to execute it via an open-standard self-description interface.

Because services provide uniform and ubiquitous distributors of information for a wide variety of computing devices (e.g. handheld computers, PDAs, cell phones or equipment) as well as software platforms (e.g. UNIX and Windows), they are the next major step in distributed computing technology. Services are provided by service providers – organizations that provide the implementation of the service, provide their descriptions of service and related technical and business support. Since different services can be available

Companies and Internet communications provide a centralized networking network for the integration and collaboration intra- and cross-company application. Service customers can be other companies 'or clients' applications, whether they are external applications, processes or clients / users. These can include external applications.

Consequently, to satisfy these requirements services should be:

- **Technology neutral**: they must be invisible through standardized lowest common denominator technologies that are available to almost all IT environments. This implies that the invocation mechanisms (protocols, descriptions and discovery mechanisms) should comply with widely accepted standards.
- **Loosely coupled**: no customer or service side needs knowledge or any internal structures or conventions (context).
- **Transparency of support locations**: Services should have their definitions and location information saved in a repository such as UDDI and accessible to a range of customers which can locate services and invoke them regardless of their location.

  Web-service interactions take place with the use of Web Service Description Language (WSDL) as the common (XML) standard when calling Simple Object Access Protocol (SOAP) containing XML data, and the web-service descriptions. WSDL is used for the publishing of web services, for port types (the conceptual description of the procedure and interchange of messages), and for binding ports and addresses (the tangible concept of which packaging and transport protocols, for instance SOAPs, are used to interlink two conversational end-points). The UDDI Standard is a directory service that contains publications of services and enables customers to find and learn about candidate services.

  The software-as-a-service concept advocated by service-oriented computing (SOC) was pioneering and first appeared on the software model ASP (Application Service Provider). An Application Service Provider (ASP) is an entity which implements, hosts and handles the access of a third party. Packaged application and provides clients with software-based services and solutions from a central data center through a broad network. Subscription or

rental applications are delivered through networks. In essence, ASPs provided businesses with a way to outsource any or all parts of their IT needs.

The ASP maintains the responsibility for managing the application in its infrastructure, using the Internet as a connection between every customer and the key software application, through a centrally hosted Intent application. What this means for an organization is for the ASP to retention and guarantee the program and data are accessible whenever appropriate, including the related infrastructure and the customer data.

While the ASP model first introduced the software as a service definition, it was not able to provide full applications with customizable due to numerous inherent constraints such as its inability of designing extremely interactive applications.. The result has been the monolithic architectures and highly vulnerable integration of applications based on tight coupling principle in customer-specific architectures.
We are in the middle of yet another significant development today in the development of a software as a service architecture for asynchronous loosely linked interactions based on XML standards with the intention of making it easier for applications to access and communicate over the Internet. The SOC model enables the idea software-as-a-service to extend to use the provision of complicated business processes and transactions as a service, and allow applications to be created on the fly and services to be replicated across and by everyone. Many ASPs are pursuing more of a digital infrastructure and business models which are similar with those of cloud service providers to the relative advantages of internet technology.

Functional and non-functional attributes consist of the web services. Quality of service (QoS) is the so called unfunctional attributes. QoS is defined as a set of nonfunctional characteristics of entities used to move from a web service repository to consumers who rely on the ability of a web service to fulfill its specified or implied needs in an end-to - end way, according to the quality definition of ISO 8402. Examples of QoS features include performance, reliability, security, accessibility, usability, discovery, adaptively and composability. A SLA that identifies the minimum (or acceptable range) values for QoS attributes to be complied with on calling the service shall establish a QoS requirement between the clients and providers.

What is Service Oriented Architecture?
Service-oriented Architecture or SOA bring us all to understand it as a architecture which orients around services.. Services are discreet software components implemented using well-defined interface standards. Service is delivered to a directory or registry until it is created and validated to allow other developers to access the service. The registry also provides a repository that contains information on the published service, for example how to create the interface, what levels of service are required, how to retain authority, etc.

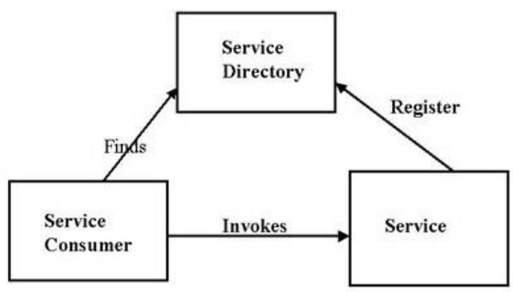


FIGURE 1.16 Service-oriented Architecture

SOA benefits
SOA services allow for agility of business. By integrating existing services, developers can create applications quickly.

The services are distinct entities and can be invoked without a platform or programming language knowledge at run-time.

The services follow a series of standards – Web Services Description Language (WSDL),

Representational State Transfer (REST), or the Simple Object Access Protocol(SOAP) – which facilitate their integration with both existing and new applications. The SOAP services are complemented by the following standards. SOAP.

Safety through Service Quality (QoS). Certain elements of QoS include authentication and authorisation, reliable and consistent messaging, permission policies, etc.
There is no interdependence of each other's service components.

SOA and cloud computing challenges
The network dependency of both of these technologies is one of the major challenges.
In addition, dependence on the cloud provider, contracts and service levels agreements is the challenges specific to cloud computing.

One of the challenges for SOA today are the requests to improve or change the service provided by SOA service providers.

Does Cloud Computing compete with SOA?

Some see cloud computing as a descendant of SOA. It would not be completely untrue, as the principles of service guidelines both apply to cloud computing and SOA. The following illustration shows how Cloud Computing Services overlap SOA-

Cloud Computing	Overlap	SOA via Web Services
• Software as a Service (SaaS) • Utility Computing • Terabytes on Demand • Data Distributed in a Cloud • Platform as a Service • Standards Evolving for Different Layers of the Stack	• Application Layer Components/Services • Network Dependence • Cloud/IP Wide Area Network (WAN)-supported Service Invocations • Leveraging Distributed Software Assets • Producer/Consumer Model	• System of Systems Integration Focus • Driving Consistency of Integration • Enterprise Application Integration (EAI) • Reasonably Mature Implementing Standards (REST,SOAP,WSDL, UDDI,etc.)

It is very important to realize that while cloud computing overlaps with SOA, they focus on various implementation projects. In order to exchange information between systems and a network of systems, SOA implementations are primarily used. Cloud computing, on the other hand, aims to leverage the network across the whole range of IT functions.

SOA is not suitable for cloud computing, actually they are additional activities. Providers need a very good service-oriented architecture to be able to provide cloud services effectively.

There are many common features of SOA and cloud computing, however, they are not and can coexist. In its requirements for delivery of digital services, SOA seems to have matured. Cloud Computing and its services are new as are numerous vendors such as public, community, hybrid and private clouds, with their offerings. They are also growing.

1.3.4 Utility-oriented computing

The concept Utility Computing pertains to utilities and business models that provide its customers with a service provider, and charges you for consumption. The computing power, storage or applications are examples of such IT services. In this scenario the customer will be the single divisions of the company as a service provider at a data center of the company.

The concept utility applies to utility services offered by a utilities provider, such as electricity, telephone, water and gas. Related to electricity or telephone, where the consumer receives the utility computing, computing power is measured and paid on the basis of a shared computer network.

The concept utility applies to utility services offered by a utilities provider, such as electricity, telephone, water and gas. Related to electricity or telephone, where the consumer receives the utility computing, computing power is measured and paid on the basis of a shared computer network.

Utility computing is very analogous to virtualization so that the total volume of web storage and the computing capacity available to customers is much greater than that of a single computer. To make this type of web server possible, several network backend servers are often used. The dedicated webservers can be used in explicitly built and leased cluster types for end users. The distributed computing is the approach used for a single 'calculation' on multiple web servers.

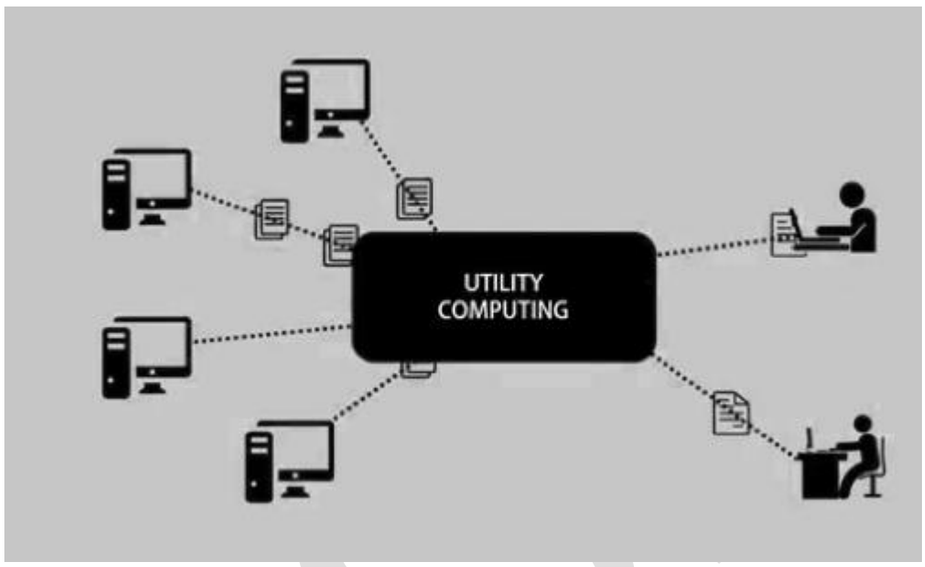


FIGURE 1.17 Cloud Computing Technology – Utility Computing

Properties of utility computing

Even though meanings of utility computing are various, they usually contain the following five characteristics.

Scalability

The utility computing shall ensure that adequate IT resources are available under all situations. Improved service demand does not suffer from its quality (e.g. response time).

Price of demand

Until now, companies must purchase their own computing power such as hardware and software. It is necessary to pay for this IT infrastructure beforehand, irrespective of its use in the future. For instance, technology providers to reach this link depends on how many CPUs the client has enabled during leasing rate for their servers. If the computer capacity to assert the individual sections actually can be measured in a company, the IT costs may be primarily attributable to each individual unit at internal cost. Additional forms of connection are possible with the use of IT costs.

Standardized Utility Computing Services

A collection of standardized services is accessible from the utility computing service provider. These agreements may differ in the level of service (Quality Agreement and IT Price). The consumer does not have any impact on the infrastructure, such as the server platform

Utility Computing and Virtualization

Virtualization technologies can be used to share web and other resources in the common pool of machines. Instead of the physical resources available, this divides the network into logical resources. No predetermined servers or storage of any other than a free server or pool memory are assigned to an application.

Automation

Repeated management activities may be automated, such as setting up new servers or downloading updates. Furthermore, the resource allocation to the services and IT service management to be optimized must be considered, along with service standard agreements and IT resource operating costs.

Advantages of Utility Computing

Utility computing lowers IT costs, despite the flexibility of existing resources. In fact, expenses are clear and can be allocated directly to the different departments of a organization. Fewer people are required for operational activities in the IT departments.

The companies are more flexible because their IT resources are adapted to fluctuating demand more quickly and easily. All in all, the entire IT system is simpler to handle, because no longer apps can take advantage of a particular IT infrastructure for any program.

1.4 Building cloud computing environments

Cloud Computing Environment Application development occurs through platforms & framework applications that provide various types of services, from the bare metal infrastructure to custom applications that serve certain purposes.

1.4.1 Application development

A powerful computing model that enables users to use application on demand is provided by cloud computing. One of the most advantageous classes of applications in this feature are Web applications. Their performance is mostly influenced by broad range of applications using various cloud services can generate workloads for specific user demands. Several factors have facilitated the rapid diffusion of Web 2.0. First, Web 2.0 builds on a variety of technological developments and advancements that allow users to easily create rich and complex applications, including enterprise applications by leveraging the Internet now as the main utility and user interaction platform. Such applications are characterized by significant complex processes caused by user interactions and by interaction between multiple steps behind the Web front. This is the application are most sensitive to improper infrastructure and service deployment sizing or work load variability.

The resource-intensive applications represent another class of applications that could potentially benefit greatly by using cloud computing. These applications may be computational or data-intensive. Significant resources are in both cases
Required in reasonable time to complete the execution. It should be noted that such huge quantities of resources are not constantly or for a long time needed. Scientific applications, for example, may require huge computational capacity to conduct large-scale testing once in a while so that infrastructure supports them cannot be purchased. Cloud computing is the solution in this case. Resource-intensive applications are not collaborative, and are characterized mainly by batch processing.

1.4.2 Infrastructure and system development

1.4.3 Computing platforms and technologies

Cloud application development involves leveraging platforms and frameworks which offer different services, from the bare metal infrastructure to personalized applications that serve specific purposes.

1.4.3.1 Amazon web services (AWS)

Amazon Web Services (AWS) is a cloud computing platform with functionalities such as database storage, delivery of content, and secure IT infrastructure for companies, among others. It is known for its on-demand services namely Elastic Compute Cloud (EC2) and Simple Storage Service (S3). Amazon EC2 and Amazon S3 are essential tools to

understand if you want to make the most of AWS cloud.

Amazon EC2 is a software for running cloud servers that is short for Elastic Cloud compute. Amazon launched EC2 in 2006, as it allowed companies to rapidly and easily spin servers into the cloud, instead of having to buy, set up, and manage their own servers on the premises.

While Amazon EC2 server instances can also have bare-metal EC2 instances, most Amazon EC2 server instances are virtual machines housed on Amazon's infrastructure. The server is operated by the cloud provider and you don't need to set up or maintain the hardware.) A vast number of EC2 instances are available for different prices; generally speaking the more computing capacity you use, the higher the EC2 instance you need. (Bare metal Cloud Instances permit you to host a working load on a physical computer, rather than a virtual machine. In certain Amazon EC2 examples, different types of applications such as the parallel processing of big data workload GPUs are optimized for use.

EC2 offers functionality such as auto-scaling, which automates the process of increasing or decreasing compute resources available for a given workload, not just to make the deployment of a server simpler and quicker. Auto-scaling thus helps to optimize costs and efficiency, especially in working conditions with significant variations in volume.

Amazon S3 is a storage service operating on the AWS cloud (as its full name, Simple Storage Service). It enables users to store virtually every form of data in the cloud and access the storage over a web interface, AWS Command Line Interface, or AWS API. You need to build what Amazon called a 'bucket' which is a specific object that you use to store and retrieve data for the purpose of using S3. If you like, you can set up many buckets.

Amazon S3 is an object storage system which works especially well for massive, uneven or highly dynamic data storage.

1.4.3.2 Google AppEngine

The Google AppEngine (GAE) is a cloud computing service (belonging to the platform as a service (PaaS) category) to create and host web-based applications within Google's data centers. GAE web applications are sandboxed and run across many redundancy servers to allow resources to be scaled up according to currently-existing traffic requirements. App Engine assigns additional resources to servers to handle increased load.

Google App Engine is a Google platform for developers and businesses to create and run apps using advanced Google infrastructure. These apps must be written in one of the few languages supported, namely Java, Python, PHP and Go. This also requires the use of Google query language and Google Big Table is the database used. The applications must comply with these standards, so that applications must either be developed in keeping with GAE or modified to comply.

GAE is a platform for running and hosting Web apps, whether on mobile devices and on the Web. Without this all-in function, developers should be responsible for creating their own servers, database software and APIs that make everyone work together correctly. GAE takes away the developers' pressure so that they can concentrate on the app's front end and features to enhance user experience.

1.4.3.3 Microsoft Azure

Microsoft Azure is a platform as a service (PaaS) to develop and manage applications for using their Microsoft products and in their data centers. This is a complete suite of cloud products that allow users to develop business-class applications without developing their own infrastructure.

Three cloud-centric products are available on the Azure Cloud platform: the Windows Azure, SQL Azure & Azure App Fabric controller. This involve the infrastructure hosting facility for the application.

In the Azure, the Cloud service role is a set of virtual platforms that work together to

accomplish basic tasks, which is managed, load-balanced and Platform-as-a-Service. Cloud Service Roles are controlled by Azure fabric controller and provide the perfect combination of scalability, control, and customization.

Web Role is the role of an Azure Cloud service which is configured and adapted to operate web applications developed on the Internet Information (IIS) programming languages and technologies, such as ASP.NET, PHP, Windows Communication Foundation and Fast CGI.

Web Role is the role of an Azure Cloud service which is configured and adapted to operate web applications developed on the Internet Information (IIS) programming languages and technologies, such as ASP.NET, PHP, Windows Communication Foundation and Fast CGI.

Worker role is any role for Azure that works on applications and services that do not usually require IIS. IIS is not enabled default in Worker Roles. They are mainly utilized to support web-based background processes and to do tasks such as compressing uploaded images automatically, run scripts, get new messages out of queue and process and more, when something changes the database.

VM Role: The VM role is a type of Azure Platform role that supports the automated management of already installed service packages, fixes, updates and applications for Windows Azure.

The principal difference is that:

A Web Role deploys and hosts the application automatically via IIS A Worker Role does not use IIS and runs the program independently The two can be handled similarly and can be run on the same Azure instances if they are deployed and supplied via the Azure Service Platform.

For certain cases, instances of Web Role and Worker Roles work together and are also used concurrently by an application. For example, a web role example can accept applications from users, and then pass them to a database worker role example.

1.4.3.4 Hadoop

Apache Hadoop is an open source software framework for storage and large-scale processing of data sets of commodity hardware clusters. Hadoop is a top-level Apache project created and operated by a global community of contributors and users. It is under the Apache License 2.0.

Two phases of MapReduce function, Map and Reduce. Map tasks are concerned with data splitting and mapping of the data, while Reduce tasks shuffle and reduce the data.
Hadoop can run MapReduce programs in a variety of languages like Java, Ruby, Python, and C++,. MapReduce program is parallel in nature and thus very useful for large-scale analyzes of data via multiple cluster machines.

The input to each phase is key-value pairs. In addition, every programmer needs to specify two functions: map function and reduce function.

1.4.3.5 Force.com and Salesforce.com

The fundamental concepts on cloud computing must be understood to understand the divergence between salesforce.com and force.com.

Salesforce is a company and salesforce.com is an application built on the basis of software as a service (SaaS) for customer relationships management (CRM). The force.com platform assists developers and business users in creating successful business applications.

Salesforce is a SaaS product that includes the Out of Box (OOB) features built into a CRM system for sales automation, marketing, service automation, etc. Some SaaS examples are Dropbox, Google Apps and GoToMeeting that refer to taking the software from your computer to the cloud.

Force.com is a PaaS (Platform-as-a-Service) product; it includes a framework that allows you to build applications. It contains a development environment. Force.com helps you to customize the user interface, functionality and business logic.

Simply put, Salesforce.com functionality saves contacts, text messages, calls and other standard functions within the iPhone application. In force.com, the applications are constructed and operated. Salesforce.com runs on force.com, like the iPhone dialer works on the iPhone OS.

1.4.3.6 Manjrasoft Aneka

MANJRASOFT Pvt. Ltd. is one of organization that works on cloud computing technology by developing software compatible with distributed networks across multiple servers.

- Create scalable, customizable building blocks essential to cloud computing platforms.
- Build software to accelerate applications that is designed for networked multi-core computers.
- Provide quality of service (QoS) and service level Agreement (SLA)-solutions based on the service level agreement (SLA) which allow the scheduling, dispatching, pricing of applications and accounting services, Business and/or public computing network environments.
- Development of applications by enabling the rapid generation of legacy and new applications using innovative parallel and distributed models of programming.
- Ability of organizations to use computing resources .Business to speed up "compute" or "data" execution-intensive applications

1.5 Summary

In this chapter, we explored the goal and advantages and challenges associated with the cloud computing. As a consequence of the development and integration of many of its supportive models and technologies, especially distributed computing, the cloud computing technologies Web 2.0, virtualization, Services orientated and Utility Computing. We are examining various definitions, meanings and implementations of the concept. Only the dynamic provision of IT services (whether it is virtual infrastructure, runtime environments or application services) and the implementation of a utility-based cost model to value such services is the component shared by all different views of cloud computing. This architecture is applied throughout the entire computing stack and allows the dynamic provision of IT and runtime resources in the context of cloud-hosted platforms to create scalable applications and their services. The cloud computing reference method is represented by this concept. This model defines three important components of Cloud computing's industry and Services there are offering: Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-service (SaaS). These components explicitly map the wide categories of the various types of cloud computing services.

1.6 Review questions

1. What is cloud computing’s innovative characteristic?
2. What are the technologies that are supported by cloud computing?
3. Provide a brief characterization of a distributed system.
4. Define cloud computing and Identify the main features of cloud computing.
5. What are the most important distributed technologies that have contributed to cloud computing?
6. What is a virtualization?
7. Explain the major revolution introduced by web 2.0
8. Give examples of applications for Web 2.0.
9. Describe the main features of the service orientation.
10. Briefly summarize the Cloud Computing Reference Model.
11. What is the major advantage of cloud computing?
12. Explain the different types of models in the cloud computing
13. Explain the three cloud services in cloud computing.
14. What is Web services? Explain the different types of web services.

1.7 Reference for further reading

1. Mastering Cloud Computing Foundations and Applications Programming Rajkumar Buyya ,Christian Vecchiola,S. Thamarai Selvi MK publications ISBN: 978-0-12-411454-8
2. Cloud Computing Concepts, Technology & Architecture Thomas Erl, Zaigham Mahmood, and Ricardo Puttini , The Prentice Hall Service Technology Series ISBN-10 : 9780133387520 ISBN-13 : 978-0133387520
3. Distributed and Cloud Computing: From Parallel Processing to the Internet of Things 1st Edition by Kai Hwang Jack Dongarra Geoffrey Fox ISBN-10 : 9789381269237 ISBN-13 : 978-9381269237
4. https://www.geeksforgeeks.org/cloud-computing/
5. https://en.wikipedia.org/wiki/Cloud_computing
6. https://aws.amazon.com/what-is-cloud-computing/
7. http://www.manjrasoft.com/aneka_architecture.html
8. https://en.wikipedia.org/wiki/Microsoft_Azure
9. https://en.wikipedia.org/wiki/Apache_Hadoop

Unit 1

Chapter 2

Unit Structure

2.0 Objective

The computing components (hardware, software, infrastructures) that allow the delivery of cloud computing services refer to a Cloud system or cloud computing technology.

Consumers can acquire new skills without investing in new hardware or software via the public cloud. Instead, they pay a subscription fee for their cloud provider or only pay for their resources. These IT assets are owned and managed through the Internet by the service providers.

This chapter presents the basic principles and models of parallel and distributed computing, which provide the foundation for building cloud computing systems and frameworks.

2.1 Eras of computing

The two most prominent computing era are sequential and parallel. In the past decade, the high performance computer searches for parallel machines have become important competitors of vector machines. Figure 2.1 provides a hundred-year overview of the development of the computing era. During these periods the four main computing elements are created like architectures, compilers, applications and problem-solving environments.

The computing era begins with the development of hardware, followed by software systems (especially in the area of compilers and operating systems), applications, and with a growing problem solving environment it enters its saturation level. Each computing element is subject to three stages: R&D, commercialization and commodity.

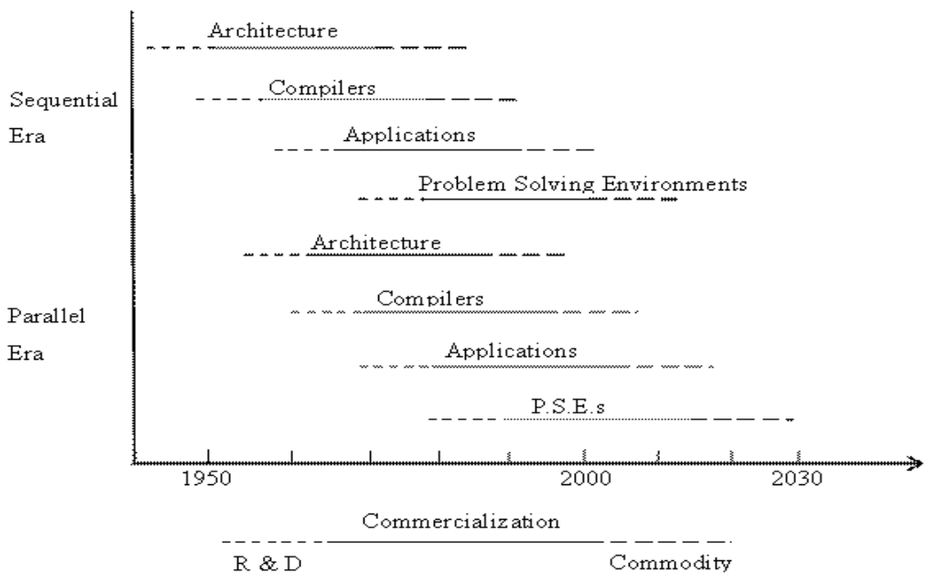


FIGURE 2.1 Eras of computing

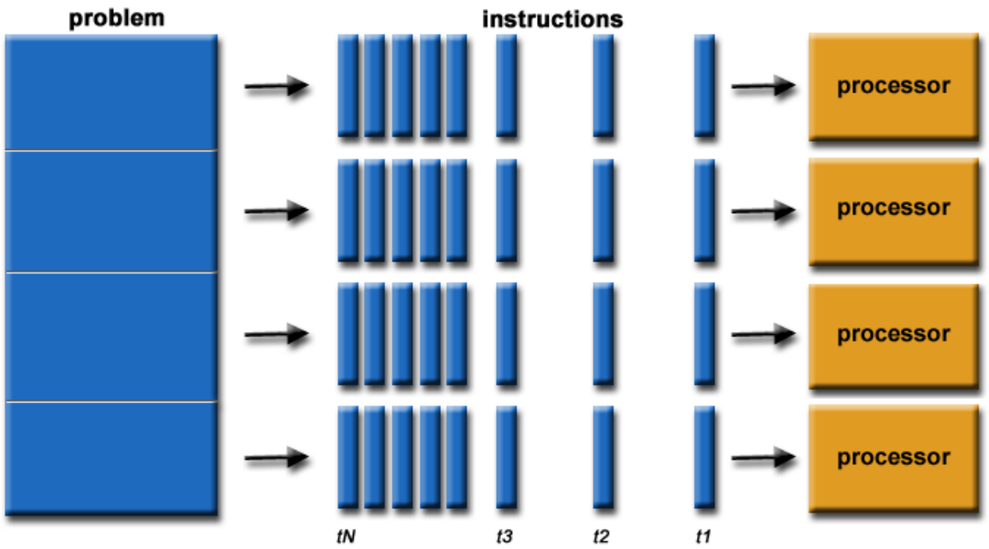
2.2 Parallel vs. distributed computing

Parallel computing and distributed computing terms, even though they are somewhat different things, are often used interchangeably. The term parallel means a tightly coupled system, whereas the distributed one refers to a wider system class, including the tightly coupled.

The concurrent use of several computer resources to solve a computational problem is parallel computing:

- A problem is divided into discrete pieces which can be solved simultaneously
- A number of instructions for each part are broken down further

- Instructions on various processors from each part run simultaneously
- An overall mechanism for control/coordination is used



For example:

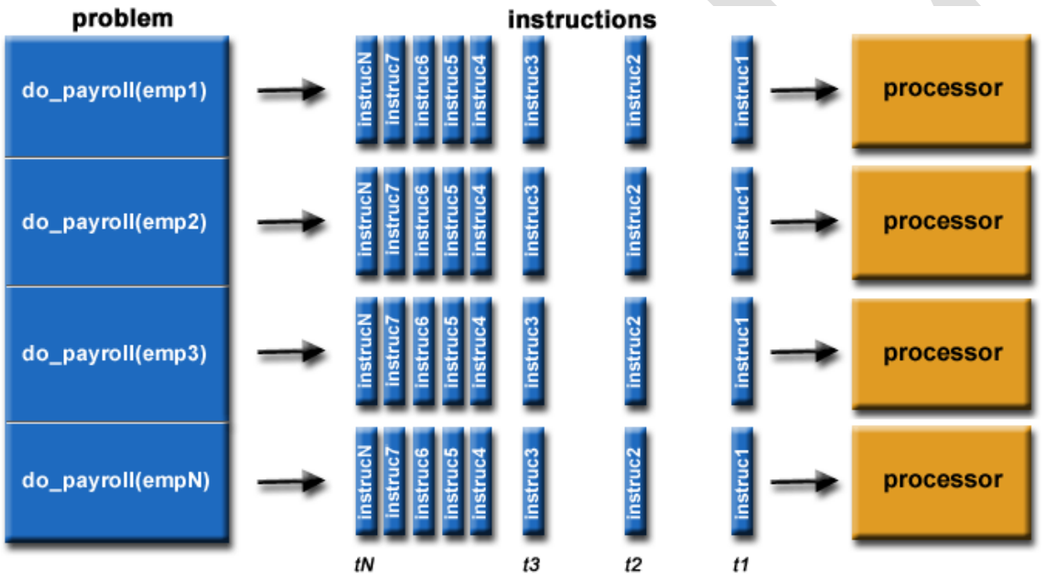


FIGURE 2.2 Sequential and Parallel Processing

The problem in the computation should be:

The problem in the computation should be:

- Be divided into discreet parts of work that can be solved at the same time;;
- At any given time, execute multiple program instructions;
- With many compute resources in less time than one compute resource, can be solved.

Typically, computation resources are:

- One computer with several processors / cores
- A random number of these computers connected through a network

Initially, only certain architectures were considered by parallel systems .It featured multiple processors with the same physical memory and a single computer. Over time, those limitations have been relaxed and parallel systems now include all architectures, whether physically present or based on the concept of shared memory, whether the library support, specific hardware and a very efficient network infrastructure are present physically or created. For example, a cluster of the nodes linked by InfiniBand can be considered a parallel system and configured with a distributing shared memory system

Computing distributed is computed by distributed independent computers communicating only over a network (Figure). Distributed computing systems are typically treated differently than parallel computing systems or shared memory systems, in which many computers share a common memory pool used to communicate processors to each other. Memory systems used multiple computers in order to solve a common problem, computing between the connected computers (nodes) and communicating between nodes through message-passage.

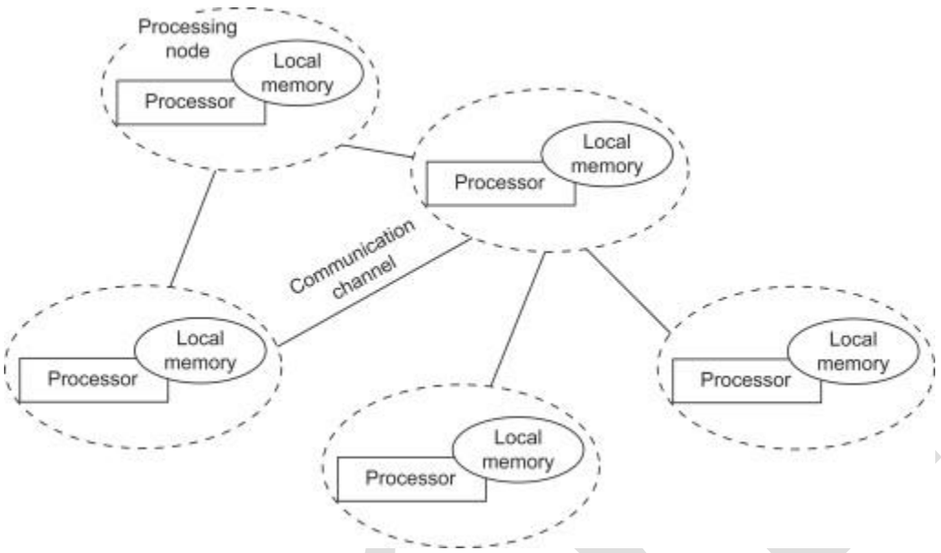


FIGURE 2.3. A distributed computing system.

Distributed computing is limited to programs in a geographically-limited area with components shared among computers. Broader definitions both include common tasks and program components. Distributed computing in the broadest sense means that something is shared between many systems, which can also happen in different locations.

Examples of distributed systems / applications of distributed computing:

- Intranets, Internet, WWW, email.
- Telecommunication networks: Telephone networks and Cellular networks.
- Network of branch office computers -Information system to handle automatic processing of orders,
- Real-time process control: Aircraft control systems,
- Electronic banking,
- Airline reservation systems,
- Sensor networks,
- Mobile and Pervasive Computing systems.

2.3 Elements of parallel computing

That is the exponential increase in computing power. In 1965, Intel's co-founder, Gordon Moore, noted that the number of transistors on a single-inch chip doubles per year, while the cost falls by about half. It's now 18 months, and it gets longer. Silicon reaches a performance limit in an increasing number of applications requiring increased speed, reduced latency and light detection. To address this constraint, the feasible solution is to connect several processors to solve "Great Challenge" problems in coordination with each other. The initial steps towards parallel computing lead to the growth. It includes technology, architecture and systems for multiple parallel activities. This section refers to its proper characterization, which includes the parallelism of the operation of multiple processors coordinating together within single computer.

2.3.1 What is parallel processing?

Parallel processing is a way to manage different parts of an overall task when comparing two or more processors. CPUs. Break up various parts of a task among several processors can reduce the time a program needs to run. Either machine with over one CPU or multi-core processors which are commonly found on computers today may perform parallel processing. In the parallel computing machine the concept known as divide and conquer .Divide and conquer is an elegant way to solve a problem. You split up problems in smaller problems of the same type may be resolved individually, and partial outcomes combined in a total solution. The approach is used to break the problem into smaller and smaller problems, until any problem is solved easily. Parallel programming is called multiprocessor system programming using the divide and conquer technique

Intensive computational problems and applications require additional processing power than it has ever been. Although the processor's speed is increasing, traditional sequential computers do not deliver the power to solve these problems. In parallel computers, an area in which many processors simultaneously take on problems, many of these problems are potentially addressed.

Several factors influence the development of parallel processing. The following are prominent among them are:

1. In many fields of science and engineering parallel computing was considered the "high end computing" to model problems that were difficult to solve: In the fields like
   - Atmosphere, Earth, Environment
   - Physics - applied, nuclear, particle, condensed matter, high pressure, fusion, photonics
   - Bioscience, Biotechnology, Genetics
   - Chemistry, Molecular Sciences
   - Geology, Seismology
   - Mechanical Engineering - from prosthetics to spacecraft
   - Electrical Engineering, Circuit Design, Microelectronics
   - Computer Science, Mathematics
   - Defense, Weapons

2. Sequential architectures are physically constrained by the speed of light and the laws of thermodynamics. The saturation point (no vertical growth) is reached by a speed at which sequential CPUs can operate. Therefore, an alternate way to achieve high computational speed is to connect several CPUs (the possibility for horizontal growth).
3. Pipeline hardware, superscale etc. improvements are not scalable and require sophisticated compiler technology. The task is difficult to develop this compiler technology
4. Another attempt to improve performance was vector processing by doing more than one task at a time. Capability to add (or subtract or multiply, or otherwise manipulate) two numerical arrays to devices has been introduced in this case. This was useful when data naturally appeared in vectors or matrices in certain engineering applications. Vector processing was not so valuable in applications with less well-formed data.
5. There is indeed extensive R&D work on development tools and environments and parallel processing technology is mature, and commercially exploitable.
6. Essential networking technology advancement paves the way for heterogeneous computing.

2.3.2 Hardware architectures for parallel processing

Parallel computers highlight the parallel processing of the operations somehow. All basic parallel processing and computing concepts have been specified in the previous unit. Parallel computers can be distinguished by data and instruction streams of computer organizations. They can also be classified on a computer structure, for example multiple processors with a separate memory or a global shared memory. In a program called grain

size, parallel levels of processing can also be defined based on the size of instructions. But computers in parallel can be classified according to different criteria

The following classification of parallel computers have been identified:
1) Classification based on the instruction and data streams
2) Classification based on the structure of computers
3) Classification based on how the memory is accessed
4) Classification based on grain size

Flynn's Classical Taxonomy

- Parallel computers are classified in different ways.
- Flynn Taxonomy has been one of the most widely used classifications used since 1966.
- Flynn's taxonomy defines the architecture of multi-processor computers according to how the two distinct aspects of instruction and data stream can be categorized. Each of these dimensions can only contain a **Single or multiple** state of one kind.
- The following matrix describes the four possible Flynn classifications**:**

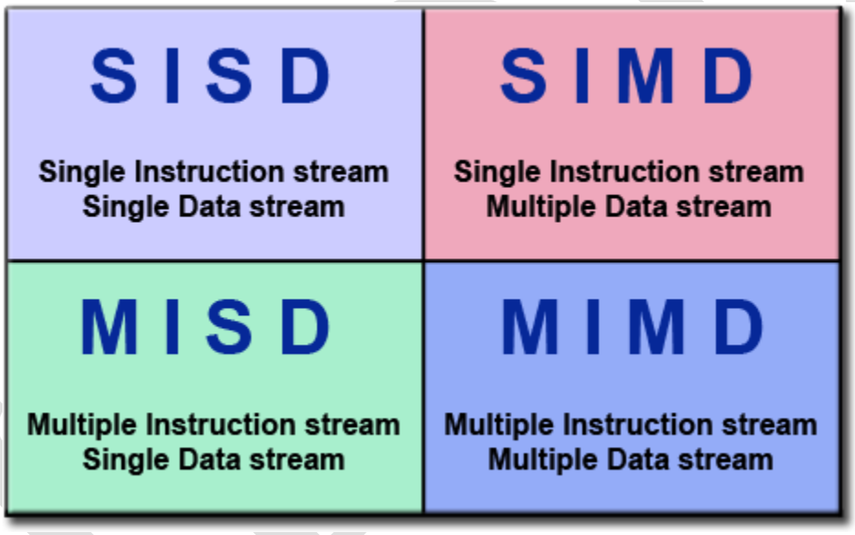


FIGURE 2.4 • Flynn Taxonomy

2.3.2.1 Single-instruction, single-data (SISD) systems

SISD computing system is a uniprocessor that can execute a single instruction on a single data stream. The SISD processes machine instructions sequentially, computers that adopt this model are commonly referred to as sequential computers. The SISD architecture is used in most conventional computers. All processing instructions and data should be stored in the primary memory. Depending on the rates at which the computer can transfer information internally, the speed is restricted in the processing element of the SISD model. The IBM PC, workstations are the prevalent representative SISD systems.

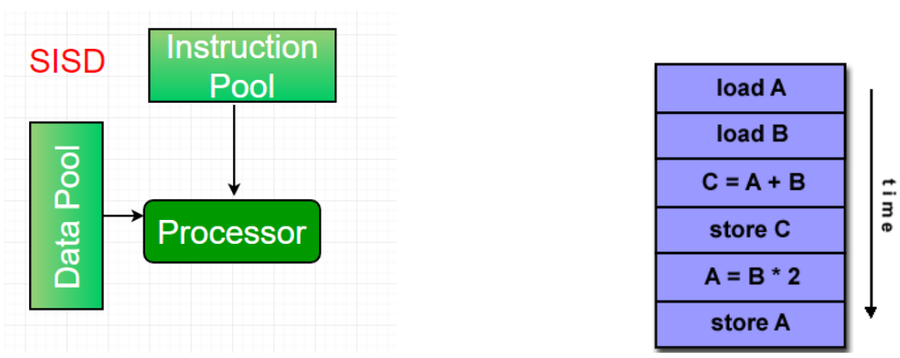


FIGURE 2.5: Single-instruction, single-data (SISD) architecture

2.3.2.2 Single-instruction, multiple-data (SIMD) systems

A SIMD system is a multi-processor system that can execute the same instruction on all CPUs but operates on many data streams. SIMD-based machines are ideal for scientific computing because they involve many vector and matrix operations. The data may be divided into multiple sets (N-sets for N PE systems) so that the information can be transferred to all the processing elements (PEs). Each PE can process the same data set .This is ideally suited for complex problems with a high degree of regularity like graphics / image processing. Most modern computers, particularly those with graphics processor units (GPUs) employ SIMD instructions and execution units. Dominant representative SIMD systems is Cray's vector processing machine.

Examples:

Processor Arrays: Thinking Machines CM-2, MasPar MP-1 & MP-2, ILLIAC IV

Vector Pipelines: IBM 9000, Cray X-MP, Y-MP & C90, Fujitsu VP, NEC SX-2, Hitachi S820, ETA10

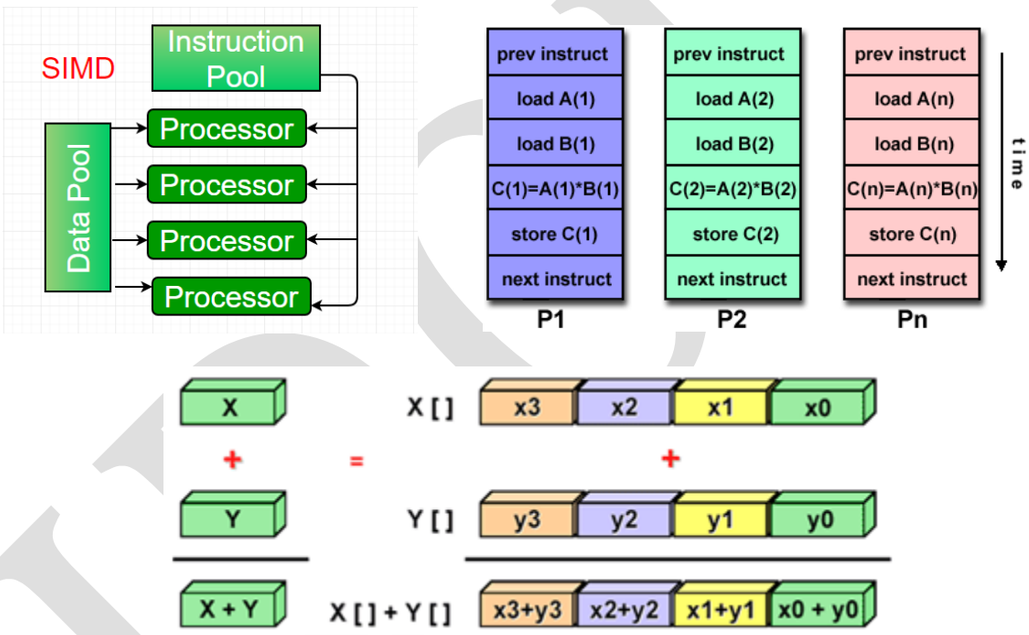


FIGURE 2.6 : Single-instruction, multiple-data (SIMD) architecture.

2.3.2.3 Multiple-instruction, single-data (MISD) systems

An MISD is a multiprocessor system that executes different instructions on different PEs, but they all operate in the same dataset.

Multiple instructions: every processing unit works independently on the data over separate streams of instructions.

Single Data: A single stream of data is fed into multiple processing units.

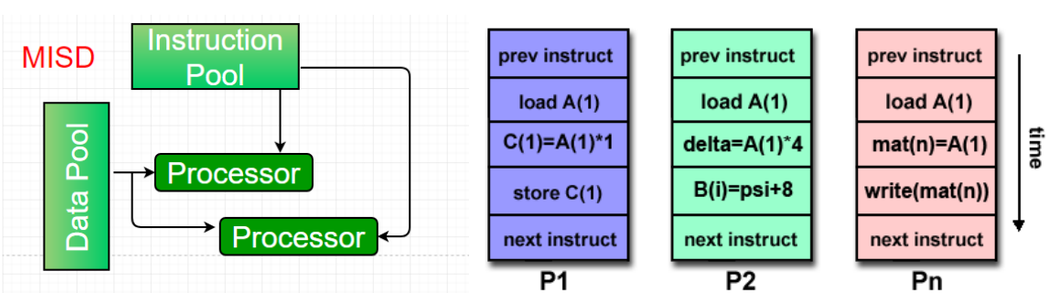


FIGURE 2.7 Multiple-instruction, single-data (MISD) architecture.

Example Z = sin(x)+cos(x)+tan(x)
On the same data set the system performs various operations. For most applications, machines designed using MISD are not useful, some are designed, but none of them are commercially available.

2.3.2.4 Multiple-instruction, multiple-data (MIMD) systems

MIMD is a multiprocessor system that can carry out multiple instructions on multiple sets of data. Every PE in a model with a MIMD has separate instructions and data streams, so any type of application can be used on machines built using this model. In comparison to SIMD and MISD, PEs can operate synchronous or asynchronous, deterministic or non-deterministic in MIMD computers. Currently, the most common type of parallel computer - most modern supercomputers fall into this category. Examples: most current supercomputers, networked parallel computer clusters and "grids", multi-processor SMP computers, multi-core PCs.

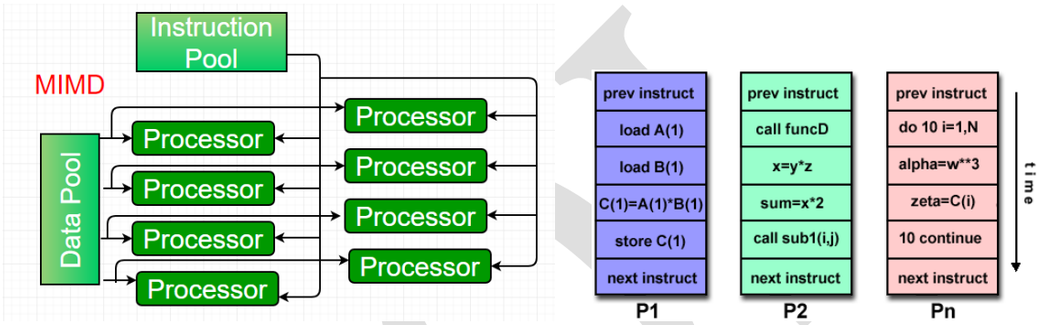


FIGURE 2.8 Multiple-instructions, multiple-data (MIMD) architecture.

MIMD machines are divided broadly into shared-memory MIMD and distributed-memory MIMD on the manner in which PEs are connected to the main memory.

Shared memory MIMD machines

All PEs are connected to a single global memory in the shared MIMD (tightly coupled multiprocessor systems) model and have all access to it. The communication between PEs within this model takes place by means of a shared memory, changes of the data stored by one PE in the global memory are visible to all other PEs. The dominant shared memory systems for Silicon Graphics and Sun / IBM (Symmetric Multi-Processing) are shared memory systems.

Distributed memory MIMD machines

All PEs have a local memory on distributed memory MIMD machines (loose multiprocessor systems). In this model, communication among PEs is carried out via the interconnection network (the inter-process communication channel or IPC). The network connecting PEs can be set up in tree, mesh or as needed.

The MIMD shared memory architecture is easier to design, but less tolerant to failure and more difficult to expand compared to the MIMD distributed memory model. Shared MIMD failures affect the entire system, but not the distributed model in which every PE can be easily isolated. In comparison, MIMD shared memory architectures are less likely to scale as the introduction of more PEs triggers memory conflict. This is not the case with distributed memory, in which each PE has its own memory. Thanks to realistic effects and consumer specifications, the distributed MIMD memory architecture is better than the others.

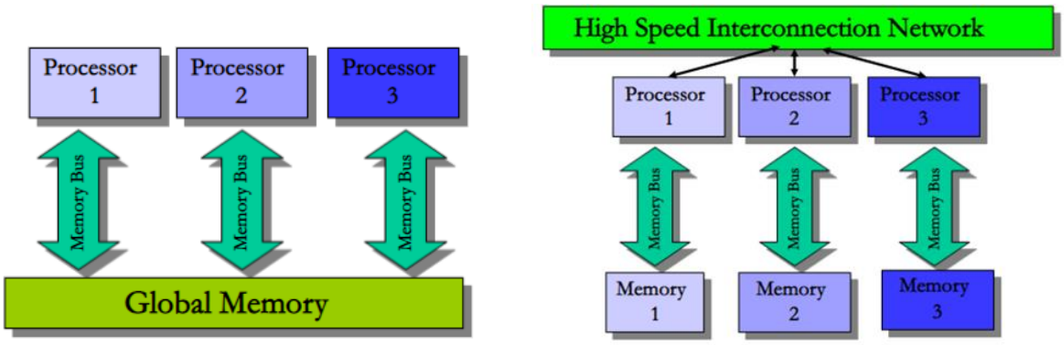


FIGURE 2.9 shared (left) and distributed (right) memory MIMD architecture.

2.3.3 Approaches to parallel programming

In general, a sequential program always runs the same sequence of instructions with the same input data and always generates the same results were as programs must be represented by splitting work into several parts running on different processors. The broken program is a parallel program.
Various methods are available for parallel programming. The most significant of these are:

- Data parallelism
- Process parallelism
- Farmer-and-worker model

Each three of these models can be used for task-level parallelism In the case of data parallelism, Divide and conquer is a multi-branched recursion-based design algorithm. A divide-and - conquer algorithm works by breaking a data into two or more similar or related data repetitively and the same instructions are used to process each data set for different PEs. This is a very useful approach for machine processing based on the SIMD model. With process parallelism, there are many (but separate) operations in a single activity that could be done on several processors. In farmer and-worker model, the main (master) computation causes many sub problems which slave fires off to be executing. The only communication between the master and slave computations is to start the master computation for slaves, and return the result of the slave computation to master.

2.3.4 Levels of parallelism

Bit-level Parallelism: In this parallelism, it's focused on the doubling of the word size of the processor. Increased parallelism in bit levels means that arithmetical operations for large numbers are executed more quickly. An 8-bit processor, for example, takes 2 cycles to perform a 16-bit addition whereas a 16-bit is a single cycle. With the advent of 64-bit processors this degree of parallelism seems to be over.

Instruction-level parallelism (ILP): This form of parallelism aims to leverage the possible overlap in a computer program between instructions. On each hardware of the processor, most ILP types are implemented and applied:

Instruction Pipelining: Execute various stages in the same cycle of various independent instructions and use all idle resources.

Task Parallelism: Task parallelism involves breaking down a task into subtasks and then assigning each of the subtasks for execution. Subtasks are carried out concurrently by the processors.

Out-of-order execution: Instructions without breaching data dependencies may be executed if even though previous instructions are still executed, a unit is available.

2.3.5 Laws of caution

Already that we have implemented certain basic elements of parallel computing in architecture and models, we can take into account some of the knowledge gained from the design and implementation of these systems. There are principles which could enable us to understand how much parallelism will help an application or a software system. Parallelism is used in many activities together in order that the machine can maximize its performance or speed. In particular, it should be kept in mind. But the relationships that manage the growth not linear pace. For instance, the user intends to speed for a given n processor increased up to n times. This is an optimal solution, but it seldom occurs due to overhead communication.
Two important guidelines are here to be considered:

- Computation speed is proportional to the system's square root costs; it is never linearly increased. The faster a system gets, the costlier its speed will be (Figure).
- Speed increases with the logarithm of the number of processors (i.e.,*y= k *log(N)*) of parallel computer. Figure illustrates that concept.

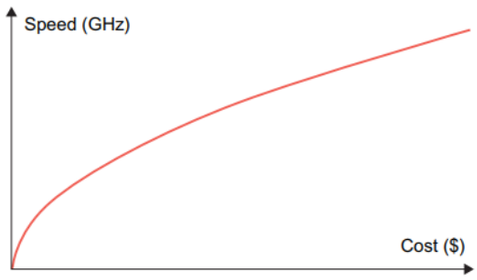


FIGURE 2.10 Cost versus speed

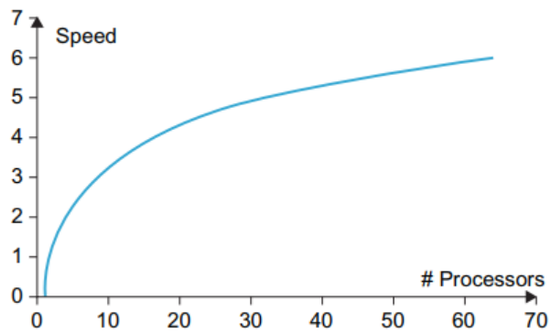


FIGURE 2.11 Number processors versus speed.

2.4 Elements of distributed computing

In this portion, we broaden these principles and discuss how different tasks can be achieved by utilizing systems consisting of many heterogeneous computer systems. They address what is commonly called distributed computing and more specifically, in terms of the software designer, they present the most important guidelines and principles for the implementation of distributed computing systems**.**

2.4.1 General concepts and definitions

Distributed computing work explores the models, architectures, and algorithms used in the design and management of distributed systems. We use the one as a general definition of the distributed system proposed by Tanenbaum

A distributed system is a collection of independent computers that appears to its users as a single coherent system

It is definitely the ideal form of a distributed system that completely hides from the user the "implementation details" of creating a powerful system from many more basic systems. Within this section, we concentrate on the architectural models that use and present a coherent system to use independent computers. The fundamental step in all distributed computer architectures is the concept of computer communication. The distributed system is an application that performs protocol collection to coordinate several communication network action processes such that all components cooperate in order to perform one or a number of similar tasks. The collaborating computers can control both remote and local resources in the distributed system over the communication network. Multiple existence Individual computers in the distributed network are transparent to the user. The user does not know the work is performed in remote areas on different machines. Coulouris definition of Distributed system

A distributed system is one in which components located at networked computers communicate and coordinate their actions only by passing messages.

The distributed system components communicate with some kind of message passing, as defined in this above description. This term covers several models of communication.

2.4.2 Components of a distributed system

Nearly all large computing systems are distributed now. Systems are distributed. The distributed system is "a set of independent machines that present to the user as one coherent system." Information processing is distributed on several machines instead of being confined to a single computer. The overviews of the various layers involved in the delivery of distributed system services are presented in Figure 2.12.

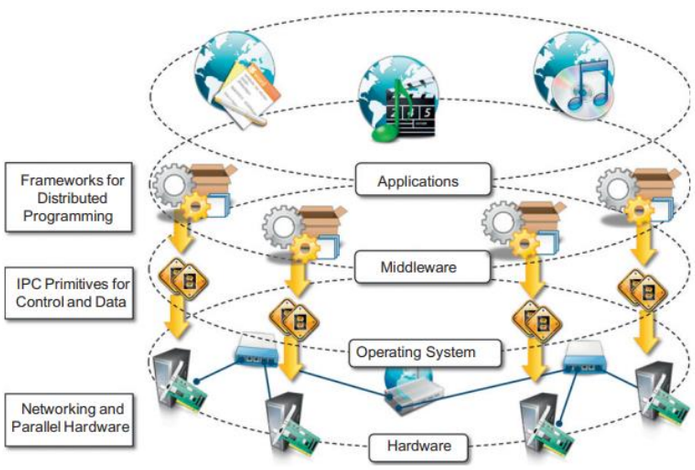


FIGURE 2.12 A layered view of a distributed system

Reference from "Mastering Cloud Computing Foundations and Applications Programming" by Rajkumar Buyya)

At the lowest level the physical infrastructure is computer and network hardware, which is explicitly supervised by the operating system that provides the essential interprocess communication (IPC) services, the scheduling and management of the process, as well as the management of resources in file systems and local systems. Combined, those other two layers become the framework on the top for specialized software to convert a number of networked computers to distributed system

The implementation of well-known operating system principles and many more on a hardware and network level allows heterogeneous components and their structure to be integrated easily into a consistent, unified framework. For instance, connectivity in the network among various devices is governed by standards, allowing for smooth interaction. IPC services at operating system level have been introduced with the introduction of standardized communication protocols like TCP / IP, User Datagram Protocol (UDP) as well as other.

The middleware layer utilizes such services to develop and deploy distributed applications in a consistent environment. When using the services provided by the operating system, middleware creates its own protocols, data formats and programming language or frameworks to create distributed apps. This layer offers support to programming paradigms for distributed systems. They all constitute an interface that is entirely independent of the underlying operating system and covers all of the lower layers' heterogeneities.

The applications and services designed and built for middleware use reflect the upper part of the distributed system stack. These can be used for many reasons. Sometimes they can view their functionality through a web browser in the form of graphical interfaces (GUIs). For example, the utilization of web technology is highly preferred in the context of a cloud computing system not only for interface applications distributed apps with consumers, but also for platform services to create distributed systems. An excellent example is the IaaS provider, for instance Amazon Web Services (AWS), who provides virtual machine creation facilities, organizes it together into a cluster and deploys applications and systems on top. Figure gives an example of how a cloud computing system's general reference architecture of a distributed system is contextualized.

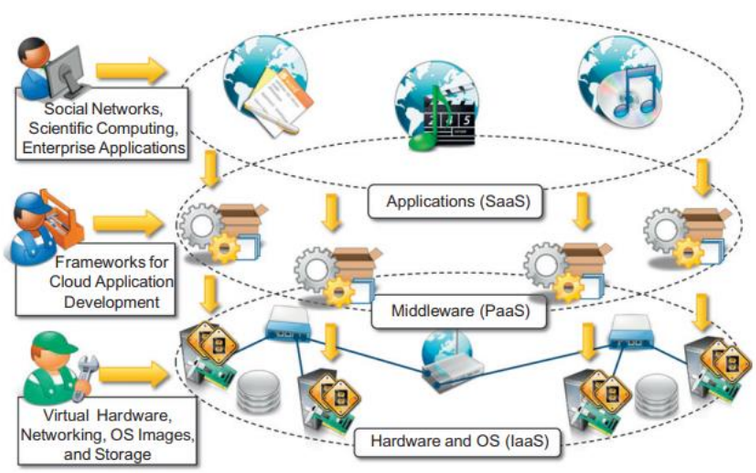


FIGURE 2.13 A cloud computing distributed system.

Reference from "Mastering Cloud Computing Foundations and Applications Programming" by Rajkumar Buyya)

2.4.2 Architectural styles for distributed computing

Distributed systems are also complex software components that are distributed through many devices by design. It is important that these processes are structured appropriately to overcome their complexities. There are various ways to see how a distributed System is organized, but it is readily apparent to distinguish between the logical organization and actual physical advancement of software components.
The distributed systems are structured mainly by the software constituent components of the network. These software architectures inform us the structure and interaction of different components of the program. In this chapter we will first concentrate on some common approaches to the organization of computer systems.
To create an effective distributed network, software modules need to be mounted on specific computers and put on them. There are a number of different choices to make. Sometimes named device architecture the final instantiation of a software architecture. In this chapter we examine traditional central architectures in which the majority of software components (and therefore features) are implemented by a single server, while remote clients can access that server with simple communication methods. Moreover, we call decentralized systems where computers perform more or less the same roles and hybrid organizations.

Architectural Styles

Originally we find the logical arrangement of distributed systems into software modules, also called software architecture. Computer architectural work has evolved dramatically and the design or adoption of architectures is now widely recognized as essential to large system growth.

The idea of an architectural style is important for our discussion. Such a style is formulated in components, the connections between components, the data exchange between components and the configuration of these elements together in one system. A part is a modular unit with well-defined interfaces and which can be replaced in its environment. As discussed below, the key point regarding a distributed system component is that the component can be substituted if its interfaces are known. A much more complex term is a connector, usually defined as a communication, coordination or co-operation mechanism between the components. For example, the (remote) procedure calls, message passing, or streaming data can be generated by a connector.

The architectural styles are organized into two main classes:

- Software architectural styles
- System architectural styles

The first class is about the software's logical structure; the second class contains all types representing the physical structure of the software systems represented by their major components.

2.4.3.1 Component and connectors

Component and connectors visions describe models consisting of elements with a certain presence over time, such as processes, objects, clients, servers, and data storage. In addition, component and connector models provide interaction mechanisms, such as communication links and protocols, information flows, and shared storage access, as components. These interactions also are conducted across complex infrastructure, such as middleware systems, communication channels, and process schedulers. *Component* is a behavioral unit. The description of the component defines what the job can do and needs to do. *Connector* is an indication that one component is usually linked by relationships such as data flow or control flow. Connector is a mechanism.

2.4.3.2 Software architectural styles

Styles and patterns in software architecture define how to organize the system components to build a complete system and to satisfy the customer's requirements. A number of

software architectural styles and patterns are available in the software industry, so that it is necessary to understand the special design of the project.

These models form the basis on which distributed systems are built logically and discussed in the following sections.

Data centered architectures

At the center of this architecture is a data store, which is often accessed through other components that update, add, delete or modify the data present in the store. This figure 2.1 4 shows a typical data-centric style. A central repository is accessed by the client software. Variation of such a method is used to turn the repository into a blackboard, as client's data or client's interest data change customer's notifications. It will facilitate integrality with this data-centered architecture. This allows for changes to existing components and the addition of new customer components to the architecture without the permission or concern of other customers. Customers may use blackboard mechanisms to transfer data.

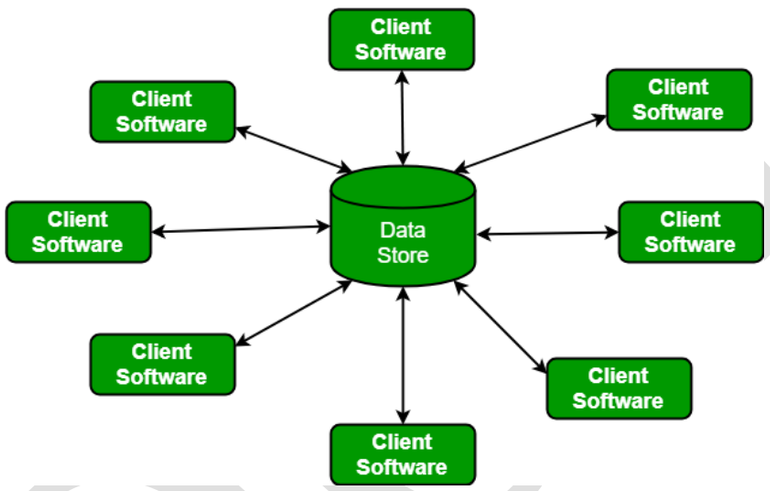


FIGURE : 2.14 Typical Data-Centric Style

A repository architecture consists of a central (often a database) data structure and an independent collection of components which function on the central data structure
For example, blackboard architectures, where a blackboards serve as communication centers of knowledge sources and repositories for multiple applications, include repository architectures. Repositories, including software development, CAD, are important in data integration and are implemented in a variety of applications.
In the blackboard style the principal components are shown in the figure 1. The problem is solved from several sources of knowledge. The problem is solved by each source of information and its solution, partial solution or suggestion is written on the blackboard. Around the same time, any other source of knowledge either modifies or extends the solution given by the previous source of knowledge or writes to solve the problem itself. Control shell is used to organize and monitor the activities of information sources to prevent them from creating a mess that may differ from the current course of the project. This is the management, monitoring and control of all activities conducted during the troubleshooting session through the control shell.
Scalability i.e. is one of the benefits of this architectural design. Source of knowledge can easily be added or removed as needed from the program. Sources of knowledge are independent and thus workable simultaneously under the control element constraint. It is a problem in this architecture that it is not known in advance when to stop the solution finding process because more and more refinement is always feasible. Pair of synchronizes. It is difficult to achieve multiple sources of knowledge

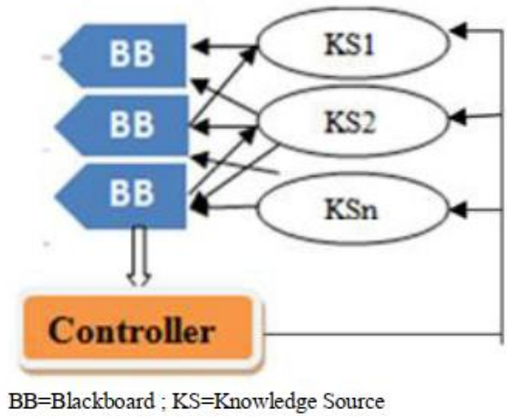


FIGURE 2.15 . Blackboard architecture

Data-flow architectures

Data Flow Architecture converts input data into output data via a collection of computational or deceptive elements. It's a computer architecture that has no program counter, so the execution is unpredictable, meaning that behaviors are indeterminate. Data flow is an aspect of the Von-neumann computer model consisting of a single program counter, sequential execution and control flow that defines fetch, execution, and commit order.

This architecture has been applied successfully.

- The architecture for data flow eliminates development time and can quickly switch from design to implementation.
- It aims primarily to achieve the reuse and alteration features.
- Through the architecture of data flows, the data can be flowed without cycles into the graph topology or into a linear structure.

The modules are implemented in two different types:

1. Batch Sequential
2. Pipe and Filter

Batch Sequential

- Batch sequential compilation in 1970 was considered to be a sequence process.
- In Batch sequential, Separate program systems are run sequentially and the data is transferred from one program to the next as an aggregation.
- This is a typical paradigm for data processing.

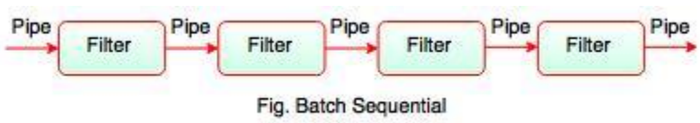


FIGURE 2.16 Batch Sequential

- The diagram above shows the batch sequential architecture flow. It offers simpler sub-system divisions and each subsystem can be an independent program which works on input and produces output data.
- The biggest downside of the sequential batch architectures is the lack of a concurrency and interactive interface. It provides high latency and low throughput.

Pipe and Filter

- Pipe is a connector that transfers data from one filter to another filter
- Pipe is a directional data stream which a data buffer implements to store all data, before the following filter has time to process it.
- It moves the data from one data source to a one data sink.
- The stateless data stream is pipes.

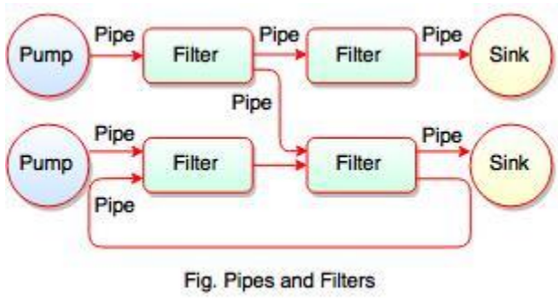


FIGURE 2.17 Pipe and Filter

- The figure above shows the sequence of the pipe filter. All filters are the processes running concurrently, which means they can run as separate threads or coroutines or be fully located on various machines.

- Every pipe has a filter connection and has its own role in filter's operation. The filters are robust, with the addition and removal of pipes on runtime.

- Filter reads the data from their input pipes, performs its function on these data and places the result on all output pipes. If the input pipes are not enough data, the filter only waits for them.

Filter

- Filter is a component**.**
- The interfaces are used to flow in a variety of inputs and to flow out a variety of outputs.
- It processes and refines the data input.
- The independent entities are filters.
- Two ways to create a filter exist:

  1. Active Filter
  2. Passive Filter

- The active filter creates the pipes' data flow.
- Data flow on the pipes is driven by the passive filter.

- Filter does not share state with other filters.
- The identity of upstream and downstream filters is unclear.
- Separate threads are used for filters. It may be threads or coroutines of hardware or software.

Advantages of Pipes and Filters

- Pipe-filter provides high throughput and excessive data processing efficiency.
- It allows maintenance of the system simpler and provides reusability.
- It has low connectivity and flexibility through sequential and parallel execution.

Disadvantages of Pipe and Filter

- Dynamic interactions cannot be accomplished with Pipe and Filter.
- For data transmission in ASCII format it needs a low common denominator.
- Pipe-filter architecture can be difficult to dynamically configure.

Virtual machine architectures

Virtual machine architecture refers to a structured system interface specification, including the logical behavior of the resources handled by interface. Implementation defines an architecture's real implementation. The levels of abstraction are the design layers, be they hardware or software, each associated with a different interface or architecture. In systems using this design, the general interface is as follows: the software (or application) determines its functions and state, as interpreted by the virtual machine engine, in an abstract format.
The implementation is based on an understanding of the program. The engine retains an internal structure of the state of the program. The rule-based systems, interpreter and command-language processors are very common examples within this group. The simplest type of artificial intelligence is rule-based systems (also known as production systems or expert systems). A rule-based program requires rules for representing knowledge with system-coded knowledge .The concepts of a rule-based system depend almost entirely on expert systems that mimic human expert reasoning in the resolution of a wisdom-intensive question. Instead of describing knowledge as a collection of true facts in a static and declarative way, a rule-based structure portrays knowledge as a series of laws that say what to do or not. The networking domain provides another fascinating use of rule-based systems: network intrusion detection systems (NIDS) also are based on a set of rules to classify suspicious behaviors associated with potential computer device intrusions.

Interpreter Style

The interpreter is an architectural style that is ideal for applications that can not specifically use the most adequate language or machine to execute the solution. The style comprises a few parts that are a program we attempt to run, an interpreter we are attempting to interpret, the program's current state and the interpreter and the memory portion that will carry the program, the program's actual state and its current state. Calls for procedures for communication between elements, and direct memory access, are the connector for the architectural style of an interpreter.

Four compositions of the interpreter:

- Engine interpreter: the interpreter 's job is completed
- Area of data storage: contains the pseudo code
- Data store field : Reports current interpreter engine state
- external data structure: Tracks the development of the source code interpreted

Input: the input to the portion of the interpreted program is forwarded to the program state where the interpreter is read by the program
Output: The output of the Program is placed in the state of the program where the data is interpreted interface system part
This model is quite useful in designing virtual machines for high-level programming (Java, C#) and scripting languages (Awk, PERL, and so on).

- Application portability and flexibility throughout different platforms
- Virtualization. Machine code for one hardware architecture can be executed on another via the virtual machine.
- System behavior defined by custom language or data structure; facilitates the development and comprehension of software.
- Dynamic change supports (Efficiency)
- Usually the interpreter only has to translate the code to a

- Intermediate representation (or not translate at all), so that it takes considerably less time to test change.

An interpreter or virtual machine does not have to follow all the instructions of the source code that it processes. It can refuse in particular to execute code which breaches any security limitations under which it operates. For example. JS-interpreter is a JavaScript interpreter that is sandboxed in JavaScript. The arbitrary JavaScript code can be executed line by line. Performance of the main JavaScript environment is completely isolated. Multi-threaded competitor JavaScript without the use of web workers are available in JS-Interpreter instances.

Call & return architectures

The most frequently used pattern on computer systems was Call & return architectures style. The mechanism of call or function call includes main programs and subroutines, remote procedure calls, object-oriented systems, and layered systems. They all come under the call and return in the style of architecture.

Top-Down Style.

The top down approach is basically the breakdown of a systems in order to get details on its compositional sub- structures in a reversing engineering manner (also known as stepwise design and stepwise refining and in some cases used in a decomposition fashion). In a top-down approach, an overview of the system is made and all first-level subsystems are defined, but not comprehensive. Each subsystem is then further modified, often at various other subsystem levels, until the full specification has been reduced to smaller elements. The "black boxes" helps define a top-down layout that is easier to manipulate. Nonetheless, black boxes could not explain or be precise enough to validate the model effectively. The big picture starts with the top down approach. It divides into smaller pieces.

A top-down approach involves dividing the problem between tasks and separating tasks into smaller subtasks. In this approach, we first develop the main module and then develop the next stage modules. This process is followed until all modules have been created.

Object-Oriented Style.

The object-oriented programme, instead of actions & logic, is a programming language paradigm structured around objects & data. In order to take data, process it and generate results, a traditional procedure program is organized. The program was centralized in terms of logic instead of data. They focus object-orientated programming on objects and their manipulation rather than on the logic that creates them.

The first phase in the OOPs is data modeling that includes defining, manipulating and relationship involving all the objects. The modeling of data is a planning phase that requires tremendous attention. We have a method to produce those objects once every object involved in the program has been identified. It is known as the class mechanism. A class includes data or properties and the logical sequence of methods for manipulating data. Every way is separate and the rationale which has already been established in other methods should not be repeated.

Architectural styles based on independent components

A number of independent processes / objects communicating via messages are part of the independent component architecture. The messages can be transmitted via publish / subscribe paradigms for a given or unnamed participant.
Components typically do not control each other by sending data. It can be changed as the components are isolated.
Examples: Event systems and communication processe are subsystems of this type.

Event systems

This paradigm separates the implementation of the component from the knowledge of component names and locations. The pattern of the publisher / subscriber, where:

Publisher(s): advertise the data you would like to share with others
Subscriber(s): Receipt of published data register interest.

For communications between components, a message manager is used. Publishers send messages to the manager who redistributes them to subscribers.

Communication process
The architectural type of communication process is also known as Client-Server architecture.
Client: begins a server call that requests for some service.
Server: provides client data.
Returns data access when the server works synchronously

2.4.3.3 System architectural styles

The Client-server and Peer-to - peer (P2P) are the two key system level architectures we use today. In our everyday lives, we use these two types of services, but the difference between them is often misinterpreted.

Client Server Architecture

Two major components are in the client server architecture. The server and the client. The server is the location of all transmission, transmission, and processing data, while the client can access the remote server services and resources. The server allows clients to make requests, and the server will reply. In general, the remote side is managed only by a computer. But in order to be on the safe side, we load balancing techniques using several servers.

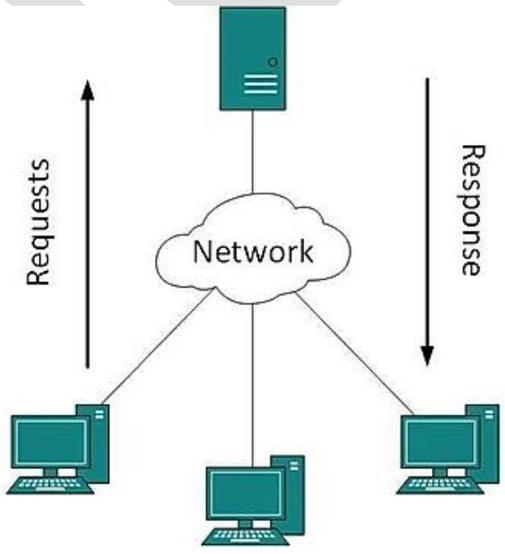


FIGURE 2.18 Client/server architectural styles

The Client Server architecture is a standard design feature with a centralized security database. This database includes information on security, such as credentials and access details. Absent security keys, users can't sign in to a server. This architecture therefore becomes a bit more stable and secure than Peer to Peer. The stability comes because the security database can make for more efficient use of resources. However, on the other hand, the system could crash because only a small amount of work can be done by a server at a certain time.

Advantages:

- Easier to Build and Maintain

- Better Security
- Stable

Disadvantages:

- Single point of failure
- Less scalable

Peer to Peer (P2P)

There is no central control in a distributed system behind peer to peer. The fundamental idea is that at a certain time each node can be a client or a server. If something is asked from the node, it could be referred to as a client and if something arrives from a node it could be referred to as a server. Usually every node is called a peer.

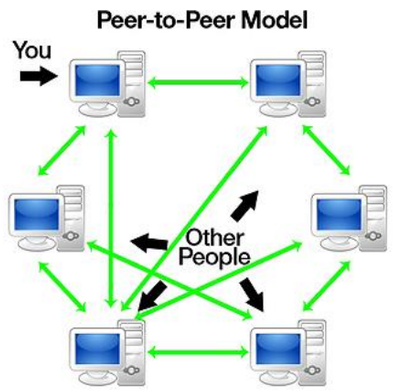


FIGURE 2.19 Peer to Peer (P2P)

Any new node will first join this network. Upon joining, they may either request or provide a service. A node's initiation phase (joining a node) can vary based on network's implementation. There are two ways a new node can learn what other nodes provide.

Centralized Lookup Server-The new node must register and mention the services on the network with the centralized look up server. So, just contact the centralized look up system anytime you need to have a service and it will direct you to the appropriate service provider.

Decentralized System-A node that seeks particular services will, broadcast and request each other node in the network, so that the service provider can respond.

A Comparison between Client Server and Peer to Peer Architectures

BASIS FOR COMAPARISON	CLIENT-SERVER	PEER-TO-PEER
Basic	There is a specific server and specific clients connected to the server	Clients and server are not distinguished; each node act as client and server.
Service	The client request for service and server respond with the service.	Each node can request for services and can also provide the services.
Focus	Sharing the information.	Connectivity.
Data	The data is stored in a centralized server.	Each peer has its own data.
Server	When several clients request for the services simultaneously, a server can get bottlenecked.	As the services are provided by several servers distributed in the peer-to-peer system, a server in not bottlenecked.
Expense	The client-server are expensive to implement.	Peer-to-peer are less expensive to implement.

Stability	Client-Server is more stable and scalable.	Peer-to Peer suffers if the number of peers increases in the system.

2.4.4 Models for interprocess communication

A distributed system is a set of computers that behave as a cohesive network to its users. One important thing is that differences between the different computers and how they interact are often hidden from users. This then gives the user a single image of the system. The OS hides all communication details among the user's processes. The user does not know that many systems exist. The inter-process communication called IPC is done by various mechanisms in distributed systems and for different systems, these mechanisms may vary. Another significant aspect is that users and applications can communicate consistent and uniform with a distributed system

Communications between various processes is the essence of all distributed systems and it is important to understand how processes can share information on different machines. In order to exchange data between two application and processes, Inter Process Communication or IPC as its name implies. Processes may be on or in a different location on the same machine. Distributed systems communication often depends on low-level messaging as the underlying network provides. Communication is difficult to communicate through message passing than primitive communication based on a shared memory available on non-distributed platforms

Inter-process Communication (IPC) is a method for the communication and synchronization of systems. The communication between such processes can be regarded as a cooperation method among them .These three methods allow processes to communicate with each other: shared memory, remote procedure call (RPC), and message passing. In distributed systems, IPCs with sockets are very popular. In short, an IP and a port number are a pair of sockets. Every one requires a socket for two processes to communicate.
If a server daemon runs on a host, it listens to its port and manages all customer requests that are sent to the client port (server socket). To submit a message, a client has to be aware of the IP and server port (server socket). Once a client starts communication with the servers and is freed once communication is over, the OS kernels also provide the client's port.
Although communication by sockets is popular and effective, it is considered low because sockets allow unstructured streams of bytes only to be transmitted between processes. The data transmitted as a byte stream is organized by client and server applications.

2.4.4.1 Message-based communication

Message abstraction is essential in the development of models and technologies, enabling distributed computing. Distributed system is a system in which components reside in networked communication and only through moving messages coordinate their functions. Within this message, any confidential data transferred from one individual to another is identified. It includes any type of data representation with size and time constraints while invoking a remote process or an object instance sequence or a common message. That is why the 'message-based communication model,' which is based on data streaming abstraction, can benefit from referencing various inter-process communication models.

Despite the abstraction that is shown to developers in programming the coordination of common components, various distributed programming models use this type of communication. Below are several major distributed models of programming using message templates,

Although communication by sockets is popular and effective, it is considered low because sockets allow unstructured streams of bytes only to be transmitted between processes. The data transmitted as a byte stream is organized by client and server applications.

Message Passing

The principle of message is implemented in this model as the main abstraction of the model. Units that exchange explicitly encoded data and information in the form of a message. The structure and message's content differ or vary according to the model. Message Passing Interface and OpenMP are significant examples of this model type.

Remote Procedure Call

This model examines the keys to the procedure call outside the limits of a single process, suggesting system execution in remote processes. It includes the main client-server. A remote process maintains a server component, allowing client processes to call on processes and returns the execution output. The messages, which are generated by the implementation of Remote Procedure Call (RPC), collect information about the method on its own and execute the arguments required for it and also return the values. The usage of messages referred to as the marshalling of the arguments and the return values.

Distributed Objects

This is an implementation of the object-orientated model Remote Procedure Call (RPC), which is understood in context for remote invocation methods that are expanded through objects. Each process assigns a series of interfaces that are remotely accessible. The client process can request and invoke the methods accessible via these interfaces. The standard runtime infrastructure transforms the local method invocation into a remote request call and collects the execution results. The interaction between the caller and the remote process takes place via messages. This model is stateless by design, the complexity of object state management and lifetime are illustrated by distributed object models. Common Object Request Broker Architecture (CORBA), Component Object Model (COM, DCOM and COM+), Java Remote Method Invocation (RMI), and .NET Remoting are some of the most important Distributed object infrastructure examples.

Active objects

Programming models based on active objects, however accessible, contain by definition the existence of instances, regardless of whether they are agents of objects. This implies that objects have a special control thread that allows them to show their activity. Such models often manually use messages to execute functions and the message is connected to a more complicated semantics.

Web Services

Web service technology offers an alternative to the RPC framework over HTTP, allowing the interaction of established components with various technologies. A web service is exposed as a remote object stored on a web server and invocations of the system are converted into HTTP requests packed using a particular protocol. It must be remembered that the concept of message is a basic abstraction of communication between interprocesses and is used either implicitly or explicitly.

2.4.4.2 Models for message-based communication

Point-to-point message model

A software or application is designed from point to point (PTP) around the idea of message queues, senders and receivers. That message is sent to a certain queue and customers receive messages from the queue(s) that are set up to hold their messages. All messages sent to them are kept until the messages are consumed or until messages expire.

Publish-and-subscribe message model

Publish-subscribe is a message service. It describes a particular type of communication between components or software modules. The name is chosen to represent the most important features of this communication model.

Software modules interact directly with each other in straightforward interactions using mechanisms and media that are recognized by all parties. For communication needs becoming more complex or demanding, other systems of communication have developed. Publish-subscribe is one such subscription and only one of many.

The core ideas for Publish-Subscribe

- There are not necessarily software components that know with whom they interact.
- The data producers publish the data in the whole network.
- Data consumers subscribe to the system and receive data from it as a whole.
- Information is named such that the available information can be defined by software modules. Sometimes this label is called the topic.

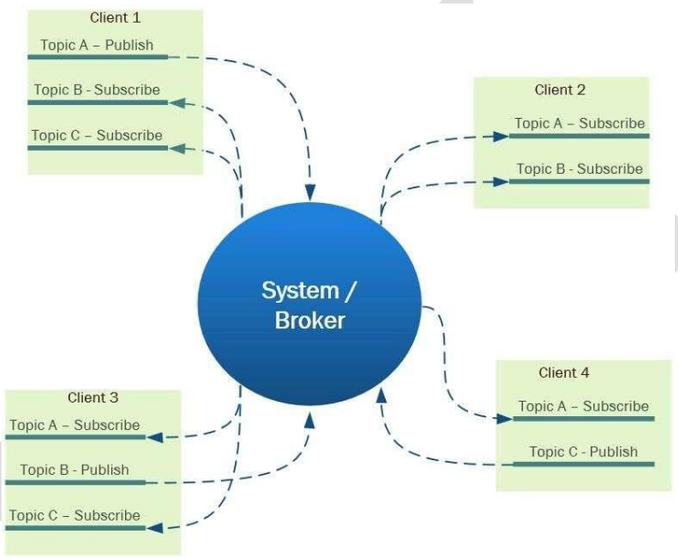


FIGURE 2.20 Publish-and-subscribe message model

A central software module ensures that all data, publishing and subscription are administered and matched. The "broker" is commonly referred to. Often brokers are a network of cooperating software modules and software modules that use broker services are called clients.

Clients that publish and also subscribe "register" with the broker for communication paths to manage, clients and other housekeeping activities to authenticate.

Message delivery to subscribers filtered in relation to content rather than topic. Instead of or with the topic, this can be used. Only a few Publish-Subscribe systems have implemented this.

Data can be "persistent," because subscribers who register on the network after last publishing the data will have the last published data on the specific topic.

Request-reply message model

A request reply messaging model is different from a traditional pub / sub or P2P model, which publishes a message to a topic or queue and enables clients to receive the message without providing the reply response.

Request reply messages may be used when a customer sends the requested message for information from a remote client application or for a processing action to be carried out. Once the client application receives the request message, it receives the necessary

information or carries out the requested action. The information is then applied to a reply message, or a confirmation of completion of the task is submitted in response to the request.

2.5 Technologies for distributed computing

In this section, we are implementing appropriate technologies which give realistic implementation of interaction models which depend mainly on message-based communication. Such systems include remote procedure call (RPC), distributed object frameworks and services-oriented computing.

2.5.1 Remote procedure call

Remote Procedure Call (RPC) is a protocol that can be used by a program to request the service on the other computer of the network without the need for the details of the network. RPC is used in remote systems to call other processes such as a local system. Sometimes, a procedure call is also called a function call or a subroutine call.
The client-server model is utilized by RPC. The program you are requesting is a client and the service provider is the server. Like an on-going or local procedure call, the RPC is a synchronous operation which requires a suspension of the requesting program until the remote procedure result are returned. Nevertheless multiple RPCs can be performed concurrently by using lightweight processes or threads that share the same space.

In Remote Procedure Call software, interface definition (IDL) language, the specification language used to describe an application programming interface (API) of a software component. IDL provides in that case a bridge between the two ends of the connection, which may be connected by different computer languages and operating systems (OSes).

RPC message procedure

When program statements using the RPC framework are compiled into an executable program, the compiled code includes a stub representing the remote procedure code. The stub receives the request and transmits it to a client runtime program on the local computer when the program is running and a call is issued. Once the client stub is first invoked, it contacts a name server to specify where the server is located.

The Client Runtime Program is familiar with how to address the remote computer and server application and sends the message over the network that requests the remote procedure. The server also has a runtime program and stub this interface with the remote procedure itself. Response-request protocols returned in the same way

When making a Remote Procedure Call

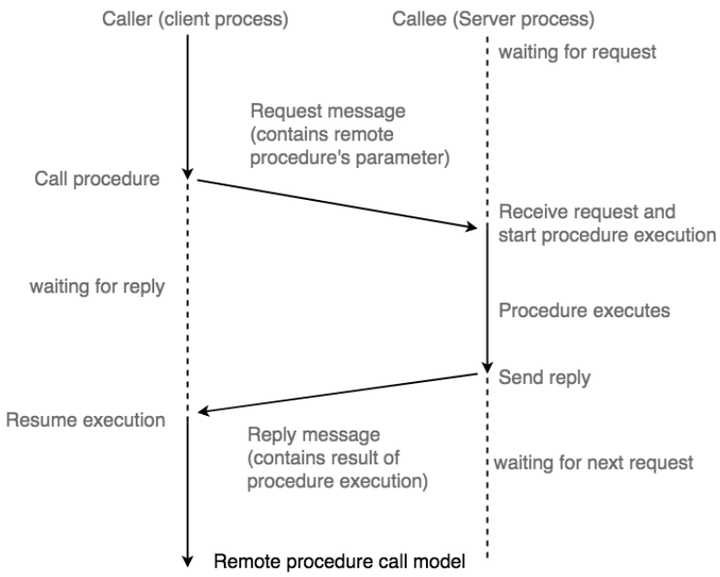


FIGURE 2.21 Remote Procedure Call

1. The calling environment is terminated, procedural parameters are transferred across the network and the procedure is execute in the environment.

2. When the procedure is completed and results are produced, its results are returned to the calling environment where it resumes to execute as if back from a regular procedure call.

NOTE: RPC is particularly suitable for the client-server interaction (e.g. query-response) between the caller and callee. The client and server are not both execute simultaneously in the concept. Instead, it jumps back and forth from the caller to the callee.

Working of RPC

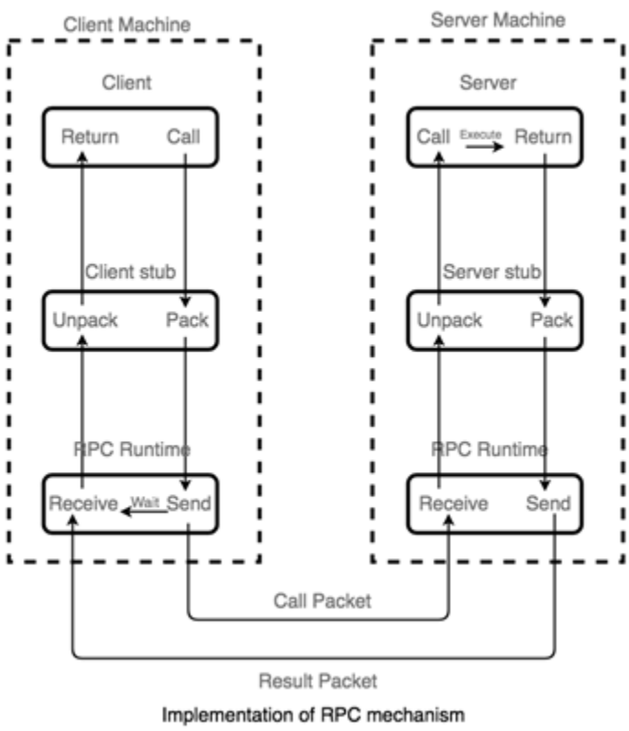


FIGURE 2.22 Working of Remote Procedure Call

In an RPC there will be the following steps:

1. A client invokes a client stub procedure which usually passes parameters. The client stub is resides in the own address area of the client.
2. The client stubbed marshalls (pack) the parameters in a message. Marshalling involves converting the parameter representation to a standard format and copying each parameter to the message.
3. The client stub transfers the message to the transportation layer and transfers it to the remote server.
4. On the server, a transport layer transfers the message to a server stub to demarshall (unpack) the parameters and uses the standard procedure call method to call the desired process routine.
5. When the server procedure finishes, it returns to the stub server (for example, through a normal procedure call return). The stub server then transmits the message to the transport layer.
6. The transport layer returns the resulting message to the client transport layer, which returns the message to the client stub.
7. The client stops the return parameters and returns the execution to the caller.

2.5.2 Distributed object frameworks

The most well-known ways to develop distributed systems or frameworks are client-server systems. An extension to this client-server model is a distributed object framework. It is a library where distributed applications can be built using object-oriented programming. Distributed objects in distributed computing are objects that are distributed in different address spaces, in different processes on the same computer, or even in multiple network-connected computers. However, they perform around each other via data sharing and invoking methods. It also involves transparency of location where remote objects appear the same as local objects. With remote method invocation, usually message-passing, the main method of distributed communication for objects is by sending a message to a different object within a remote machine or process to perform some task. The results are returned to the object that you call.

The Remote procedure Call (RPC) method applies to distributed environments the common programming abstraction of the procedure call allowing the call process to call the remote node as local.

Remote method invocation (RMI) resembles RPC for distributed objects, but has additional advantages in terms of the use of object-oriented programming concepts for distributed systems and extends to the distributed Global environment the concept of the object reference and enables the use of object references such as Parameter in remote invocation.
Remote Procedure call – The client calls procedures in a different server program
Remote method invocation (RMI) – an object can invoke object methods in a different process
Event notification – Objects receive notification of events in other objects they have registered for

2.5.2.1 Examples of distributed object frameworks

Distributed programming environment (DPE)-software can be developed and managed by programmers distributed around the world Practically supporting distributed object computing, such as Internet, on the distributed system. The research is aimed at developing a programming environment that supports an effective distributed environment programming. A system that uses distributed objects provides the distributed and parallel programming environment with flexible and scalable programming. A lot of them are distributed object computing systems like CORBA, DCOM, and Java are supported

Common object request broker architecture (CORBA)

Common Object Request Broker Architecture (CORBA), is a consortium of more than 800 companies that supports the most famous middleware. With the exception of Microsoft, this consortium is the majority of computing companies which has its own Distributed

Component Object Model (DCOM) object broker. The object bus of CORBA sets and defines the object components Interoperability. An object bus is the Object Request broker (ORB). CORBA is conducted to discover and interoperate object components within an object bus. CORBA supports transparent object references through object interfaces between distributed objects

CORBA is essentially a design specification for an Object Request Broker (ORB) that provides an ORB mechanism to allow distributed objects, either locally or on remote devices, written in different languages or in various network locations, to communication with each other.

The CORBA Interface Definition Language or IDL enables language development, location-independent interface development and distribution of distributed objects. The application components can communicate with each other via CORBA, regardless of where they are or who designed them. CORBA ensures transparency of the location in order to execute these requests.

CORBA is usually described as a "software bus" because the objects are located and accessed via a software communication interface. The following illustration identifies the main components in the implementation of CORBA.

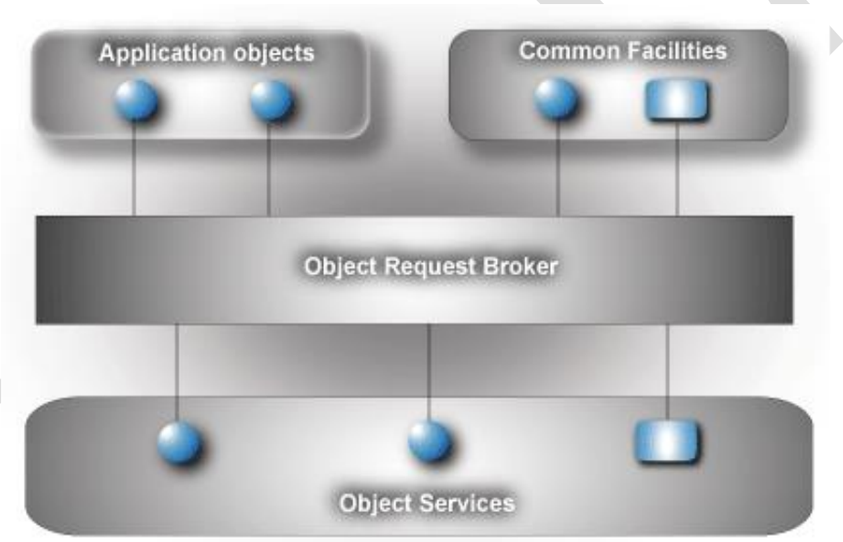


FIGURE 2.23 Common object request broker architecture (CORBA)

A well-defined object-oriented interface ensures data transmission from client to server. The Object Request Broker (ORB) sets the target object's location, sends the request to it and returns the caller with any response. With this object-oriented technology, developers can use characteristics such as legacy, encapsulation, polymorphism and dynamic binding during runtime. These features allow for the modification, modification and reutilization of applications with minimal parent interface changes. The following illustration shows how a client transmits a request through the ORB to a server:

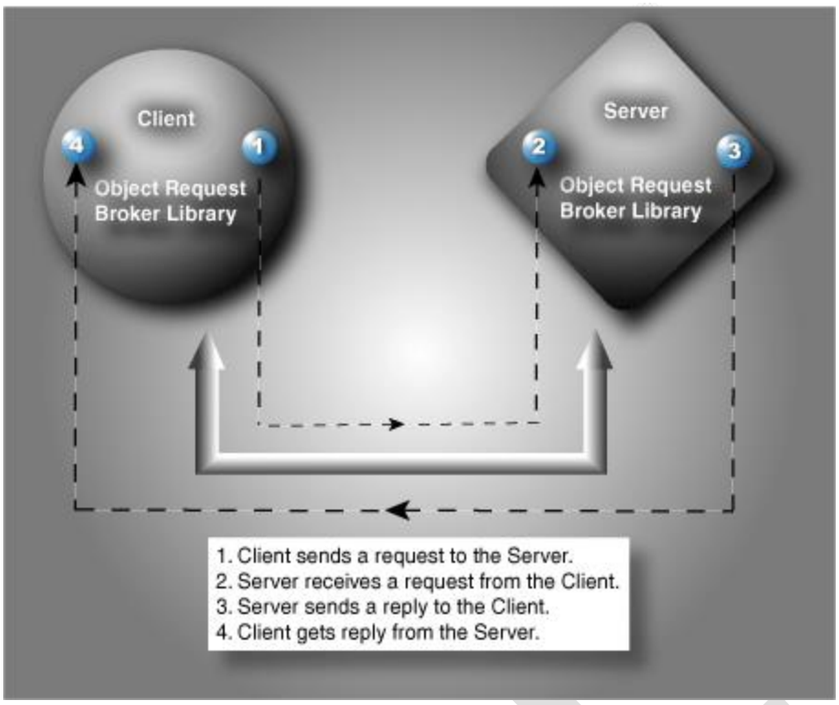


FIGURE 2.24 working of Common object request broker architecture (CORBA)

Interface Definition Language (IDL)
The Interface Definition Language is a key pillar of the CORBA standards. IDL is OMG for the definition of language-neutral APIs and provides a platform-independent line-up of distributed object interfaces. Client / server interface-standardized data and operations begin to provide a consistent approach between the CORBA environments and clients in heterogeneous environments. This mechanism is the IDL and is used by CORBA to describe the object interfaces.

For applications, IDL defines and does not take programming language as modules, interfaces, and operations. The various programming languages, such as Ada, C++ , C # and Java, provide standardized IDL mapping to the implementation of the interface.

The IDL compiler creates a stub-and-skeleton code to marshalling and unmarshalling the parameters from the network stream to memory instances of the language implemented, etc. The stub is a client proxy for an object reference accessed from a servant and is a proxy for the servant's client. Language-specific IDL stubs and skeletons can communicate with a skeleton in a language. The stub code is linked to the client code and the skeleton code is connected to the object implementation and communicates in order to implement remote operations with the ORB run time system.

IIOP (Internet Inter-ORB Protocol) is a protocol which allows distributed programs to communicate on the Internet in various programming languages. The Common Object Request Broker Architecture (CORBA) is a key element of a strategic industry standard. Using the CORBA IIOP and related procedures, a company may develop programs which are able to communicate, wherever they are and without having to understand anything about the program other than its own service, or its name, with existing or future programs of their own company or another.

Distributed component object model (DCOM/COM)

The **Distributed component object model** (DCOM) is a proprietary Microsoft communication technology between software components that are spread across networked computers. DCOM is a distributed component object model. The Distributed Component Object Model is a component object model (COM) network expansion technology that enables network-wide, interprocess communication. By managing low-level network protocol details, DCOM supports communication among objects within the network. This enables multiple processes to work together to achieve a single task by using distributed programs.

Java remote method invocation (RMI)

RMI implies Remote Method Invocation. A mechanism that permits the access / invoke of an object in one program (JVM) on another JVM. It enables remote communication between programs in Java, RMI is used to create distributed applications.

We create two programs in an RMI application: the server program (residing on the server) and the client program (residing on the client).

The server program creates a remote object and provides the client with a reference to that object (using the registry).
The client program requests remote objects and tries to invoke its methods on the server.

The following diagram shows the architecture of an RMI application.

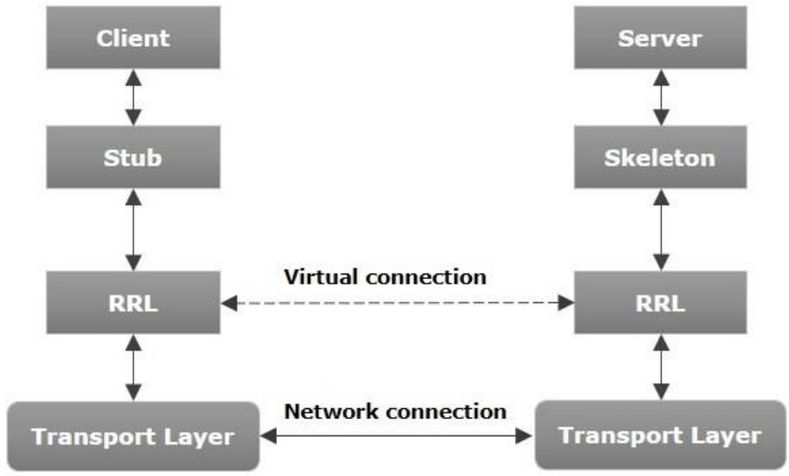


FIGURE 2.25 Java remote method invocation (RMI)

- Transport Layer− using this layer the client are connected with the server. This connection is maintained with existing connection and new connections are also created.
- Stub − the stub is the proxy of a client remote object. This is located in the client system; it serves as the client's gateway.
- Skeleton − It's the object on the server side. To pass the request on to a remote object, Stub interacts with the skeleton.
- RRL (Remote Reference Layer) − this is the layer that manages the client's remote object reference.

The following points sum up how an RMI program works.

- Whenever the client makes a request to the remote object, the stub receives the request to the RRL.
- If the RRL from the client receives the request, it uses a method called invoke () from the remoteRef object. The request is passed on the server side to the RRL.
- The server's RRL passes the client to the server skeleton that eventually calls the object on a server.
- The results are passed to the client

When a client invokes a method that supports remote object parameters, the parameters shall be enclosed in a message before they are transmitted through the network. Such may be primitive-type parameters or objects. If the primitive type is used, the parameters are assembled and the header is attached. If the parameter is an object, it is serialized. This method is referred to as marshalling.

The packed parameters are unbundled on the server side and the appropriate method is invoked. This method is referred to as unmarshalling.

RMI Registry is a name space that contains all server objects. The server registers this object into an RMIregistry (using bind method () or Rebind () methods) (methods), any time an object is created. Those are registered using a single name known as the bind name.

The client requires a reference to that object to invoke a remote object. The client must then retrieve the object from the registry by its bind name (using the lookup () method).

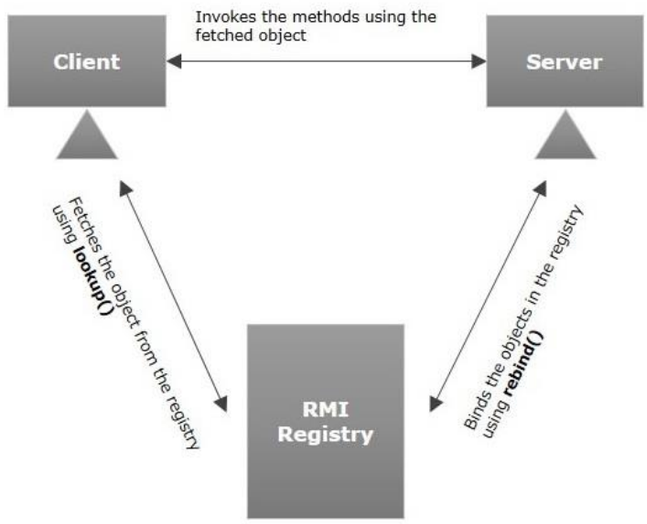


FIGURE 2.25 RMI program works

.NET remoting

The. NET remote system offers an interprocess communication between Application Domains through the use of the Remoting Framework. The programs may be installed on the same computer or on different computers on the same network. Through the use of Binary or SOAP formatters in the data stream the .NET Remoting facilitates distributed object communications over TCP and HTTP channels.

The three main components of the Remoting Framework are:

1. Remote object
2. Remote Listener Application-(Remote Object requests)
3. Remote Client Application-(makes Remote Object requests)

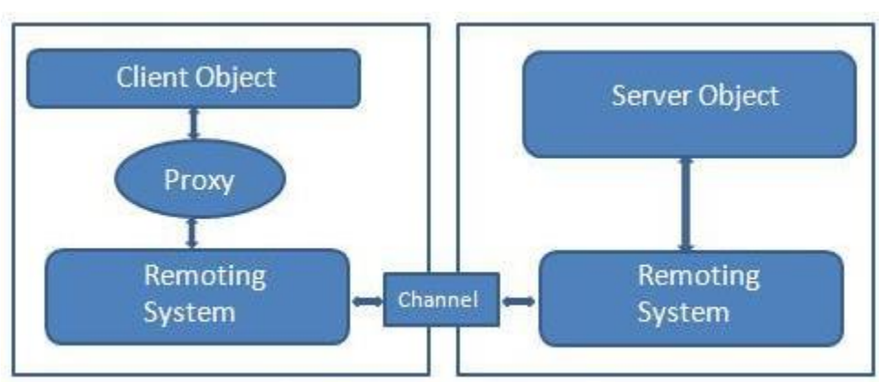


FIGURE 2.26 .NET remoting Framework

The Remote Object is implemented in the MarshalByRefObject class.

The basic workflow of.Net Remoting can be seen from the figure above. In addition, if a client calls Remote method, the client does not directly call the methods. The remote object receives a proxy and is used to call up the remote object method. The message is encrypted with a corresponding Formatter (Binary Formatter or SOAP Formatter) in the Configuration File when the proxy receives a process call from the Server then the call will be sent to the Server using a channel selected (TcpChannel or HttpChannel). The server side channel accepts the request from the proxy and sends it to the server on the Remoting system where the remote object methods are located and invoked methods on the Remote Object. Once the remote procedure is executed, every call outcome is returned to the client in the same way. It must be generated and initialized in a process known as Activation before an object instance of a Remotable type can be accessed. The activation is classified as Client Activated Objects and Server Activated Objects in two types.

2.5.3.2 Service-oriented architecture (SOA)

Service-Oriented Architecture (SOA) is a software style in which services via an interconnected communication protocol to other components are distributed by application components. The principles are separate from the manufacturers and others. Most services communicate with one another in a service-oriented architecture: through data transmission or through two or more services that coordinate the activity. It is just one term for service architecture.

Service-oriented architecture characteristics

- Business value
- Strategic goals
- Intrinsic inter-operability
- Shared services
- Flexibility
- Evolutionary refinement

Both of these core principles could be shown through an older distributed application paradigm, to service-oriented, cloud-related architecture (which also is considered to be a service-oriented architecture offshoot).

Service-Oriented Architecture Patterns

FIGURE 2.27 Service-Oriented Architecture

Each of the building blocks for the Service-oriented Architecture consists of three roles: service provider; service broker, service registry, service repository and customer / requester service.

In accordance with the service registry, a service provider is responsible for addressing whether and how services are rendered, such as security, availability, costs, and more. The type of service and any trade agreements are also decided by this role.

The service broker provides the requester with information about the service. Whoever implements the broker's scope is determined.
The service requestor locates and then adds the entries to the broker registry. You can access multiple services or you may not; this depends on the service applicant's capacity.

Implementing Service-Oriented Architecture

There are a wide variety of technologies that can be used when it comes to implementing service-oriented architecture (SOA), depending on the ultimate objective and what you're trying to achieve.
Service-Oriented Architecture is typically implemented with web services which make 'functional building blocks via standard Internet protocols' available.
SOAP, which stands for Simple Object Access Protocol, is an example of a web service standard. Briefly speaking, SOAP 'is a messaging protocol specification for standardized information sharing in computer network implementation of web services. Although SOAP was initially not well received, it has grown in popularity since 2003 and is being used and accepted more widely. Jini, COBRA, or REST are other options for implementation of Service-Oriented Architecture.
It is important to remember that architectures can be applied in different ways, including messaging, such as ActiveMQ, Apache Thrift and SORCER, "regardless of the particular technologies."

Why Service-Oriented Architecture Is Important

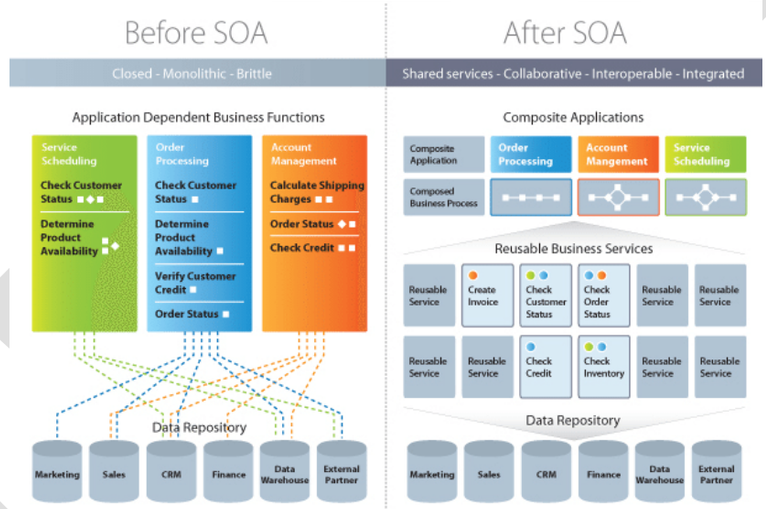


FIGURE 2.28 Before and After Service-Oriented Architecture

Service-oriented architecture has many benefits, particularly in a web-based business. Here, we will quickly discuss some of those advantages:
To create the reusable code, use Service-Oriented Architecture: Not only is it time-consumptive, but it is not necessary to reinvent your coding wheel whenever a new service or process is needed. The SOA also allows that coding languages to be used, since all runs via a central interface.
Using Service-Oriented Architecture to facilitate interaction: A common mode of communication is generated with Service-Oriented Architecture that enables different systems and platforms to operate independently of each other. By this connection, the Service-Oriented Architecture can also work around firewalls that enable "companies to share operationally important services."
Using the scalability Service-Oriented Architecture: it is vital to be able to scale a business to meet customer's requirements, however some dependencies can be prevented from using it. Use Service-Oriented Architecture reduces the interaction between customers, which makes it easier to scale.
Using Service-oriented Architecture to reduce costs: with a Service-oriented Architecture it is possible to decrease costs while still "maintaining a desired performance." It is possible for businesses to restrict the amount of analyzes they need to create custom solutions using Service-oriented Architecture.

2.5.3.3 Web services

Web Service is a structured method for distributing client-server communication on the World Wide Web. A web service is a software module that performs a variety of tasks.

You can search for the web services across the network and invoke them appropriately. The web service will, when invoked, provide the customer with the features that the web service invokes.

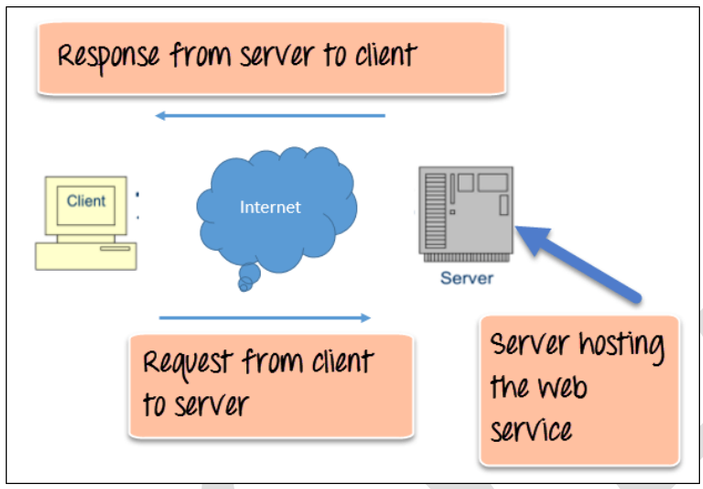


FIGURE 2.29 Web Service Architecture Diagram

The above diagram gives a very clear view of the internal working of a web service. The customer will make a series of web service calls to a server to host the current web service via request. These applications are rendered through so-called remote procedure calls. Remote Procedure Call (RPC) are calls made using the webservice hosting service procedures. Amazon provides a web service for products sold online through amazon.com, for example. The front end and layer of presentation may be in. Net or Java, but the web service will interact in either programming language.

Data transmitted between the client and the server is the primary component of a web service, namely XML. An XML is HTML equivalent, and the intermediate language that many programming languages can easy to understand and they only speak in XML while applications talk to each other. This provides a can application interface for interacting with one another in different programming languages. Web services use SOAP (Simple Object Access Protocol) to transfer XML data between applications. The data is transmitted through standard HTTP. The data that is transmitted to the program from the web server is called SOAP. The message from SOAP is just XML. The client application that calls to the Web service can be written in any programming language, as this document is written in XML.

Why do you need a Web Service?

Every day software systems use a wide range of web-based programming tools. Several apps in Java, others in Net, others in Angular JS, Node.js, etc. can be built. These heterogeneous applications most often require some kind of communication between them. Since they are constructed in different programming languages, effective communication between applications is very difficult to ensure.
Here web services are offered. Web services provide a shared platform that enables multiple applications could base on various programming languages to communicate with each other.

Type of Web Service

Two kinds of web services are mainly available.

1. SOAP web services.
2. RESTful web services.

There are some components which must be in place to make a web service fully functional. Regardless of which programming language is being used to program the web service, these components must be present.

Let us take a closer look at these elements

SOAP is regarded as an independent message protocol for transport. SOAP is based on the SOAP Messages transfer of XML data. Every message has a document called an XML document. Only the XML document structure follows a certain pattern, but the contents do not follow. The best component of Web services and SOAP is that they are all delivered via HTTP, the standard web protocol .
This is the message of a SOAP

A root element called the < Envelope > is needed in every SOAP document. The first element of an XML document is the root element.
The envelope is divided into 2 parts in turn. The first is the header and the second is the body.
The header comprises the routing data, the information to which the XML document should be sent to.
The actual message is in the body.
A simple example of communication through SOAP is given in the diagram below.

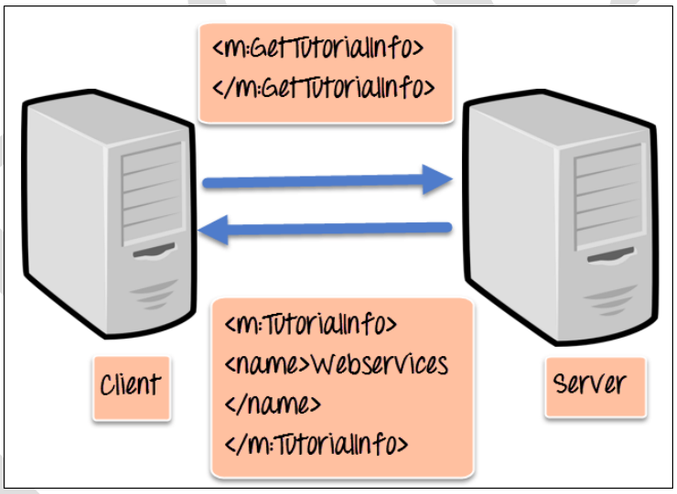


FIGURE 2.30 WSDL (Web services description language)

If it is found, a web service will not be used. The client invoking the web service should know the location of the web service.

Second, the client application wants to learn what the web service does to invoke the right web service. It is achieved using WSDL, known as the Web services description language. The WSDL file is another XML file which mainly tells the web service what its client application does. The client applications will understand the location and use of the web services by using the WSDL document.

Web Service Example
An example of a WSDL file is given below.

```
<definitions>
  <message name="TutorialRequest">
    <part name="TutorialID" type="xsd:string"/>
  </message>

  <message name="TutorialResponse">
    <part name="TutorialName" type="xsd:string"/>
```

```
  </message>

  <portType name="Tutorial_PortType">
    <operation name="Tutorial">
      <input message="tns:TutorialRequest"/>
      <output message="tns:TutorialResponse"/>
    </operation>
  </portType>

  <binding name="Tutorial_Binding" type="tns:Tutorial_PortType">
    <soap:binding style="rpc"
      transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="Tutorial">
      <soap:operation soapAction="Tutorial"/>
      <input>
        <soap:body
          encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
          namespace="urn:examples:Tutorialservice"
          use="encoded"/>
      </input>

                <output>
        <soap:body
          encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
          namespace="urn:examples:Tutorialservice"
          use="encoded"/>
      </output>
    </operation>
  </binding>
</definitions>
```

The main aspects of the above WSDL declaration are the following;

< message > – The WSDL description message parameter is used to describe the different data items per Web service operation. In this example, there are two messages, one being the "TutorialRequest" and the the other being the "TutorialResponse" operation, which can be exchanged between the web service and the client application. The TutorialRequest contains an item of the string form "TutorialID." Similarly, an element called "TutorialName," also a form string is found in TutorialResponse.
< portType >-In fact, this defines the Web service operation that is referred to in our case as known as Tutorial. This procedure will obtain 2 messages, one is input and the other is output.
< binding >-The protocol that is used contains this element. And we describe this in our case to use http (http:/schemas.xmlsoap.org/soap/http). Additional details on the body of the operation are specified, including namespace and the encoding of the message.

Universal Description, Discovery, and Integration (UDDI)

UDDI is the standard in which webservices offered by a particular provider are described, published and discovered. It provides a specification for hosting web services content.

In the previous topic, we discussed WSDL and how it provides details about the actual activities of the Web service. Yet how can a client application consider a WSDL file to recognize the various web-based operations. UDDI provides the solution and a server that can host WSDL files. This means that the client application has full access to the UDDI, a database which contains all WSDL files.

Just as a phone directory has a certain person's name, address and telephone number, so the UDDI registry is fitted with the related web service information. That's why a developer user knows where to find it.

We now also realize why web services first came about, which were to provide a platform to talk to each other with different applications.

But let's discuss some other advantages as to why web services are relevant.

Exposing Business Functionality on the network-a web server is a unit of managed code which offers client applications or end users with some type of functionality. The HTTP protocol allows this functionality to be called, so that it also can be called up via the Internet. Both programs are already available on the internet, which makes web services more useful. It ensures that the web service can be available on the Web anywhere and can provide the required functionality.

Interoperability between applications-Web services allow different applications to talk to each other and to share data and services. You can speak to each other about any kind of query. And you can now write generic code that can be understood by all applications in lieu of writing a specific code that only specific applications to understand.

A Standardized Protocol which everybody understands- Web services use a standardized industry protocol to communicate, which everybody understands. All four layers (Transport service, XML Messaging, Service Description and Service Discovery layers) use well-defined web services network stack protocols.

Reduction in cost of communication-Web providers use SOAP over HTTP protocol to implement their web-based services using the existing low-cost internet.

2.5.3.4 Service orientation and cloud computing

Service orientation is a built-in architectural approach that uses automated software resources to incorporate business processes. Such business services comprise of a collection of loosely coupled components designed to reduce dependency, designed to support a business function that is well specified. The creation of modular business service systems contributes to more versatile and effective IT systems.

Systems designed to integrate service orientation allow businesses to utilize existing resources and easily manage the unavoidable changes that a dynamic company is experiencing. There are also circumstances where the combination of a number of services is needed. It means that these combined workloads will operate with less latency than with loosely coupled parts.

Hybrid cloud environments become important because organizations constantly reinvent themselves and become more competitive, in order to respond to change. IT must be at the frontline of an innovation and transformation-based business strategy. Organizations understand that for all kinds of workloads it is difficult to find one best IT computing approach. Thereby, a hybrid cloud system is the most realistic solution.

A high degree of flexibility and modularity to make a cloud infrastructure work in the real world. To support a range of workloads and business services a cloud must be designed. One can tell when a service will be upgraded and when it can be downgraded.

Specifically, this service-based architectural design approach supports key cloud characteristics of elasticity, self-support, standard-based interfaces and flexibility in pay-as-you-go. Combining a service-oriented approach with cloud services enables businesses to decrease costs and improve flexibility in business. Scalabilities and elasticity for public and private cloud systems are interchangeable and loosely mixed.

2.6 Summary

In this chapter we introduced parallel and distributed computing as a framework on which cloud computing can be properly described. The solution of a major issue emerged out of parallel and distributed computing by using several processing components first and then multiple network computer nodes.

2.7 Review questions

1. Differentiate between parallel and distributed computing.
2. What is an SIMD architecture?
3. Explain the major categories of parallel computing systems.
4. Explain the different levels of parallelism that can be obtained in a computing system
5. What is a distributed system? What are the components that characterize it?
6. What is an architectural style and how does it handle a distributed system?
7. List the most important software architectural styles.
8. What are the fundamental system architectural styles?
9. Describe the most important model for message-based communication.
10. Discuss RPC and how it enables interprocess communication.
11. What is CORBA?
12. What is service-oriented computing?
13. What is market-oriented cloud computing?

2.8 Reference for further reading

1. Mastering Cloud Computing Foundations and Applications Programming Rajkumar Buyya ,Christian Vecchiola,S. Thamarai Selvi MK publications ISBN: 978-0-12-411454-8
2. Cloud Computing Concepts, Technology & Architecture Thomas Erl, Zaigham Mahmood, and Ricardo Puttini , The Prentice Hall Service Technology Series ISBN-10 : 9780133387520 ISBN-13 : 978-0133387520
3. Distributed and Cloud Computing: From Parallel Processing to the Internet of Things 1st Edition by Kai Hwang Jack Dongarra Geoffrey Fox ISBN-10 : 9789381269237 ISBN-13 : 978-9381269237

Unit 2
Chapter 3

Unit Structure

3.0 Objective

Virtualization abstracts hardware that can share common resources with multiple workloads. A variety of workloads can be co-located on shared virtualized hardware while maintaining complete insulation, migrating freely through the infrastructures and scaling, when required.

Businesses are generating considerable assets and efficiency through virtualization, as this results in enhanced server usage and consolidation, dynamic assignment and management of resources, isolation of working loads, security and automation. The virtualization enables self-provision on-demand services and software-defined resource orchestration, which is available on-site or off-site to any place in a hybrid cloud, according to specific business needs.

3.1 Introduction

Cloud Virtualization makes server operating system and storage devices a virtual platform. This will enable the user to also share a single physical resource instance or application with several users by providing multiple machines. Cloud virtualizations also administer work through the transformation, scalability, economics and efficiency of traditional computing.

Cloud computing virtualizations quickly integrate the key computing method. One of the key features of virtualization is that it allows multiple customers and companies to share their applications.

The virtualization environment can also be referred to as cloud-based services and applications. Either public or private this environment. The customer can maximize resources through virtualization and reduce the physical system needed.

Recently, due to the confluence of several phenomena, virtualization technology has become more interested:

Increased performance and computing capacity:

A unique corporate data center is, in most instances, unable to compete in terms of security, performance, speed and cost - effectiveness with the network of data centers provided by service provider. Since the majority of services are available on demand, in a short period of time users can also have large amounts of computing resources, with tremendous ease and flexibility and without any costly investment.

In turn, Cloud services offer you the ability to free up memory and computing power on your individual computers through remote hosting of platforms, software and databases. The obvious result, in fact, is a significant performance improvement.

Underutilized hardware and software resources.

Underutilization of hardware and software is caused by increased computing and performance and constrained or infrequent resource usages. Computer systems have become so powerful today that in certain instances those who are only a fraction of its capacity is used by an application or the system. Furthermore, when taking into consideration the company's IT infrastructure, numerous computer systems are only partly utilized whereas they can be used 24/7/365 services without interruption. For instance,

desktop PCs mainly for office automation tasks and used by administration personnel are used only for working hours. The efficiency of the IT infrastructure can be enhanced by using these resources for other purposes. A completely separate environment, which can be achieved via virtualization, is needed to provide such a service transparently.

Lack of space.

Data centers are continuously expanding with the necessity for extra infrastructure, be it storage or computing power. Organizations like Google and Microsoft are expanding their infrastructure by constructing data centers as compare as football grounds in which contains thousands of nodes .While this is feasible for IT big players, companies are often unable to build an additional data center to accommodate extra resource capacity. , Together with this situation, unused of hardware resources which led to the diffusion of a server consolidation, fundamental to the virtualization is used in the technique.

Greening initiatives

Virtualization is a core technology for the deployment of a cloud-based infrastructure to run multiple operating system images simultaneously on a single physical server. As a consolidation enabler, server virtualization reduces the overall physical server size, with the green benefits inherent.

From the perspective of resource efficiency, fewer workloads are required, which proactively reduce the space in a datacenter and the eventual footprint of e-waste. From an energy-efficiency point of view, a data center will consume less electricity with fewer physical equipment. Cooling in data centers is a major requirement and can help with high power consumption. Through free cooling methods, such as the use of air and water compared to air conditioning and cooling, data centers can reduce their cooling costs. The data center managers can save on electricity costs with solar panels, temperature controls and wind energy panels.

Rise of administrative costs.

Power consumption and cooling costs are increasing as well as IT device costs. In addition, increased demand for extra capacity that transforms into more servers in a data center leads to an increase in administrative costs significantly. Computers — especially servers — will not all work independently, but require system administrator care and attention. Hardware monitoring, flawed equipment replacement, server installation and updates, server resources monitoring and backups are part of common system administration tasks. These operations are time consuming and the more servers to handle, the higher administrative expenses. The more administrative expenses are involved, virtualization can contribute to reducing the number of servers required for a particular workload and reducing administrative staff costs.

3.2 Major Components of Virtualization Environment

Virtualization is the way to create the physical machine's 'virtual version.' Using a virtual machine monitor, virtualization is achieved. It enables several virtual machines to operate on one single physical device. Without any changes observe in virtual machines it can easily be moved from hardware to another. In cloud computing, virtualization is widely used. Virtualization helps to run multiple operating systems and applications on the same hardware components on each of them.

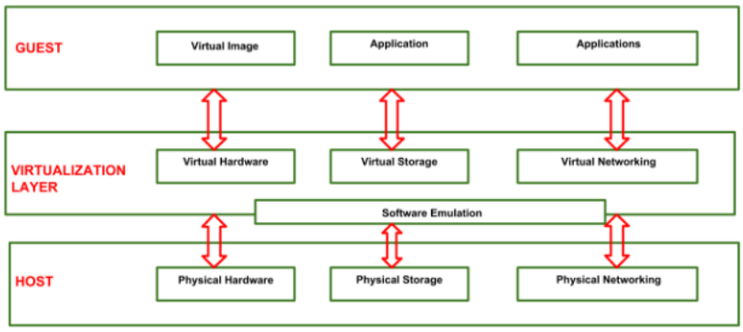


Figure: 3.1 Reference Model of Virtualization.
(Reference from "Mastering Cloud Computing Foundations and Applications Programming" by Rajkumar Buyya)

In a virtualized environment, three main components fall into this category:

1. GUEST:
As usual, the guest denotes the system component interacting with the virtualization layer instead with the host machine. Usually one or more virtual disk and VM definition files are presented to guests. A host application which looks and manages every virtual machine as a different application is centrally operated by virtual machines.

2. Hosts:
The host is the original environment in which the guest is to be managed. Each host uses the common resources that the host gives to each guest. The OS works as a host and manages the physical management of resources and the support of the device.

3. Virtualization Layer
The virtualization layer ensures that the same or different environment where the guest operates is recreated. It is an extra layer of abstract between the hardware, the computing and the application running in the network and storage. It usually helps to operate a single operating system per machine which, compared with virtualization, is very inflexible.

3.2.1 Characteristics of Virtualization

1. Increased Security –

The ability to fully transparently govern the execution of a guest program creates new opportunities for providing a safe, controlled execution environment. All guest programs operate usually against the virtual machine, translating them and using them for host program.
A virtual machine manager can govern and filter guest programs' activity so as to prevent harmful operations from being carried out. Resources exposed by the host can then be hidden or just protected against the guest.

Example-1: In Cuckoo sandboxes environment, untrusted code can be evaluated. In the term sandbox, the instructions may be filtered and blocked in the isolated execution environment before translating and executing in the actual execution environment.
Example 2: The Java Virtual Machine (JVM) expression sandboxed means a particular JVM configuration where instructions that are regarded as possibly harmful can be blocked through a security policy.

2. Execution Managed –

In particular, the most important features are sharing, aggregation, emulation and isolation.

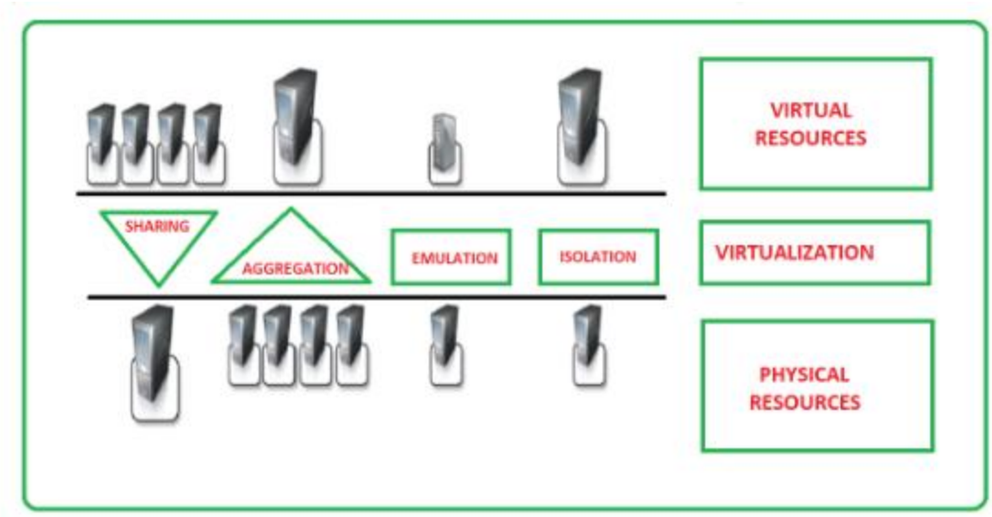


Figure: 3.2 Functions enabled by managed execution
(Reference from "Mastering Cloud Computing Foundations and Applications Programming" by Rajkumar Buyya)

3. **Sharing –**
   Virtualization makes it possible to create a separate computing environment in the same host. This common function reduces the amount of active servers and reduces energy consumption.

4. **Aggregation –**
   The physical resource can not only be shared between several guests, but virtualization also enables aggregation. A group of individual hosts can be linked and represented as a single virtual host. This functionality is implemented using the Cluster Management Software, which uses and represents the physical resources of a uniform group of machines.

5. **Emulation –**
   In the virtualization layer, which is essentially a program, guest programs are executed within an environment. An entirely different environment can also be emulated with regard to the host, so that guest programs that require certain features not present in the physical host can be carried out.

6. **Isolation –**
   Virtualization allows guests to provide an entirely separate environment in that they are executed — if they are operating systems, applications or other entities. The guest program operates through an abstraction layer that offers access to the underlying resources. The virtual machine is able to filter the guest's activities and prevent dangerous operations against the host.
   In addition to these features, performance tuning is another important feature enabled by virtualization. This feature is available a reality owing to the considerable progress in virtualization supporting software and hardware. By finely adjusting the properties of the resources exposed in the virtual environment, the guests' performance is easier to control. It offers a means to implement a quality of service (QoS) infrastructure effectively.

7. **Portability –**
   Dependent on a specific type of virtualization, the concept of portability applies in different ways**.**
   In the case of a hardware virtualization, the guest is packed in a virtual image which can be moved and executed safely on various virtual machines in many instances.
   With the virtualization of the programming level, as carried out in JVM or in. NET runtime, the binary code of the application components (jars or assemblies) may work on the respective virtual machine without recompilation**.**

3.3 Taxonomy of virtualization techniques

Virtualization encompasses a wide range of emulation techniques applied in various computing areas. A classification of such methods allows us to understand and use them.
The service or entity is discriminated against by first classification that is being emulated. Virtualization is used primarily to emulation in execution, storage and networking environments. The most oldest, popular and developed area of these categories is execution virtualization. It therefore needs further research and classification. We can especially divide the techniques of virtualization by examining the type of host they require in two main categories.

We can especially divide the techniques of virtualization by examining the type of host they require in two main categories.
Process-level techniques are implemented in addition to an existing operating system with full hardware control.
System levels technique are carried out directly on hardware and require no support from an existing operating system, or require limited support. In these two categories, we can outline different methods providing guests a different virtual computing environment: bare hardware, the resources of operating systems, low level programming language and the application libraries.

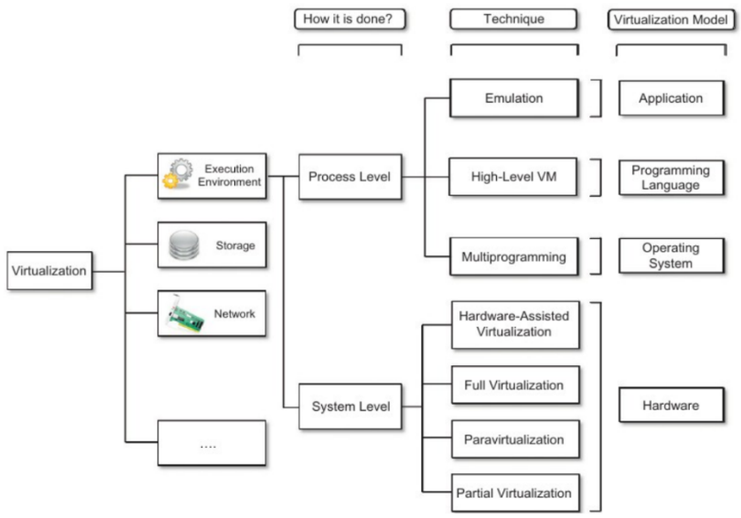


FIGURE 3.3 A taxonomy of virtualization techniques.
(Reference from "Mastering Cloud Computing Foundations and Applications Programming" by Rajkumar Buyya)

3.3.1 Execution virtualization

Execution Virtualization involves all the methods to imitate an execution environment that is separate from the virtualization layer host. All these techniques are focused on supporting program execution, whether it be the operating system, a binary program's specification compiled against the model or application of an abstract machine model. Therefore, the operating system, an application and libraries can directly or dynamically connected to the application image on top of the hardware**.**

3.3.2 Machine reference model

If execution environment is virtualized at levels other than the computation stack then a reference framework needs to be developed that defines the interfaces within the abstract level and this level of abstraction masks the details of the implementations.
This suggests that virtualization techniques can replace each layer and intercept the calls to it. For this reason a clear separation between the layers can simplify their

implementation, where only the interfaces need to be emulated and the subordinate layer is responded to.

On the base layer, the hardware model is declared or demonstrated according to an architecture, i.e. Instruction Set Architecture (ISA).

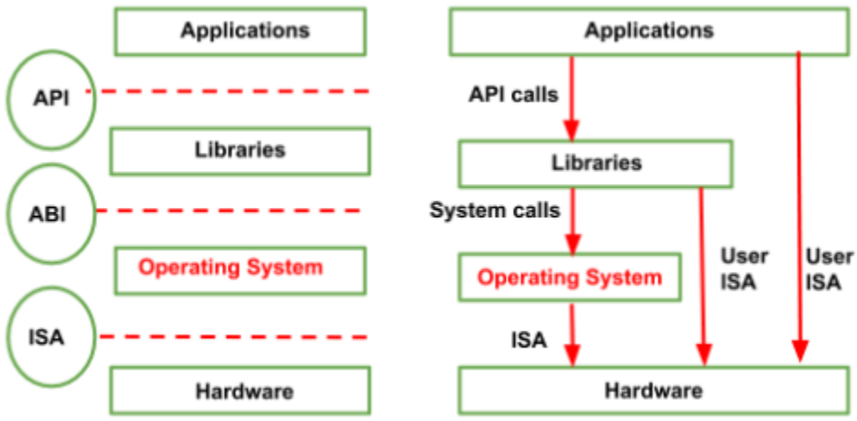


FIGURE 3.4 A machine reference model
(Reference from "Mastering Cloud Computing Foundations and Applications Programming" by Rajkumar Buyya)

3.3.2.1 Instruction Set Architecture (ISA)

The instruction set, known as ISA, is component of a computer which related to programming, which essentially is a machine's language. The instruction set provides the processor with instructions to tell it what to do. The set of instructions consists of addressing modes, instructions, native data types, registries, memory architecture, interruption and exception handling, and external I / O.

An example of the instruction set is the x86 instruction set, common on computers today. Throughout a still quite varying internal design, various computer processors can use almost the same set of instructions. Both processors Intel Pentium and AMD Athlon use almost the same x86 instruction set. An instruction set can be incorporated in the processor's hardware or emulated by an interpreter using software. The hardware design for running programs is more efficient and faster than the emulated program version.

3.3.2.2 Application Binary Interface

ABI is the Application Binary Interface. A Binary Code ABI defines how to invoke the functions, how parameters are passed between caller and callee, how return values are given to callers, how libraries are deployed and how programs are loaded into a memory. The linker thus applies an ABI: an ABI is the rules of how unrelated code works in conjunction. An ABI also governs the co-existence of processes on the same system. For example, an ABI could specify on the UNIX system how signals are executed, how a process invokes systems calls, what endianness is used and stacks are developed. An ABI is a set of rules that are implemented in a particular architecture by the operating system.

The kernel, toolchain and architecture troika define an ABI. It must be agreed by everybody on it. The architectures generally design a preferred or standardized ABI, and operating systems abide to that standardization more or less. Such information is usually documented in the reference manual for the architecture. For instance, x86-64,

3.4 Security Rings and Privileged Mode

The CPU operates mainly on two levels of privilege:

User Mode: Memory access is restricted in this mode to a certain extent whereas peripherals access is denied.
Kernel Mode: CPUs have instructions for managing and accessing memory in this mode and also have instructions for accessing peripherals such as disks and network cards. CPU switches automatically from one running program to another running program. This layered approach simplifies the expansions and applications of the computing system. This

layered approach simplifies the application of multi-tasking and coexistence of multiple executions.

The first one can be made in a privileged and unprivileged instructions. The instructions that can be used with interrupting with another task can be called the Non-Privileged Instruction. It is also called as it is not accessible by shared resources. Ex- contains all fixed points and floating and arithmetic instructions. Instructions that are executed under specific restrictions and that are commonly used for sensitive operations (expose behavior-sensitive or alter sensitive controls) are known as privileged instructions.

The OS manages the resources of a computer such as CPU processing time and memory access. Computers often run several software processes simultaneously, requiring different levels of access to resources and hardware.

Processes are performed in layered "rings" with different rights of access to resources at each ring. The central ring has the highest privileges and access is reduced in every subsequent layer. A common implementation of the x86 processor protection ring (a common CPU type) has four rings, from 0 to 3

There are two main advantages to the layered model. First of all, it protects from system crashes. Errors can usually be retrieved in higher rings (with less access). Because Ring 0 has direct access to the memory and CPU, it can be restarted without data loss or a CPU error in an outer ring crashing process. Secondly, it provides enhanced security. The procedure requires permission from the operating system to execute instructions that require greater access to resources. Then the OS can decide whether or not to grant the request. This selection process helps to prevent unwanted or malicious behavior of your system.

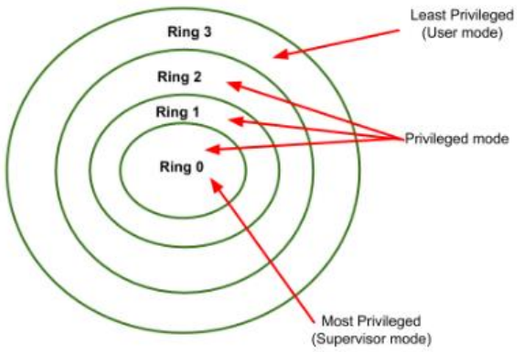


FIGURE 3.5 Security rings and privilege modes
(Reference from "Mastering Cloud Computing Foundations and Applications Programming" by Rajkumar Buyya)

3.4.1 Ring 0 (most privileged) and 3 (least privileged)

The kernel Ring 0 is accessible, which is a core component of most operating systems and can access everything, Code running in kernel mode is said to operate. Processes running in Kernel can significantly impact the whole system; if anything fails, a system shutdown will probably occur. This ring has direct access to the CPU and system memory, which means there are any instructions that require the use of either.

Ring 3, the least privileged ring, is available for user processes in user mode. This is the location for most applications that operate on your computer. This ring does not have direct access to the CPU or memory and must thus pass instructions to ring 0.

3.4.2 Rings 1 and 2

Special privileges exist for rings 1 and 2 that do not exist in ring 3 (user mode). Ring 1 is used to interact with your computer-connected hardware and control it. Playing a song via speakers or headphones or showing video on your monitor are examples of how to use this

ring. Ring 2 is used for instructions that interact with system storage, load or save files. These types of permissions are referred to as input and output because data is moved in or from a working memory (RAM). For example, it is in ring 2 to load a Word storage document. Document viewing and editing, the application layer, would fall under ring 3.

In a hypervisor environment, guest operating systems code is expected to run in the user to prevent the user from accessing OS status directly. When non-privileged instructions are implemented it is no longer possible to completely isolate the guest OS. The differentiation among user and supervisor mode enables us to understand the hypervisor's role. The hypervisor is conceptually running above the supervisor and the prefix hyper- is used. Hypervisors actually operate in a supervisor mode, and there are challenges in the design of virtual machine managers in the division between privileged and non-privileged instructions. All sensitive instructions are expected to be performed in a privileged mode that requires supervisor mode to prevent traps. Without this assumption, CPU status for guest operating systems cannot be fully emulated and managed. This is unfortunately not the case for the original ISA, which allows 17 sensitive user mode instructions. This prevents the separation and change of multiple operating systems managed by a single hypervisor system. Recent implementations of ISA (Intel VT, AMD Pacifica) have resolved this issue by revamping such instructions as privileged ones.

3.5 Hardware-level virtualization

Hardware-level virtualization is a virtualization technique that provides technique that enables abstract computer hardware execution environment where the guest operating system can be executed. The guest is defined in this model through the operating system and the host via the physical computer
The hardware, the emulation of the virtual machine and the hypervisor's virtual machine manager (see Figure 3.6). The hypervisor is usually a software / hardware program that enables the physical hardware underlying to be abstracted. Hardware level virtualization, which represents a system's hardware interface, is also referred to as system virtualization as ISA provides virtual machines. This means that the virtual machines that expose ABI to virtually different processes are distinguished.

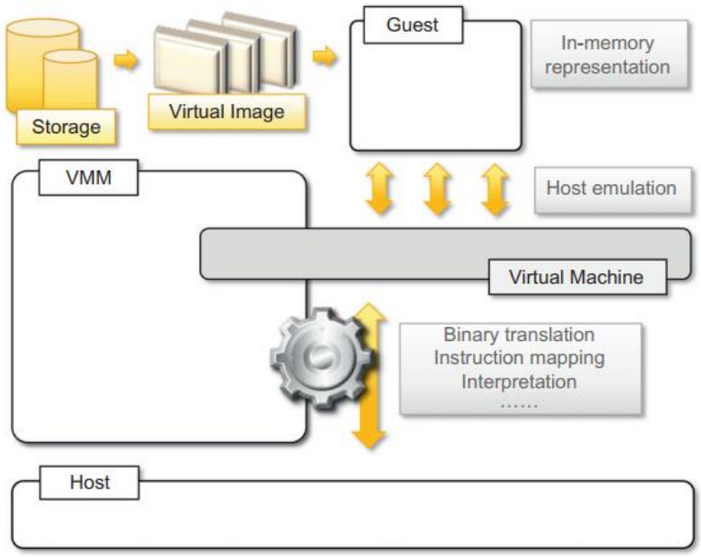


FIGURE 3.6 A hardware virtualization reference model.
(Reference from "Mastering Cloud Computing Foundations and Applications Programming" by Rajkumar Buyya)

3.6 Hypervisors

A hypervisor is a key software piece which enables virtualization. It abstracts from the actual hardware the guest machines and the operating system they use.
Hypervisors create the CPU / Processor, RAM and other physical resources virtualized layer that separates you from the virtual devices you are creating.

The hypervisor on which we install the machine is called a host machine, compared with virtual guest machines running over it. Hypervisors emulate resources available for guest machines to use. Regardless of which operating system you are booting with an actual hardware, it believes that real physical hardware is available. From the viewpoint of VM, the physical and virtual environment is unlike any difference. In the virtual environment, Guest machines do not know that the hypervisor has created them. Or share the computing power available. VMs run on the hardware that powers them simultaneously, and they are therefore fully dependent upon their stability operation.

- Type 1 Hypervisor (also called bare metal or native)
- Type 2 Hypervisor (also known as hosted hypervisors)

3.6.1 Type 1 Hypervisor

A bare-metal hypervisor (type 1) is a software layer which is installed directly above a physical server and its underlying hardware. Examples of Type 1 hypervisors include VMware ESXi, Citrix XenServer and Microsoft Hyper-V hypervisor.

There is no intermediate software or operating system, therefore bare-metal hypervisor is the name. A Type 1 hypervisor, which does not run inside Windows or any other operating system so it is proven to provide excellent performance and stability.

Type 1 hypervisors are a very basic OS themselves, on which virtual machines can be operated. The hypervisor's physical machine is used for server virtualization purpose only. For anything else, you can't use it. In enterprise environments, type 1 hypervisors are mostly found.

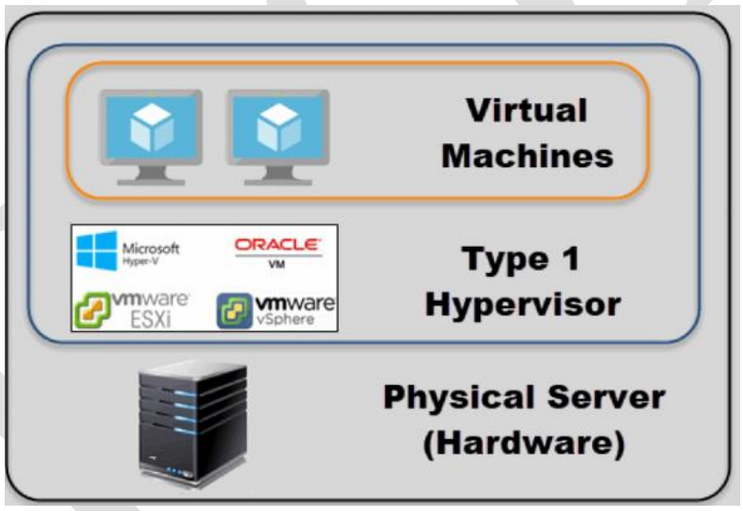


FIGURE 3.7 Type 1 Hypervisor

3.6.2 Type 2 Hypervisor

This type of hypervisor runs within a physical host operating system. .Example of Type 2 hypervisor include VMware Player or Parallels Desktop.

That is why we call type 2 hypervisors – hosted hypervisors. In contrast to type 1 hypervisors, which run directly on the hardware, hosted hypervisors have one underlying layer of the software. Here we have the following:

- A physical machine.
- An installed hardware operating system (Windows, Linux, macOS).
- Software for the type 2 hypervisor in this operating system.
- The current instances of virtual guest machines.

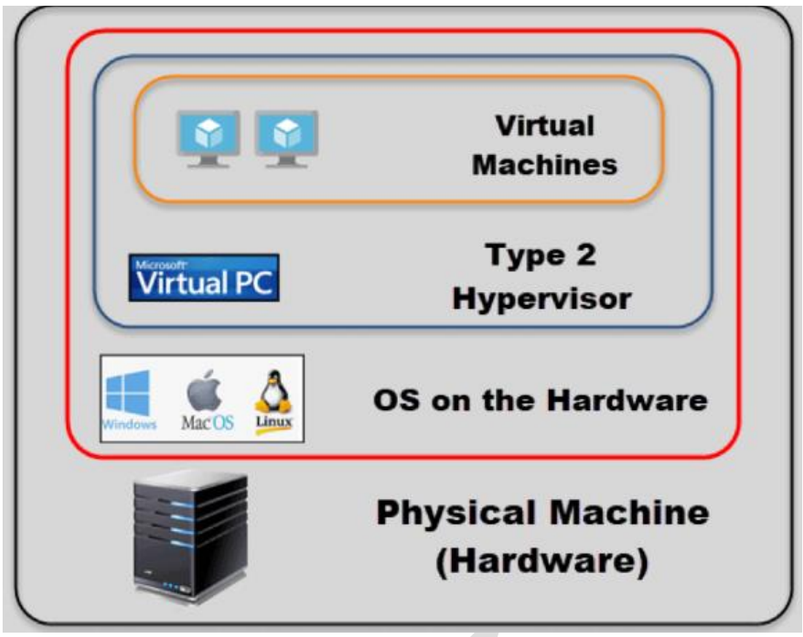


FIGURE 3.8 Type 2 Hypervisor

3.6.3 Choosing the right hypervisor

Type 1 hypervisors offer much better performance than Type 2 Those are the logical choice for mission-critical applications and workloads because there is no middle layer. However, that's not saying the hosted hypervisors that are hosting do not have their place. They're much easier to set up, so it's a good bet if, say, you're going to have to quickly implement an environment of the test. . One of the best ways to find out which hypervisor meets your needs is to compare their performance metrics. The following factors must be examined before selecting the appropriate hypervisor: These include CPU overhead, maximum host and guest memory, and support for virtual processors

1. **Understand your needs:** the data center (and your job) for the company and its applications. In addition to the requirements of your company, you (and your IT staff) also have your own requirements.

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

2. **The cost of a hypervisor:** For many buyers, a hypervisor is the most difficult thing in selecting the right balance between cost and functionality. While some entry-level solutions are free or practically free, prices can be staggering at the opposite end of the market. The frameworks for licensing vary too, so it is important to know what your money gets precisely.

3. **Virtual machine performance:** Virtual systems should achieve or exceed, in relation at least, to the server applications, their physical counterparts' performance. All that goes beyond this benchmark is profit.

4. **Ecosystem:** The role that an ecosystems hypervisor can have in deciding on whether a solution is cost-effective or not is tempting to ignore – that is, availability of documentation, support, training , development and consultancy services.

5. **Test for yourself:** You can obtain basic experience from your existing desktop or laptop. To build a nice virtual learning and testing environment, you can run VMware vSphere and Microsoft Hyper-V in either VMware Workstation or VMware Fusion.

3.6.7 Hypervisor Reference Model

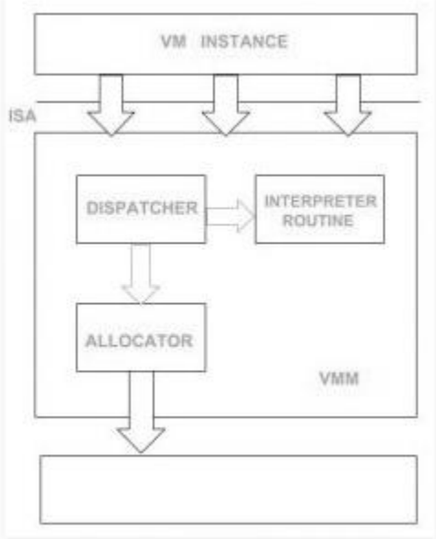


FIGURE 3.9 A hypervisor reference architecture.
(Reference from "Mastering Cloud Computing Foundations and Applications Programming" by Rajkumar Buyya)

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

1. Dispatcher
2. Allocator
3. Interpreter

Dispatcher:
The dispatcher acts as the monitor entry point, rerouting virtual machine instance instructions to one of the other two modules.

ALLOCATOR:
The allocator is responsible for deciding the system resources to be given to the virtual machine instance. It indicates that the dispatcher invokes the allocator whenever virtual machine attempts to execute instructions that modify the machine resources associated with the virtual machine.

Interpreter:
The interpreter module consists of routines. These are executed while virtual machine executes a privileged instruction.

The requirements for Popek and Goldberg are a number of conditions which are sufficient for a computer architecture to effectively support system virtualization. In their 1974 article entitled "Formal Requirements for Virtualizable Third Generation Architectures" introduced by Gerald J. Popek and Robert P. Goldberg. Although simplifying assumptions are taken into account, the requirements remain a useful way in which to determine whether a computer architecture supports efficient virtualization and provides guidelines to design virtualized computer architectures.

Virtual system machines can virtualize an entire range of hardware resources, such as processors, memory and storage resources and peripheral devices. A virtual machine monitor is a software component which provides the abstraction of a virtual machine, also known as a hypervisor. In analyzing the environment created by VMM there are three properties are follows:-

Equivalence / Fidelity
Under the VMM a program is running should behave essentially the same as when running directly on an equivalent machine.

Resource control / Safety
The virtualized resources must be fully controlled by VMM.

Efficiency / Performance
Without VMM intervention, a statistically dominant fraction of machine instructions must be performed.

A VMM must contain all three properties in the terminology of Popek and Goldberg. VMMs are typically assumed to fulfill the equivalence and resource control properties, which are also known as effective VMMs. The characteristics that Popek and Goldberg must have in order to perform VMMs that have the above properties, are described in the instruction set architecture (ISA). This model comprises a system or user mode processor which has access to linear, consistently addressable memory. A subset of the instruction set is assumed to be only available in system mode and the memory related to the relocation register is addressed. Interrupts and I / O are not modeled.

Popek and Goldberg classify the ISA instructions into three different groups in order to derive their theorems of virtualization, which give sufficient (but not necessary) virtualization conditions:

Privileged instructions
Those that trap in user mode when the processor is in system mode do not trap (Supervisor mode).

Control sensitive instructions
Those who try to change the system resource configuration.

Behavior sensitive instructions
The behavior of those whose results depend on the resource's configuration (relocation's registry content or processor mode).

This can then be the main result of the analysis by Popek and Goldberg.

Theorem 1. An effective VMM can be built on any conventional third-generation computer if the sensitive set of instructions is a subset of the privileged instructions for that computer.

The theorem states intuitively that all instructions that could affect VMM (sensitive instructions) to correctly function always trap and transfer the control to VMM are sufficient for the building of a VMM. This ensures the property of the resource control. Instead, native (i.e. efficiently) non-privileged instructs should be executed. It is also necessary to keep the equivalency property.

This theorem also provides a simple method for VMM, known more recently as a classical virtualization called trap-and-emulate virtualization: all the VMM's sensitive instructions have to do is trap and emulate each of them.

A related problem is that sufficient conditions are obtained for recursive virtualization, i.e. conditions under which a VMM can be created that can work with a copy of itself. The following (sufficient) conditions are presented by Popek and Goldberg.

Theorem 2. A conventional third-generation computer is recursively Virtualizable if:

- **It is Virtualizable and**
- **A VMM without any timing dependencies can be constructed for it.**

There are architectures that do not meet those conditions, such as the non-hardware-assisted x86, therefore they cannot be virtualized as usual. However, architectures (in x86

case, CPU / MMU level) can still be fully virtualized using various techniques such as binary translation, which replaces sensitive instructions that do not generate traps sometimes referred to as critical instructions. However, this extra processing reduces the theoretical efficiency of the VMM, while also hardware traps are cost-effective. Comparable efficiency can be achieved in a smoothly tuned binary translation system, which only allows sensitive instructions to be trapped in relation to first-generation x86 hardware assist.

Theorem 3. A hybrid VMM may be constructed for any third generation machine in which the set of user sensitive instructions are a subset of the set of privileged instructions:

A third-generation hybrid VMM may be built on which the user-sensitive instructions are part of the privileged instructions set.
– More instructions will be interpreted in HVM instead of directly executed.

3.7 Hardware virtualization techniques

Hardware-assisted virtualization, the first virtual machine operating system (VM/370 in 1972), was introduced on the IBM System/370. In the latter 70's, Virtualization was forgotten, but the development of x86 servers has re-enlightened the interest in virtualizing driven for a server consolidation requirement. Virtualization allowed a single server to replace multiple underutilized dedicated servers.

The x86 architecture, however, did not meet the criteria of Goldberg and Popek for "classical virtualization." In order to compensate for these limitations: Virtualization of an x86 architecture was carried out by two methods: full virtualization or paravirtualization. The illusion of physical hardware is created to achieve the objective to independently manage the operating system from hardware, but to achieve some performance and complexity.

Intel and AMD introduced new technologies for virtualization, a number of new instructions, and – most importantly – a new level of privilege. The hypervisor is now present at "Ring -1" so that the guest operating system can operate at ring 0.

Virtualization of hardware leverages virtualization functionality incorporated into the latest generation of Intel and AMD CPUs. These technologies, respectively called Intel VT and AMD-V, offer enhancements needed to run non - modified virtual machines without the overhead of the full CPU virtualization emulation. These new processors include an additional privilege mode, below ring 0, in which the hypervisor essentially can operate leaving ring 0 for unmodified guest operating systems.

The VMM can efficiently virtualize the entire X86 instruction with hardware-assisted virtualization using the classically-used hardware trap-and-emulate model, rather than software, by handling these sensitive instructions. CPU access can be accessible at Ring 1 and guest OSes by hypervisors that support this technology, the same way as they would when operating on a physical host. This makes it possible to virtualize guest OSes without any changes.

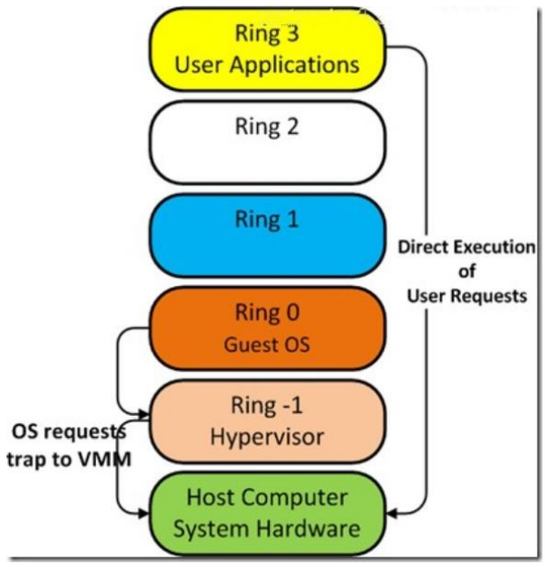


FIGURE 3.10 New level of privilege in x86 architecture

3.7.1 Advantages of Hardware-Assisted Virtualization

Hardware-assisted virtualization changes the operating system access. Operating systems of x86 have direct access to running system resources. VMM emulates the necessary hardware into the operating system with software virtualization. The operating system provides direct access to resources without an emulation or modification with hardware-assisted virtualization, and this improves overall performance.

This implies that OS kernels need not be tweaked and can run as is (as in par virtualization). The hypervisor does not have to take part in the inefficient binary translation of the sensitive instructions at the same time. Thus, it not only complies with the Popek and Goldberg criteria (of full virtualization), but also improves its efficiency, because the instructions are now trapped and emulated directly in the hardware.

3.8 Full virtualization

Full virtualization is a technique for the virtualization of a VME that simulates the underlying hardware completely. Any software that can run on physical hardware can be run in this type of environment in the VM, and any OS supported by the underlying hardware can be run with each VM. Users can simultaneously run several different guest OSes. The VM simulates sufficient hardware for unmodified Guest OS to run in isolation in full virtualization software. In a number of situations, this is particularly helpful. Experimental new code, for example, can run in an OS development in a separate VM simultaneously with older versions. The Hypervisor delivers every VM, including a virtual BIOS, virtual devices and virtualized memory management, all services of the physical system. The Guest OS is completely unconnected with the virtualization layer from the underlying hardware.

Full virtualization is achieved through the use of binary and direct execution combinations. The physical CPU executes at native speed nonsensitive instructions with full Virtualization Hypervisors, translates the OS instruction and is cached for future use, and the instructions at user level are executed at native speed with no change. Full virtualization provides optimal isolation and security for VMs, making migration and portability easier as virtualized and native hardware are used by the same guest OS instance. The concept of complete virtualization is shown in Figure

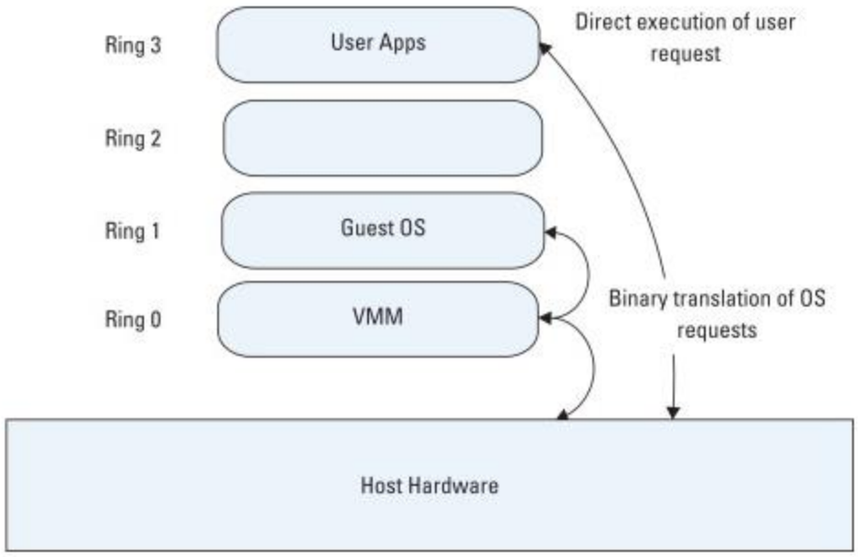


FIGURE 3.11 Full Virtualization

3.9 Paravirtualization

Paravirtualization is another approach to server visualization whereby paravirtualization is a thin layer that does not imitate a complete hardware environment; it makes sure that all guest systems share their system resources and work around each other well. The "Para" is an English affix of Greek origin which means "beside," "with," or "alongside."

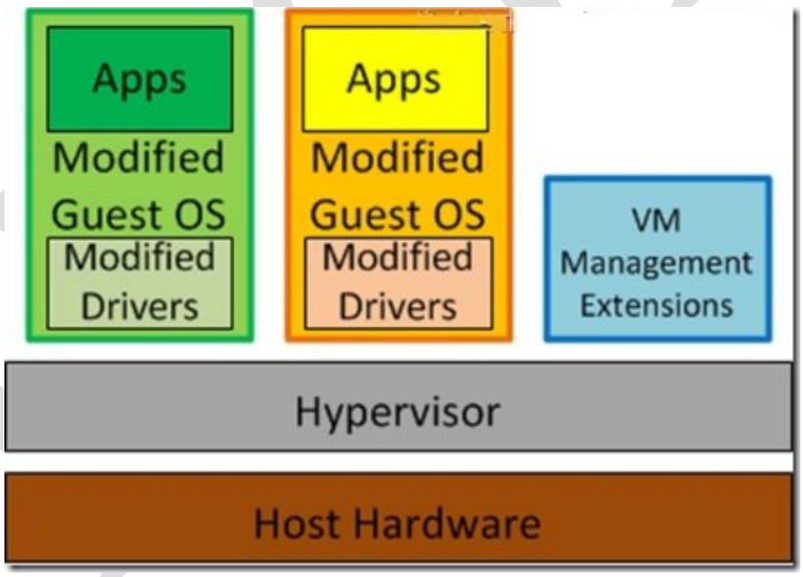


FIGURE 3.12 Paravirtualization

The Guest Operating System kernel will be altered to run on the hypervisor under the paravirtualization. This usually requires a replacement in the ring 0 of the CPU of privileged operations by calling a hypervisor (called hypercalls). The hypervisor, in turn, performs the task on behalf of the guest kernel and offers hypercall interfaces to other crucial kernel operations like memory management, interrupt handling and time keeping.

Paravirtualisation attempts to correct all problems related to virtualization by allowing the guest operating systems to directly access the subordinate hardware and thus to improve communication between the Guest OS and the hypervisor. Because it contains OS modifications, paravirtualization is sometimes called OS-Assisted Virtualization as well.

Paravirtualization, in which the guest OS "knows" how it is virtualized, is different from the full virtualization, in which the unmodified OS does not know that it is virtualized and sensitive OS calls trapped by binary translation.

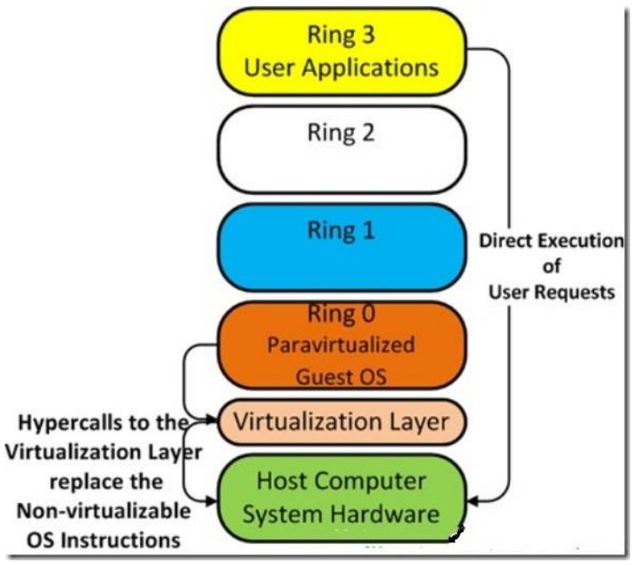


FIGURE 3.13 Hypercalls to virtualization in Paravirtualization

Paravirtualization Advantages

This approach has two advantages:

The guest kernel's ability to communicate with the hypervisor directly leads to greater levels of performance. You will recall that a complete virtualization inserts a complete layer of the hardware emulation between the guest OS and the physical hardware. The thin software layer of paravirtualization acts more like a virtualized server such as air traffic controller, which gives a guest OS access to the hardware physical resources while all other guest OSs stop simultaneously accessing the same resources. The value offering of paravirtualisation is a lower overhead virtualization, but the performance advantage over full virtualization depends on the workload; this method generally is much more efficient than conventional hardware emulation virtualizations;

The second advantage of the paravirtualization approach in comparison to full virtualization is that paravirtualization does not confine you to device drivers included in the virtualization software. It uses instead the device drivers, known as the privileged guest, in one of our guest operating systems. If you don't get too much into this architecture here, you just have to say that this is an advantage since it gives organizations the opportunity to benefit from all the hardware capabilities of the server, instead of being limited to the hardware for which drivers are available as a whole in virtualization programs.

Paravirtualization Limitations

In paravirtualization, the guest operating systems must be altered to interact with the paravirtualization interfaces. This usually limits support for open source operating systems such as Linux, which can be openly modified and proprietary operating systems, where the owners agree to make codes for a specific hypervisor. Since paravirtualization cannot support unmodified OS (e.g. Windows family) it is not compatible and portable;

Paravirtualization can also introduce major production-based support and maintenance issues because deep OS kernel amendments are needed.

3.9.1.2 Partial virtualization

In computer science, partial virtualization is a virtualization technique that has been employed to implement a virtual machine environment: one providing a "partial simulation of the underlying hardware." Most, though not all, of the hardware functionalities are simulated which results in virtual machines that can operate certain or all software without modification. In general, it means that the whole operating systems "could not," but that many of the applications can run, run on the virtual machine. This is a sign of full virtualization.

The ‘address space virtualization,' in that each virtual machine comprises of a distinct address space, is the key to partial virtualization. This capability necessitates relocation hardware and has adapted partial virtualization in other practical examples.
A major historical landmark on the path to full virtualization was partial virtualization. It has been used in the time-sharing CTSS system of first generation and in the experimental paging system was IBM M44/44X. The concept can be used to define any operating system with separate address spaces for independent users or processes, which include numerous that currently do not meet the criteria as virtual machine systems,. The experience and limitations of partial virtualization have led to the first full virtualization system

Partial virtualization is much easier than full virtualization. It has frequently provided useful, strong virtual machines that support major applications. Its drawback is in situations where backward compatibility or portability is needed (in contrast with full virtualization). When certain hardware features are not simulated, all software using such features is unsuccessful. In addition, the features used for a particular application can be difficult to predict precisely.

Partial virtualization has proven extremely successful for multiple users to share computer resources.

3.9.2 Operating system-level virtualization

Operating system level virtualization (OS virtualization) is a server virtualization technology that includes altering the operating system to allow different applications to be run simultaneously on one computer at one time by different users. Although they operate on the same computer, virtual operating systems do not interfere with each other independently. The standard OS is altered and adapted for operation of independent systems. This virtual system is designed to comply with the user's commands that can run various applications on the machine simultaneously. The virtual operating system processes each user request individually. An advantage of operating system level virtualization is that the availability of applications will have a minimal impact even during system upgrades and security patches. Virtualization of the operating system allows vital applications to be moved into other virtual operating systems so that performance can continues.

A methodology in which the kernel of an operating system enables for several isolated user-space instances uses in this kind of server virtualization. The instances are running on top of the previous host operating system and feature a set of libraries with which applications interact, illustrating how they run on a machine dedicated to their use. Containers, virtual private servers or virtual environments are known as instances.

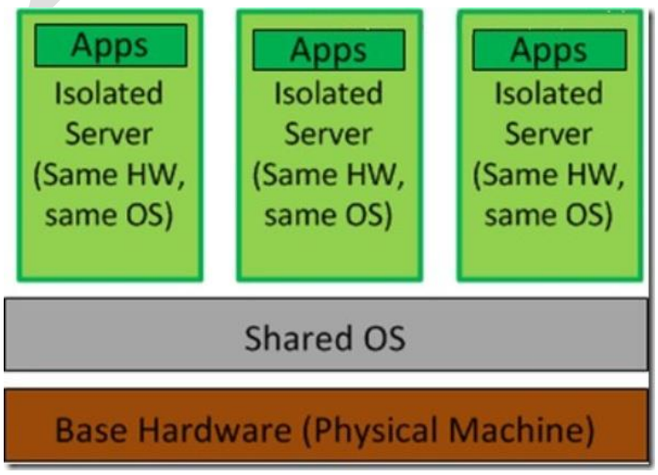


FIGURE 3.14 Operating system-level virtualization

The host system with a single OS kernel and its control of the guest functionality of the operating system virtualizing system level is achieved. In this virtualization of a shared kernel, the Virtual guest systems each have a root file system of their own.

Host Virtualization In which the hypervisor (the container) has a very limited functionality which is depend upon Host OS for CPU scheduling and memory management This method, which uses OS-level virtualization, does not even include the application of a real hypervisor, but is a component of the operating system that performs all of the hypervisor 's tasks.

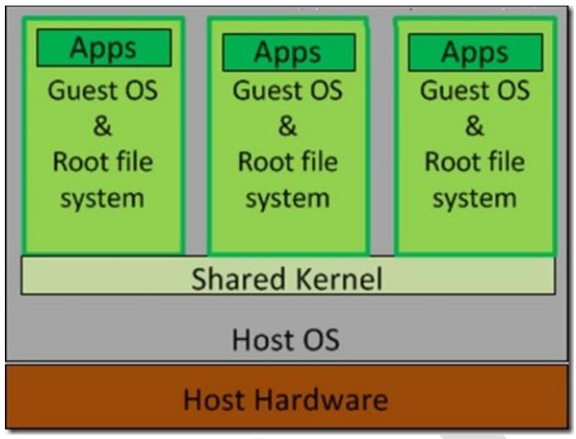


FIGURE 3.15 Operating system-level virtualization 2

This means that OS virtualization is relied on the creation on a single physical server of isolated containers or partitions including the use of OS instances to operate individually from the other partitions, in each guest application environment. This technique installs the software layer for virtualization on the operating system and the system for all guests operates on this layer using the same operating system as the host operating system, but each guest has its own resources and runs in complete isolation from the guests.

It is arguable that this is not in strict sense virtualization; rather it is a technique which only enables the consolidation of the machine.

3.10 Programming language-level virtualization

3.10.1 Application-level virtualization

Application virtualization is a mechanism that tricks a standardized application into believing it interacts directly with the functionality of an operating system whereas, in reality, it would not.
That requires a layer of virtualization inserted between the application and the OS. This layer, or system, needs to run the subsets of an app virtually without any affecting the underlying OS. The virtualization layer substitutes a part of the runtime environment typically provided by the OS, transparently diverting files to a single executable file, and changes in the registry log.

Through diverting the processes of the app into one file rather than several scattered around the OS, the app runs easily on another device, and apps that were previously incompatible may now operate adjacently.
Desktop virtualization is used in conjunction with application virtualization — the separation from the end-user system that accesses the physical desktop environment and its related app software.

Benefits of Application Virtualization

- Enables legacy apps (e.g. OS platforms such as Windows 7 or XP who development is end) can be run.
- It allows cross-platform operations (e.g., running iOS, Android, macOS and Chrome OS applications).
- Prevents conflicts with other virtualized application
- Allows users to operate multiple app instances — unless they are virtualized, several applications can detect and not allow new instances to runs

Limitation of Application Virtualization

- This is difficult to virtualize all computer programs. Applications which require a system driver (a type of OS integration) and 16-bit applications to run in shared memory space are some of the examples.
- Anti-virus software and programs that need high OS integration are difficult to virtualize, such as WindowBlinds or StyleXP.
- Application virtualization is subject to major license flaws in software licensing, particularly because it must correctly license both the application virtualization software and virtualized applications.
- Whilst application virtualization can fix issues between old applications and newer operating systems in terms of file-and-registry compatibility, applications that don't handle the heap properly won't work with Windows Vista because it does still assign memory, no matter how virtualized. Therefore, even when the program is virtualized, specific application compatibility fixes (shims) may be required.

3.11 Other types of virtualization

Many forms of virtualization have an abstract environment for other than virtualization interact with them. They include storage, networking and interaction between client and server.

3.11.1 Storage virtualization

Storage is another part virtualization computing concept. The definition of Storage virtualization is Storage virtualization refers to the method of physical storage abstraction. While the RAID functionality at the base level, the word storage virtualization typically involves additional concepts such as data migration and caching. Storage virtualization is difficult to describe in a specific way as it is possible to have a variety of different functionalities. It is typically provided as a function of:

- Host Based with Special Device Drivers
- Array Controllers
- Network Switches
- Stand Alone Network Appliances

In this respect, each vendor has a different approach. The primary way of classifying storage virtualization is whether in-band or out-of-band. In-band (often called symmetric) between the host and the storage device permits caching. Virtualizing out-of-band (often called asymmetrical) uses host-based drivers that first check at the metadata (indicating the location of the file) and then enable the host to access the file directly from the stored location. This method does not require caching at the virtualization level.

General benefits of storage virtualization include:

- **Migration** – Data can quickly be transferred through storage locations without interrupting the live access of most technologies to the virtual partition.
- **Utilization** – The usage of storage devices can be managed for addressing over and over using in the same way as server virtualization.
- **Management** – Most hosts may use storage to centrally manage a physical device.

Some of the disadvantages include:

- **Lack of Standards and Interoperability**– Storage virtualization is a term, not a standard. This also means the vendors don't interoperate easily.
- **Metadata** – The storage metadata and management are essential to a functioning reliable system as there is a correlation between logical and physical location.
- **Backout** – Mapping the backout of virtualized infrastructure from the network from digital to physical locations is often less than trivial.

3.11.2 Network Virtualization

In the field of computing, Network virtualization integrates hardware, software and network infrastructure with network functionalities into a single virtual network administrative entity. Virtualization of networks requires platform virtualization, often coupled with virtualization of resources. Network virtualization is defined as either outside or merged into a virtual unit or internal, which gives the software containers a network-like functionality on a single system.

With the internal definition of the phrase, virtualization of the desktop and server offers networking connectivity both for the host and guest and for several guests. Virtual switches are recognized as part of a virtualization stack on the server side. Nonetheless, the external concept of network virtualization is likely the most used version. Virtual private networks (VPNs), with most enterprises supporting VPNs, have been standard components in the Toolbox for years. A further widely used definition of network virtualization is virtual LANs (VLANs). The networks need to be organized solely along regional lines with network developments as 10 gigabit Ethernet is not a long time.

In general benefits of network virtualization include:

Customization of Access – administrators can easily customize access and network options, including bandwidth throttling and quality of service.
Consolidation – Virtual networks can be merged into one virtual network to simplify management overall.

Like server virtualization, network virtualization will add complexity, contribute to overhead performance and the need for administrators to have a greater degree of ability.

3.11.3 Desktop virtualization

Desktop virtualization is a software paradigm that takes the conventional thin-client model from the cloud, but is designed to provide the world's best to administrators and end users: hosting and central management in the data center of virtual machines while giving end-users the fullest PC desktop experience.

Hosted application Virtualization is identical to the hosted desktop virtualization, which extends user experience to the whole desktop. Microsoft's Terminal Services, Citrix's XenDesktop and VMware's VDI are all commercial products.

Advantages of Desktop virtualization cover the majority of those with virtualization applications as well as:

- **High availabilit**y – Downtime with network replication and fault tolerant settings can be reduced.
- **Extended Refresh Cycles** – Larger capacity servers and fewer requirements on the client PCs will increase their lifespan.
- **Multiple Desktops** – Users can control multiple desktops from the same client PC, which can handle different tasks.

Desktop virtualization also has drawbacks are close to the server virtualization. The additional downside is that consumers need to be networked to access their virtual desktops. For offline jobs, this is troublesome as well as increasing network demand in the workplace.

3.11.4 Application server virtualization

Application server virtualization describes a selection by means of load balancing techniques and a high-value architecture for the services hosted in the application server that offer the same services as a single virtual application server. This is a special type of virtualization that aims to make virtual storage more efficient than to emulate another environment. It is a different virtualization.

3.12 Virtualization and cloud computing

Cloud computing, which is flexible, scalable and often reduces in the cost and complication of applications, is today one of the biggest sounding and exciting technologies. Virtualization is the primary technology for the use of cloud computing. Virtualization is a component of cloud computing. Based on cloud virtualization, workloads can be quickly deployed and scaled through the rapid provisioning of virtual and physical machines. Clouds are seen as a pool of virtualized resources that are easily used and accessible. Software as a service model, Platform as a services model and Infrastructure as a service model are three cloud service models. Software is a service model for the development of services that guarantees consumers are paying for not owning apps that is using. Platform as a service model gives users a platform for creating and configuring their software accordingly. Infrastructure as a service model is the self-managed model for controlling and monitoring remote data.

Cloud provider administrates networking, storage and computing services. Cloud computing basically offers access to resources required to carry out various activities with increasing user demands. The concept behind cloud computing is to enable businesses to increase their computer hardware's performance, use of resources and flexibility. Virtualization is the most relevant technology today. For cloud computing, virtualization plays a key role as it allows a degree of customization, protection, isolation and manageability necessary for on-demand service delivery.

The method of consolidation hereby revealed assigned VMs to a physical server and uses parameters such as minimal (min) the required amount of Resource by each VM and maximal resource permissible (max) by each VM application to decide the amount of the VM resource. These virtualization parameters are important frameworks to guarantee an intelligent distribution of resources among various applications (especially when a diverse range of VM applications are available in various preferences and resources). The method of consolidation is aimed at placing VM's in order to position the application VM's combinations on each physical server, taking into account resource allocation for each VM, based on various priorities and affinities of resources. The effectiveness of a unified program has a beneficial effect on this granularity in the distribution of resources. This process is often called server consolidation, while virtual machine migration is the transfer of virtual machine instances

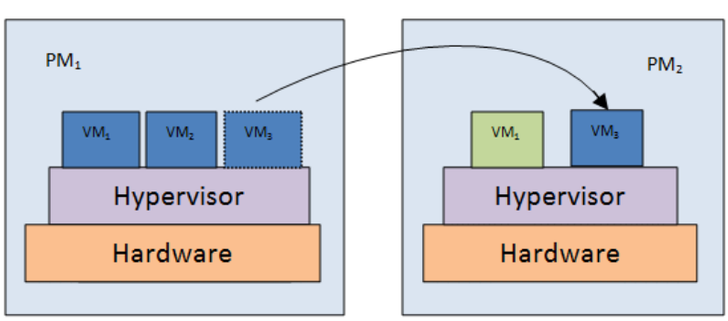


FIGURE 3.16 virtual machine migration

Live Migration moves a running VM from a physical server to a different one without interrupting the virtual machine's availability to the users. Live Migration. The aim of a VM live migration is to allow maintenance or updates without interruption on a virtual machine during migration on a VM. Often known as seamless live migration, when the end user does not discover downtime during the migration cycle.

3.12.1 Pros and cons of virtualization

Most companies are keen to upgrade their solutions on virtual machines with the rise of virtualization growing. Nonetheless, it is important to consider the advantages and disadvantages of virtualization before any improvements are made. The benefits and drawbacks of both physical and virtual systems, each of which has time and location.

Virtual technology has obvious advantages, however there are a few drawbacks. To order to decide how it best matches the company requirements we have summarized the pros and cons of virtualization.

3.12.1.1Advantages of virtualization

Scalability
A virtual machine is merely as scalable as any other solution. One of the key benefits of virtualization is that several systems can be integrated. It offers you unbelievable flexibility that with a physical and bare metal system which cannot be possible. This flexibility has a direct impact on how companies can grow quickly and efficiently. Virtualization allows data migration, upgrade, and instant performance improvement into new VMs in a short time.

Consolidation of servers

The design of virtual machines will replace almost 10:1 the physical machines. This eliminates the need for physical computers while providing efficient operation of systems and specifications. Such consolidation will minimize costs and the requisite physical space for computer systems.

Improved System Reliability
One of the reasons for virtualization is its ability to help avoid failures in the system. Memory corruption caused by system drivers and the like is the most common crashes preventing a VM. Such systems describe the DMA architecture to improve I / O isolation. It offers improved security and reliability.

Virtual WorkStations
Virtualization provides the global versatility to allow multiple systems to be run on a single computer to operate the systems remotely. VM also reduces all hardware and desktop footprint.

3.12.1.2 Disadvantages of virtualization

Programs that Require Physical Hardware
Virtualization does not work well for any applications that require physical hardware. An example is something using a dongle or other hardware attached. Since the program needs to be a physical piece, virtualization would cause more headache than remaining on a physical system.

Performance Quality Can Decrease
If you run an application which runs RAM or CPU use, virtualization can cause a performance delay. A VM operates in layers on its hosting systems so that any operation with extreme performance will have reduced performance if you do not use one application or server. The downside of virtualization is that many applications can run on small physical servers, so it is difficult to spend a single host on one server.

Testing is Critical
The goal of IT is really to achieve your company objectives so that you won't have an untested software platform for your business. This is particularly true for virtualization, as it doesn't work as if you can turn it off and back on. A system that works smoothly already can perform virtualization, leading to errors and potential waste of time and expense. Before switching to VM, always check.

Unexpected Expenses
Initially, it might seem that virtualization saves you a little money. But this is a process which needs to be completed and done correctly for the first time. You will spend more than initially planned in order to take account of this attention to time and detail. Before taking the plunge, please review the tools and management systems you may need to help you transition to a virtual machine.

Data can be at Risk

Your data is hosting on a third-party network while operating on virtual instances on shared hardware resources. This can vulnerability the data to threats or unauthorized access. It is a problem if the security solution of your service provider does not secure your virtual instance and data. Especially for virtualization of storage is true.

Quick Scalability is a Challenge

Scaling on Virtualization is a limited time, so it doesn't matter if it's to be achieved in such a short time. With physical set-up, new equipment and scale, even though it entails some initial problems, can be built easily. Virtualization can be a tedious task to ensure all necessary software, protection, adequate storage, and availability of resources. This takes longer than you might plan as a third-party vendor is involved. However, further management problems are also the additional costs involved in increasing the usage of cost.
Unintended Server Sprawl

Unintentionally extending the server sprawl is a big problem for both administrators and users alike. Some of the issues posed by people at the service desk are application extensions. Installation of a physical server takes time and resources, while in a matter of minutes a virtual server can be built. Users build new servers every time rather than re-using the same virtual server, as it allows them to start again. The server administrator who has five or six servers has 20 virtual servers to handle. This could lead to a big difficulty in smooth operations and the forced end of some servers could also result in data loss.

3.13 Technology examples

3.13.1 Xen: paravirtualization

Xen is a paravirtualized open source hypervisor. Paravirtualization is the most popular application. Xen has been expanded with hardware-assisted virtualization to be fully compliant. It allows for high efficiency in the guest operating system. That would likely be accomplished by eliminating the performance loss when executing the instructions requiring considerable supervision and changing part of Xen's guest operating system with regard to the performance of these instructions. This especially supports x86, the most common architecture on commodity and server machines.

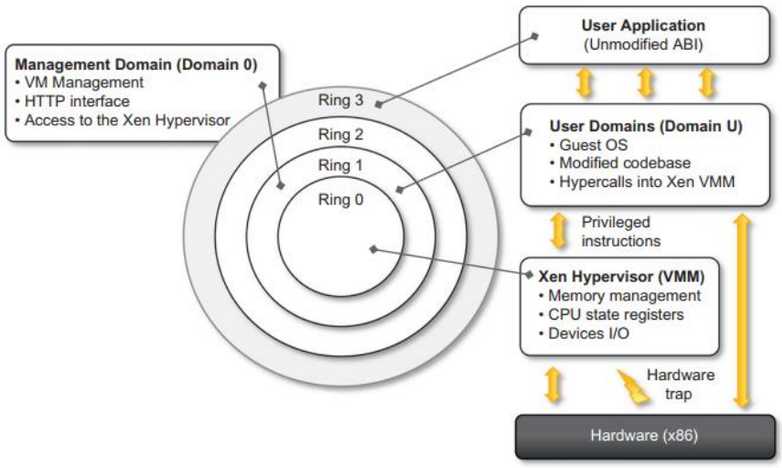


FIGURE 3.16 Xen architecture and guest OS management.

Figure above defines the Xen and its mapping to a classic x86 paradigm of privilege. Xen hypervisor is operating a Xen-based system that is operating in the most comfortable mode and retains the guest operating system's access to essential hardware. Guest operating system that runs between domains, representing instances of virtual machines.
However, a certain control program with privileged host access and management of all other guest operating systems operates on a specific domain called Domain 0. This is the onl y loaded one after a fully-booted virtual machine manager hosts an HTTP server which provides requests to build, configure and terminate virtual machine. The component provides the first version of an IaaS (Infrastructure-as-a-Service) solution, shared virtual machine manager (VMM). The program is essential for cloud-based computers.

Different x86 implementations allow four different safety levels, called rings, i.e., Ring 0, Ring 1, Ring 2, and Ring 3.

Here, Ring 0 is the most privileged level and Ring 3 is the less privileged level. Nearly every OS, except OS/2, uses only two different levels, i.e. Ring 3, for user program and non-privilege OS, Ring 0 for kernel code, and. It gives the Xen an opportunity to achieve paravirtualization. This makes it possible to manage the Application Binary Interface (ABI) unchanged and thus to switch from an application point of view to xen-virtualized solutions.

The structure of the set of instructions x86 enables the execution of code in the Ring 3 to move to ring 0 (kernel-mode). Such an operation is performed at the hardware level, and thus, it can lead to TRAP or a silent fault in a virtualized system, thus preventing the overall operation of the guest OS in ring 1.

In theory, this condition exists via a subset of system calls. Implementing the operating system needs a modification, and all of the critical system calls need re-implementation by hypercalls to eradicate this situation. Here, hypercalls are the special calls exposed via the Xen Virtual Machine (VM) interface, and Xen's hypervisor appears to obtain, manage and return the control with the aid of the supplied handler to the Guest OS.

Paravirtualization calls for a shift to the OS-code base such that in a xen-based environment, no guest OS is available for all operating systems. This condition is used to prevent free hardware-assisted virtualization, which requires the hypervisor to operate in Ring 1 and the guest OS at Ring 0. Xen thus demonstrates some drawbacks with respect to legacy hardware and legacy OS.

Paravirtualization calls for a shift to the OS-code base such that in a xen-based environment, no guest OS is available for all operating systems. This condition is used to prevent free hardware-assisted virtualization, which requires the hypervisor to operate in Ring 1 and the guest OS at Ring 0. Xen thus demonstrates some drawbacks with respect to legacy hardware and legacy OS.

In reality, they cannot be changed in a responsible way to run ring 1, as their codebase is unreachable and the primary hardware currently has no support for running it in a more privileged mode than ring 0. Open source OS like Linux can be updated simply because it has open code and Xen provides full virtualization support, while Windows components are essentially not compliant with Xen until hardware-assisted virtualization is available. With the introduction of new releases of OS, the issue is solved and new hardware will support virtualization of x86.

3.13.2 VMware: full virtualization

In full virtualization main hardware is duplicate and made available to the guest operating system that is unaware of the abstraction and no criteria to change. . VMware technology is based on the full Virtualization. VMware implements full virtualization either in the desktop environment, using the hypervisor Type-II, or in the server environment, using the Type-I hypervisor .For all cases, a full virtualization of the non-sensitive instructions may

be achieved directly and a binary translation f or sensitive instructions or hardware traps, which allows architecture such as x86 to become virtualized.

3.13.3 Full Virtualization and Binary Translation

VMware is frequently used because x86 architectures are essentially virtualized, and it executes their hypervisors unmodified on the top. Full virtualization is possible by implementing hardware-assisted virtualization by supporting hardware. Nevertheless, previously, x86 guest systems could only be implemented with dynamic binary translation without modification in a virtualized environment.

Since sensitive instruction does not constitute a privileged instruction, the first theorem of virtualization is not fulfilled with the design of x86 architecture. Due to this particular activity, these instructions are not implemented in Ring 0, which is usual in a virtualization environment in which the guest operating system is run at Ring 1. Essentially a trap is generated and the process is used in which the solution for x86 is distinguished. In the case of dynamic binary translation, the trap is converted into a series of instructions that specify the same target without making exceptions. Therefore, the corresponding instructions are stored in order to improve the efficiency, hence the translation is no longer necessary for the further encounters of the same instructions. The figure showing it is below.

The main advantage of this method is that guests can operate unmodified in a virtualized environment, an essential component of the operating system, which does not have the source code. For full virtualization, binary translation is portable. Besides translating the instructions during runtime, an additional overhead is not present in other forms, such as paravirtualization or hardware-assisted virtualization. In comparison, binary translation is only done in a subset of the instruction sets, whereas the others are done on the main hardware directly. This somehow reduces the effect on binary translation efficiency.

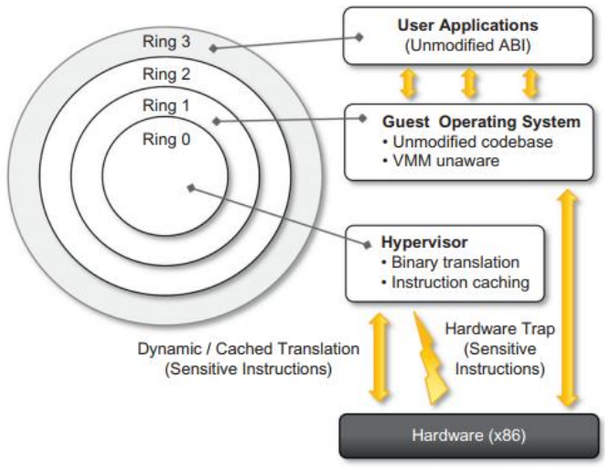


FIGURE 3.17 A full virtualization reference model.
(Reference from "Mastering Cloud Computing Foundations and Applications Programming" by Rajkumar Buyya)

Advantages of Binary Translation

- This method of virtualization provides Virtual Machines with the best isolation and security.
- Many guest OS will actually run concurrently on the same hardware in a very isolated way.
- This is only applied without hardware support or operating system support in the virtualization of sensitive instructions and privileged instructions.

Disadvantages of Binary Translation

- It is time consuming at run-time.
- It achieves a high overhead performance.

- The code cache is used to store the most used translated instructions for improving performance, but it increases memory usage with hardware costs.
- On the x86 architecture, the performance of the complete virtualization is 80-95% of the host machine.

3.13.5 Virtualization solutions

VMware is a pioneer in virtualization and cloud infrastructure solutions that allow our 350,000-plus enterprise and customers to succeed in the cloud age. VMware simplifies its complexity across the entire data center and enables customers with software-defined data center solutions to become hybrid cloud computing and the mobile workspace.

3.13.5 End-user (desktop) virtualization

VMware desktop and app-virtualization technologies give IT a streamlined method for providing, securing and maintaining Windows and Linux desktops and applications on site or on the cloud, reducing costs and ensuring end users can operate everywhere and everywhere. VMware Workstation allows users to run different operating systems on one and the same Windows or Linux PC concurrently. Create real Linux and Windows VMs as well as other desktop, server and tablet environments, complete with virtual networking configurable and network simulation, for use with code development, solution architecture and application testing and product demonstrations, and much.
VMware Fusion allows Mac users the ability to run Windows on Mac together with hundreds of several other operating systems side-by - side without rebooting. Fusion is easy enough for home users and sufficiently efficient for IT experts, developers and businesses. Other than setting up an independent computing environment, the two products enable a guest operating system to exploit host machine resources (USB devices, folder sharing and integration with the host operating system's graphical user interface (GUI). Figure provides a description of the systems' architecture.

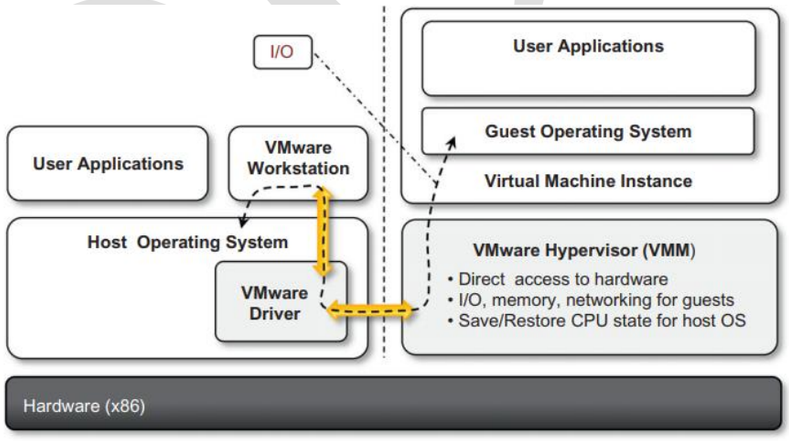


FIGURE 3.18 VMware workstation architecture.
(Reference from "Mastering Cloud Computing Foundations and Applications Programming" by Rajkumar Buyya)

A guest operating system installed application creates the virtualization environment, which enables those operating systems to fully virtualize the hardware that underlie it. This is done through the installation in the host operating system of a special driver which provides two main services:

- It uses a virtual machine manager, which can be used in privileged mode.

- The VMware application offers straps for processing particular I / O requests by subsequently forwarding these requests via system calls to the host operating system.

This architecture, also known as Hosted Virtual Machine Architecture, can both separate virtual machine instances inside an application's memory space and provide decent efficiency, as VMware application's involvement is only essential for instructions, for example, I / O devices which require binary translation. The Virtual Machine manager manages the CPU and the MMU and changes the operational functionality of the CPU and MMU with the host OS. Virtual machine images are stored in a host file system catalogue, and that both VMware and VMware Fusion allow new images to be created, run, create snapshot and undo operational activities by turning back to a previous virtual machine state

VMware Player, VMware ACE and VMware ThinApp are additional technologies relevant to the virtualization of end-user computing environments. VMware Player is a limited VMware Workstation version which enables the creation and emulation of virtual machines of an operating environment such as Windows or Linux. VMware ACE is same as VMware Workstation for developing the policy wrapped virtual machines for provisioning the secure deployment of client virtual environments on end user computers. VMware ThinApp is an application's virtualization solution. It offers an independent development environment to prevent variations due to versioning and incompatible applications. It identifies the operating environment changes by installing a specific app and stores these in a package that can be executed with VMware ThinApp along with the binary app.

3.13.6 Server virtualization

GSX Server is a Windows and Linux virtualized server system developed and distributed by VMware, a subsidiary of EMC Corporation, The program promotes remote management, provisioning and application standardization. Figure demonstrates the architecture of the VMware GSX Server.

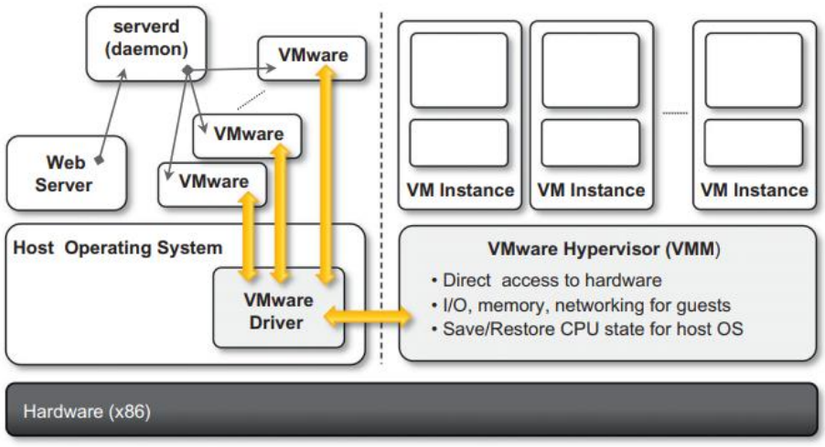


FIGURE 3.19 VMware GSX server architecture.
(Reference from "Mastering Cloud Computing Foundations and Applications Programming" by Rajkumar Buyya)

VMware GSX server converts computers into a collection of virtual machines. Operating systems and frameworks are in separation different Virtual machines on a single hardware device. VMware GSX Server offers wide support for inherited hardware support for the device from the host. The reliable architecture and integration capability of the product VMware GSX Server makes Windows and Linux host environments simple to use and manage. A host program for VMware GSX Server helps you to deploy, monitor and control your application and multiple servers on virtual machines operating remotely
The architecture is designed primarily for web server virtualization. A serverd called daemon process monitors and manages applications for VMware. The VMware driver on the host operating system then connects these programs to the virtual machine instances. The VMM is used to handle virtual machine instances as earlier defined. User requests for managing and providing virtual machines are redirected from the Web server via the serverd via the VMM.

The hypervisor-based method is demonstrated by VMware ESX Server and its improved edition VMWare ESXi Server. Each can be mounted on bare metal servers and provide VM management services. These two products offer similar services, but differ in the internal architecture, particularly in the hypervisor kernel organization. An updated Linux operating system version that allows access to the hypervisor via the service console is implemented by VMware ESX. VMware ESXI introduces an incredibly thin OS layer and substitutes the service console with remote monitoring interfaces and utilities, thus greatly reducing hypervisor code size and memory footprint Figure demonstrates the architecture of VMware ESXi.

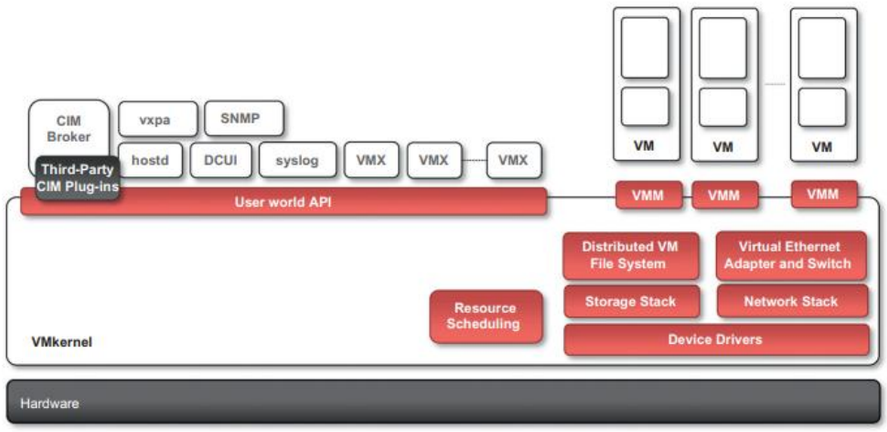


FIGURE 3.20 VMware ESXi server architecture
(Reference from "Mastering Cloud Computing Foundations and Applications Programming" by Rajkumar Buyya)

3.14 Microsoft Hyper-V

Hyper-V is a hypervisor, a virtualizing software of Microsoft which can build and host the enterprise virtual machines on x86-64 systems, such as desktops and servers. Basically, hypervisor is a software that permits several virtual servers (guest machine) to be operated on the physical (host) server. A Hyper-V server may be configured to expose virtual machines to one or more networks. The Windows Server 2008 first launched Hyper-V

3.14.1 Architecture

In terms of partition, Hyper-V implements virtual machine isolation. A partition, supported by the hypervisor, is a logical isolation unit in which every guest operating device performs. In a hypervisor instance, there must be at least one parent partition running Windows Server (2008 and later) enabled. Within the parent partition the virtualization software works and has direct access to hardware devices. The parent partition produces child's partitions containing the guest operating systems. A parent partition uses a hypercall API, the application framework, to build children's partitions exposed to the Hyper-V.

A child partition has no access or its actual interrupts to the physical processor. Rather, the processor has a virtual view and operates in the guest virtual address, which may not actually be the entire virtual address space depending upon on configuration of the hypervisor. Hyper-V will only show a subset of processors on each partition, depending on the VM configuration. The hypervisor manages the interrupts to the processor with the aid of a logical Synthetic Interrupt controller (SynIC) to the respective partition.

Child partitions provide a virtual view of the resources of the virtual machines, rather than a direct access for hardware sources. Each virtual device request is sent to the parent partition devices, which handle the requests, through the VMBus. The VMBus is a logical channel that allows communication between sections. The response is also forwarded through the VMBus. When the parent partition devices are also virtual devices, they are routed to the parent partition where physical devices are accessed. Parent partitions operate a Virtualization Service Provider (VSP) that connects to the VMBus and handles requests for system access from the child partitions. Child's virtual partition devices run the

Virtualization Service Client (VSC) internally, redirecting the program to the VSPs on the VMBus parent. To the guest OS, this whole process is transparent.

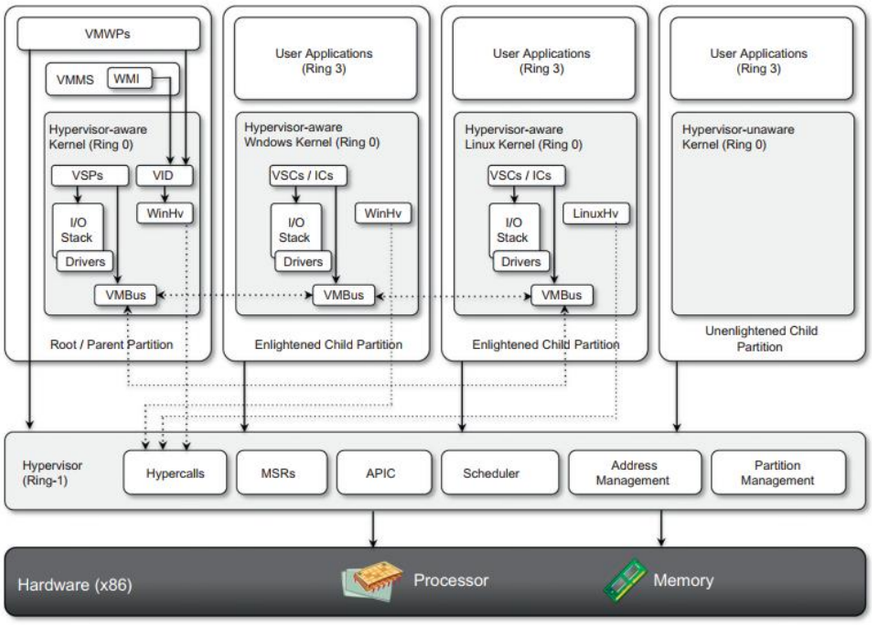


FIGURE 3.17 Microsoft Hyper-V architecture

(Reference from "Mastering Cloud Computing Foundations and Applications Programming" by Rajkumar Buyya)

3.15 Summary

Virtualization is an essential framework for a number of technologies and concepts. The popular source for virtualization is the ability to demonstrate, using some kind of emulation or abstraction layer, a given runtime environment, whether a software, a storage facility, a network connection or a remote desktop. In building cloud services and infrastructure, all these principles play an essential role, wherein hardware, IT infrastructure, applications and services are provided on demand via the Internet or usually through a network connection.

3.16 Review questions

1. Define Virtualization. What are the advantages of virtualization?
2. What are the characteristics of virtualized environments?
3. Describe classification or taxonomy of virtualization at different levels.
4. Discuss the execution virtualization machine reference model.
5. What are the techniques of hardware virtualization?
6. List and discuss various virtualization types.
7. What are the benefits of virtualization in the context of cloud computing?
8. What are the disadvantages of virtualization?
9. What is Xen? Discuss its elements for virtualization.
10. Discuss the reference model of full virtualization.
11. Discuss the reference model of paravirtualization.
12. Discuss the architecture of Hyper-V. Discuss its use in cloud computing

3.17 Reference for further reading

1. Mastering Cloud Computing Foundations and Applications Programming Rajkumar Buyya ,Christian Vecchiola,S. Thamarai Selvi MK publications ISBN: 978-0-12-411454-8

2. Cloud Computing Concepts, Technology & Architecture Thomas Erl, Zaigham Mahmood, and Ricardo Puttini , The Prentice Hall Service Technology Series ISBN-10 : 9780133387520 ISBN-13 : 978-0133387520

3. Distributed and Cloud Computing: From Parallel Processing to the Internet of Things 1st Edition by Kai Hwang Jack Dongarra Geoffrey Fox ISBN-10 : 9789381269237 ISBN-13 : 978-9381269237

Unit 2

Chapter 1

2.1.0. Objective

2.1.1. Introduction to Cloud Computing Architecture

2.1.1.1. Architecture

2.1.2. Fundamental concepts and models

2.1.2.1. Roles and Boundaries

2.1.2.1.1. Cloud Provider

2.1.2.1.2. Cloud Consumer

2.1.2.1.2.1. Cloud Service Owner

2.1.2.1.4. Cloud Resource Administrator

2.1.4. Cloud Characteristics

2.1.5. Cloud Delivery models.

2.1.5.1. On-Demand Usage

2.1.5.2. Ubiquitous Access

2.1.5.2.1. Multitenancy

2.1.5.4. Elasticity

2.1.5.5. Measured Usage

2.1.5.6. Resiliency

2.1.6. Cloud Deployment models

2.1.6.1. Infrastructure-as-a-Service (IaaS)

2.1.6.2. Platform-as-a-Service (PaaS)

2.1.6.2.1. Software-as-a-Service (SaaS)

2.1.7. Economics of the cloud

2.1.7.1. Public Clouds

2.1.7.2. Community Clouds

2.1.7.2.1. Private Clouds

2.1.7.4. Hybrid Clouds

2.1.8. Open challenges.

2.1.0. Objective

After going through this chapter, you will be able to:

- Cloud Computing Architecture
- Introduction
- Fundamental concepts and models
- Roles and boundaries
- Cloud Characteristics
- Cloud Delivery models
- Cloud Deployment models
- Economics of the cloud
- Open challenges.

2.1.1. Prologue to Cloud Computing Architecture:

Distributed computing bolsters any IT administration which will be devoured as an utility and conveyed through a system, apparently the web . Such portrayal incorporates very various perspectives: framework, advancement stages, application and administrations.

2.1.1.1. Engineering

It is conceivable to orchestrate all the solid acknowledge of distributed computing into a layered view covering the entire stack (see Figure 2.1.1), from equipment machines to programming frameworks. Cloud assets are tackled to flexibly "registering pull" required for offering types of assistance. Frequently, this layer is actualized utilizing a datacenter during which hundreds and thousands of hubs are stacked together. Cloud framework are regularly heterogeneous in nature in bright of the detail that a spread of assets, similar to bunches and even organized PCs, are frequently wont to construct it. Additionally, database frameworks and other stockpiling administrations likewise can be a piece of the foundation.

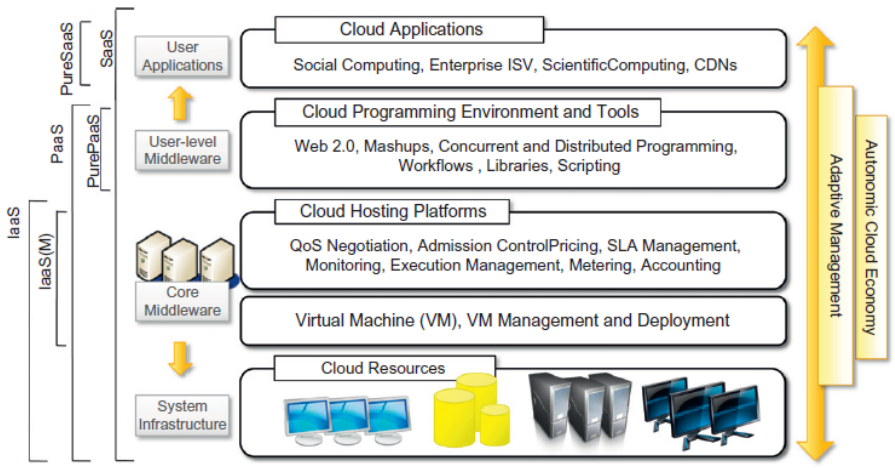


Figure 2.1.1 The cloud computing architecture

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

The physical framework is accomplished by the center middleware, the destinations of which are to flexibly a fitting runtime condition for applications and to best use assets. At the absolute

bottom of the stack, virtualization innovations are wont to ensure runtime condition customization, application seclusion, sandboxing, and nature of administration.

Equipment virtualization is most normally utilized at this level. Hypervisors accomplish the pool of properties and uncover the scattered framework as a lot of virtual machines. By devouring virtual machine innovation it's probably going to incredibly divider the equipment assets like CPU and memory and to virtualize explicit gadgets, therefore meeting the needs of clients and applications. This goal is regularly matched with capacity and system virtualization approaches, which license the framework to be totally virtualized and controlled. reliable with the exact assistance offered to complete clients, other virtualization procedures are regularly utilized; for instance, development-level virtualization supports in making a transportable runtime condition where applications are frequently run and controlled. This situation for the most part suggests that applications facilitated inside the cloud be created with a chose innovation or a programing language, similar to Java, .NET, or Python. during this case, the client doesn't have to assemble its framework from exposed metal. Foundation the executives is the principle motivation behind center middleware, which underpins skills like the trade off of the standard of administration, affirmation control, execution the board and checking, bookkeeping, and charging.

2.1.2. Essential ideas and models

The blend of cloud facilitating stages and assets is typically delegated an Infrastructure-asa-Service (IaaS) arrangement. we will found the various instances of IaaS into two classifications: various them convey both the administration layer and subsequently the physical framework; others convey just the administration layer (IaaS (M)). during this subsequent case, the administration layer is normally incorporated with different IaaS arrangements that give physical foundation and increases the value of them.

IaaS arrangements are appropriate for planning the framework foundation however offer constrained types of assistance to make applications. Such assistance is given by cloud programming conditions and apparatuses, which structure a trade layer for offering clients an improvement stage for applications. The scope of instruments incorporate Web-based interfaces, order line apparatuses, and structures for simultaneous and circulated programming. In this circumstance, clients build up their applications decisively for the cloud by utilizing the API uncovered at the client level middleware. Thus, this strategy is otherwise called Platform-as-a-Service (PaaS) on the grounds that the office offered to the client is an improvement stage instead of a foundation. PaaS arrangements for the maximum share incorporate the framework too, which is packaged as a major aspect of the administration gave to clients. On account of Unadulterated PaaS, just the client level middleware is offered, and it must be supplemented with a virtual or physical foundation.

The top layer of the reference model depicted in Figure 2.1.1 spreads offices brought at the application level. These are commonly referenced to as Software-as-a-Service (SaaS). By and large, these are Web-put together applications that trust by reverence to the cloud to offer support to end-clients. The strength of the cloud-conveyed by IaaS and PaaS goals permits self-governing programming sellers to convey their application offices over the Internet.

Extra applications setting off to this layer are those that unequivocally impact the Internet for their center functionalities that trust on the cloud to withstand a bigger number of clients; this is the situation of gaming entryways and, all in all, person to person communication sites.

Table 2.1.1 typifies the attributes of the three primary classifications used to classify cloud ascertaining arrangements. In the accompanying segment, we incidentally purposeful these qualities alongside certain directions to reasonable usage.

Category	Characteristics	Product Type	Vendors and Products
SaaS	Consumers are delivered with application that are available anytime and from anywhere	Web application and services(Web 2.0)	SalesForcce.com(CRM) Clarizen.com(Project management) Google Apps
PaaS	Consumers are delivered with a stage emerging applications hosted in the doud	Programming APIs and frameworks Deployment systems	Google AppEngine Microsoft Azure Manjrasoft Aneka Data Synapse
IaaS/HaaS	Consumers are delivered with virtualized hardware and storage on top of which they can build their infrastructure.	Virtual machine management infrastructure storage management network management	Amazon EC2 and S2 GoGrid Nirvanix

2.1.2.1. Jobs and cutoff points

Associations and people can accept contrasting sorts of pre-characterized jobs relying on how they identify with or potentially interface with a cloud and its introduced IT assets. Every one of things to come parts contributes in and passes on out accountabilities regarding cloud-based activity. the accompanying segments depict these parts and recognize their principle associations.

2.1.2.1.1. Cloud Provider

The association that gives cloud-based IT assets is that the cloud supplier. While assuming the job of cloud provider, a company is at risk for making cloud administrations accessible to cloud customers, according to endorsed SLA ensures. The cloud supplier is extra entrusted with any basic administration and regulatory duties to ensure the on-going procedure of the general cloud foundation. Cloud providers normally own the IT properties that are made available for rent by cloud purchasers; be that as it may, some cloud providers additionally "exchange" IT assets rented from other cloud suppliers.

2.1.2.1.2. Cloud Consumer

A cloud purchaser is an affiliation (or a human) that has official agreement or course of action with a cloud supplier to utilize IT properties made accessible by the cloud supplier. Expressly, the cloud buyer utilizes a cloud administration shopper to get to a cloud administration (Figure).

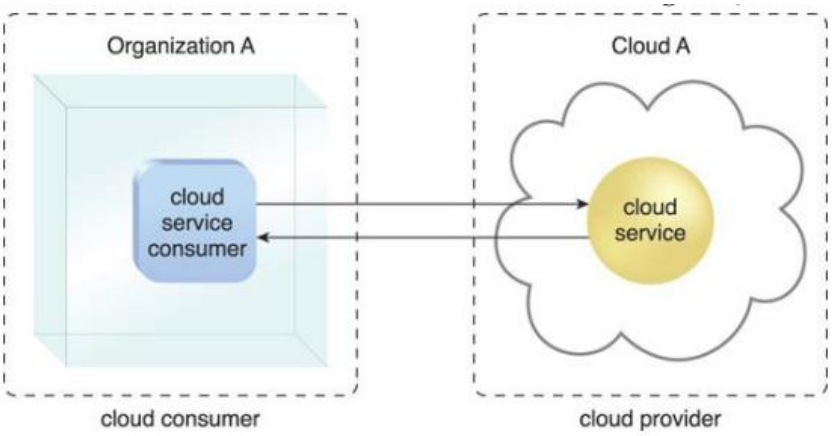


Figure: A cloud consumer (Organization A) cooperates with a cloud service from a cloud provider (that owns Cloud A). Within Group A, the cloud service customer is being used to access the cloud service.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

2.1.2.1.2.1. Cloud Service Owner

The individual or affiliation that authoritatively claims a cloud administration is known as a cloud administration proprietor. The cloud administration proprietor can be the cloud shopper or the cloud provider that claims the cloud inside which the cloud administration dwells. For instance, in addition, the cloud customer of Cloud X or the cloud supplier of Cloud X could possess Cloud Service A (Figures 2.1.2 and 2.1.2.1).

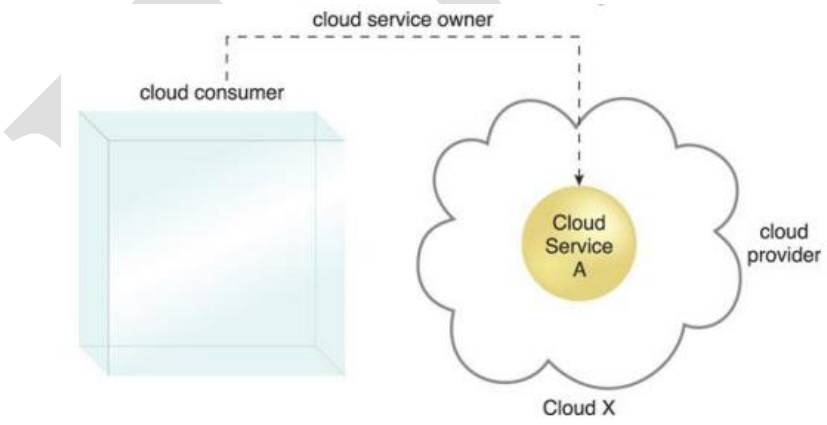


Figure2.1.2. A cloud consumer can be a cloud service owner when it deploys its own service in a cloud.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

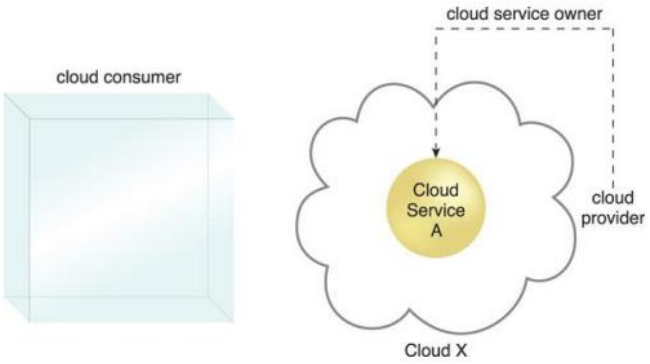


Figure2.1.2.1. A cloud provider becomes a cloud service owner if it deploys its own cloud service, typically for other cloud consumers to use.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

2.1.2.1.4. Cloud Resource Administrator

A cloud asset executive is an individual or affiliation responsible for taking care of a cloud-based IT asset (counting cloud administrations). The cloud asset overseer can be (or have a place with) the cloud customer or cloud supplier of the cloud inside which the cloud administration exists in. Else, it very well may be (or have a place with) an outsider affiliation contracted to coordinate the cloud-based IT asset. For instance, a cloud office owner can get a cloud asset head to oversee a cloud administration (Figures 2.1.4 and 2.1.5).

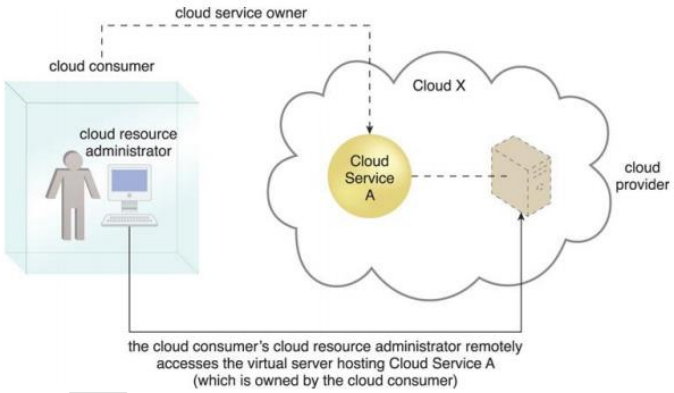


Figure2.1.4. A cloud resource administrator can be with a cloud consumer organization and administer remotely accessible IT resources that belong to the cloud consumer.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

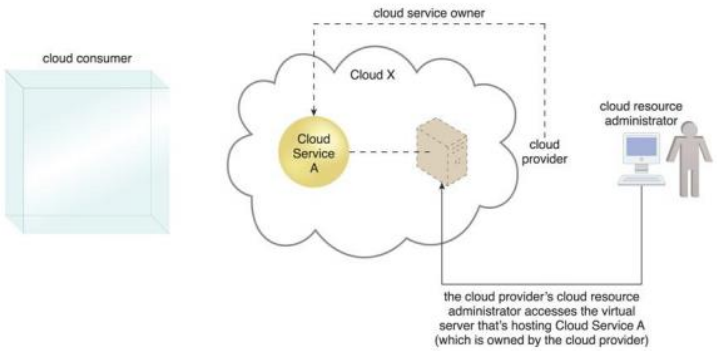


Figure2.1.5. A cloud resource administrator can be with a cloud provider organization for which it can administer the cloud provider's internally and externally available IT resources.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

2.1.2.1.4. Cloud Resource Administrator

A cloud asset executive is an individual or affiliation responsible for taking care of a cloud-based IT asset (counting cloud administrations). The cloud asset overseer can be (or have a place with) the cloud customer or cloud supplier of the cloud inside which the cloud administration exists in. Else, it very well may be (or have a place with) an outsider affiliation contracted to coordinate the cloud-based IT asset. For instance, a cloud office owner can get a cloud asset head to oversee a cloud administration (Figures 2.1.4 and 2.1.5).

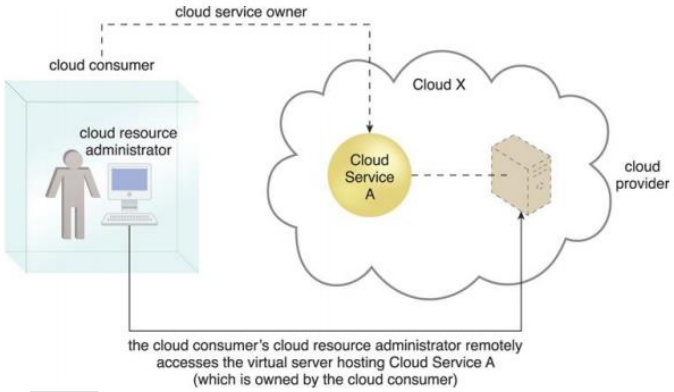


Figure2.1.4. A cloud resource administrator can be with a cloud consumer organization and administer remotely accessible IT resources that belong to the cloud consumer.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

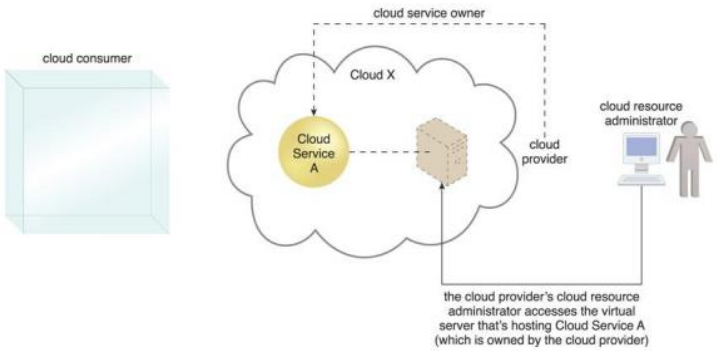


Figure2.1.5. A cloud resource administrator can be with a cloud provider organization for which it can administer the cloud provider's internally and externally available IT resources.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

The explanation a cloud asset executive isn't alluded to as a "cloud administration manager" is on the lands that this job might be answerable for directing cloud-based IT assets that don't exist as cloud administrations. For instance, if the cloud asset chairman fits to (or is Shrunk by) the cloud supplier, IT assets not made remotely available might be controlled by this job (and these sorts of IT assets are not delegated cloud administrations).

2.1.4. Limit

2.1.4.1. Hierarchical Boundary

A hierarchical limit implies the physical outskirt that conditions a lot of IT capitals that are had and controlled by an association. The authoritative limit doesn't show the limit of a real association, just a hierarchical arrangement of IT properties and IT assets. Also, mists have an authoritative limit (Figure 2.1.6.).

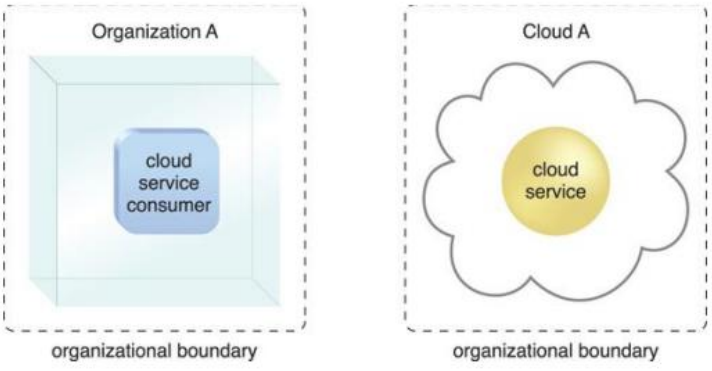


Figure2.1.6. Organizational boundaries of a cloud consumer (left), and a cloud provider (right), represented by a broken line notation.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

2.1.4.2. Trust Boundary

At the point when an association embraces the job of cloud client to get to cloud-based IT assets, it needs to spread its trust outside the physical limit of the association to contain portions of the cloud condition. A trust limit is a coherent fringe that normally ranges outside physical limits to connote the degree to which IT assets are trusted (Figure 2.1.7). When investigating cloud situations, the trust limit is most every now and again associated with the trust gave by the association going about as the cloud purchaser.

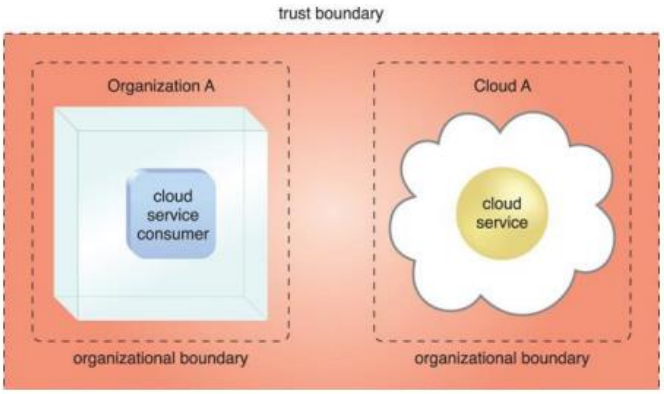


Figure 2.1.7. An extended trust boundary encompasses the organizational boundaries of the cloud provider and the cloud consumer.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

2.1.5. Cloud Characteristics

An IT air needs a careful arrangement of qualities to empower the removed provisioning of walkable and estimated IT capitals in a real manner. These qualities need to exist to an expressive degree for the IT climate to be estimated a viable cloud.

The accompanying six accurate attributes are normal to the standard of cloud situations: • on-request utilization
• pervasive access
• multitenancy (and asset pooling)
• flexibility
• estimated utilization
• strength
Cloud suppliers and cloud clients can quantify these attributes independently and together to gauge the worth commitment of a given cloud stage. Despite the fact that cloud-based administrations and IT assets will get and display particular qualities to variable degrees, generally the better how much they are strengthened and applied, the better the resulting esteem proposal.

2.1.5.1. On-Demand Usage

A cloud client can separately get to cloud-based IT properties giving the cloud client the self-rule to self-arrangement these IT properties. When sorted out, utilization of oneself provisioned IT properties can be programmed, needful no extra human interest by the cloud client or cloud supplier. This outcomes in an on-request utilization circumstance. Otherwise called "on-request self-administration utilization," this trademark permits the administration based and use driven highlights begin in customary mists.

2.1.5.2. Universal Access

Universal access connotes the fitness for a cloud administration to be widely accessible. Establishing omnipresent access for a cloud administration can require arrangement for a scope of systems, transportation conventions, limits, and wellbeing advances. To allow this degree of access normally needs that the cloud administration design be customized to the particular prerequisites of various cloud administration clients.

2.1.5.2.1. Multitenancy (and Resource Pooling)

The quality of a product bundle that permits a case of the program to support various clients (occupants) whereby each is remote from the other, is referenced to as multitenancy. A cloud supplier pools its IT properties to enable various cloud to support clients by utilizing multitenancy imitations that routinely trust on the utilization of virtualization advancements. Over the utilization of multitenancy innovation, IT properties can be animatedly allotted and reallocated, rendering to cloud administration client requests.

2.1.5.4. Versatility

Versatility is the programmed inclination of a cloud to unmistakably scale IT properties, as required in answer to runtime circumstances or as customized by the cloud client or cloud supplier. Versatility is regularly estimated a center resistance for the acknowledgment of distributed computing, primarily because of the way that it is firmly related with the Abridged Asset and Comparative Costs advantage. Cloud suppliers with tremendous IT properties can offer the most noteworthy scope of versatility

2.1.5.5. Estimated Usage

The deliberate utilization trademark means the fitness of a cloud stage to keep way of the use of its IT assets, for the most part by cloud clients. Established on what is estimated, the cloud supplier can charge a cloud client just for the IT properties truly utilized as well as for the time span through which access to the IT properties was chosen. In this unique circumstance, estimated utilization is firmly associated with the on-request trademark.

2.1.5.6. Strength

Strong computing is a type of failover that dispenses excess utilizations of IT properties across physical spots. IT properties can be pre-arranged so that in the event that one gets lacking, agreement is consequently given over to extra excess application. Inside distributed computing, the attribute of flexibility can make reference to repetitive IT properties inside a similar cloud (however in various physical areas) or over various mists. Cloud clients can

development both the reliability and availability of their applications by utilizing the strength of cloud-based IT properties.

2.1.6. Cloud conveyance model

A cloud conveyance model connotes an assigned, pre-bundled blend of IT assets available by a cloud supplier. Three common cloud conveyance models turned out to be comprehensively perceived and honorable:

• Infrastructure-as-a-Service (IaaS)

• Platform-as-a-Service (PaaS)

• Software-as-a-Service (SaaS)

2.1.6.1. Framework as-a-Service (IaaS)

The IaaS circulation model implies an independent IT climate contained of foundation driven IT assets which will be recovered and accomplished by means of cloud administration based interfaces and devices. This climate can incorporate equipment, organize, network, working frameworks, and other "crude" IT assets. In distinction to conventional facilitating or redistributing environmental factors, with IaaS, IT assets are normally virtualized and pressed into wraps that compress in advance runtime climbing and customization of the framework. the broadly useful of an IaaS domain is to flexibly cloud customers with an elevated level of control and responsibility over its.

design and use. The IT assets gave by IaaS are by and large not pre-arranged, setting the official obligation straightforwardly upon the cloud shopper. This model is consequently utilized by cloud buyers that need a significant level of command over the cloud-based condition they will make. Here and there cloud suppliers will contract IaaS contributions from other cloud suppliers in order to scale their own cloud surroundings. the sorts and makes of the IT assets gave by IaaS items offered by various cloud suppliers can change. IT assets accessible through IaaS conditions are for the most part offered as newly instated virtual occurrences. A focal and first IT asset inside a run of the mill IaaS condition is that the virtual server. Virtual servers are rented by indicating server equipment necessities, similar to processor limit, memory, and local space for putting away, as appeared in Figure

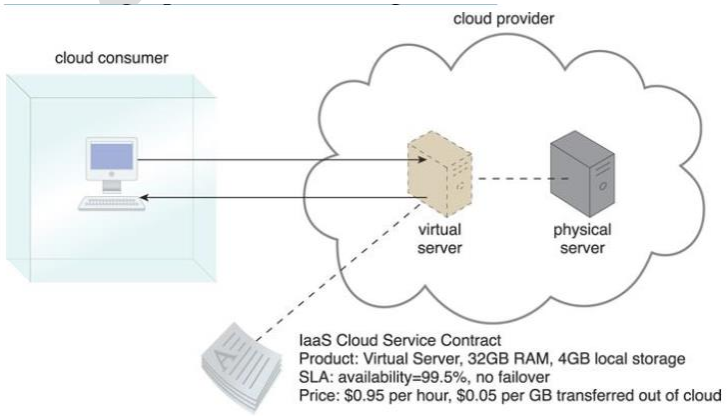


Figure 2.1.8. A cloud customer is using a virtual server within an IaaS atmosphere. Cloud consumers are delivered with a range of contractual guarantees by the cloud provider, relating to physiognomies such as capacity, performance, and availability.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

2.1.6.2. Stage as-a-Service (PaaS)

The PaaS conveyance model says to a pre-categorized "prepared to-utilize" condition ordinarily involved previously sent and arranged IT assets. In specific, PaaS be subject to on the application of an prompt area that sets up a lot of pre-hustled stuffs and instruments used to help the whole conveyance lifecycle of custom applications.

Even motives a cloud buyer would apply and place properties into a PaaS domain include:

• The cloud buyer needs to reach out on-premise conditions into the cloud for versatility and financial purposes.

• The cloud customer utilizes the instant condition to totally substitute an on-premise condition.

• The cloud consumer wants to go into a cloud dealer and takes its personal cloud administrations to be complete available to additional outer cloud buyers.

By employed privileged an immediate phase, the cloud shopper is saved the authoritative mass of location up and keeping up the exposed foundation IT assets gave by means of the IaaS model. On the other hand, the cloud customer is conceded a lower level of authority over the fundamental IT assets that host and arrangement the stage (Figure 4.12).

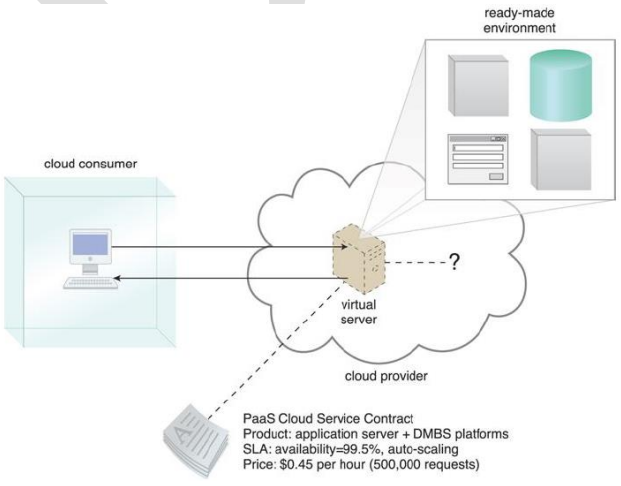


Figure 2.1.9. A cloud consumer is accessing a ready-made PaaS environment. The question mark indicates that the cloud consumer is intentionally shielded from the implementation details of the platform.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

PaaS products are available with different development stacks. For example, Google App Engine offers a Java and Python-based environment.

2.1.6.2.1. Programming as-a-Service (SaaS)

A product program situated as a typical cloud administration and made open as an "item" or general worth connotes the standard profile of a SaaS offering. The SaaS conveyance model is normally wont to make a refillable cloud administration extensively available (frequently monetarily) to an assorted variety of cloud clients. Complete profitable center occurs about SaaS matters which will be borrowed and applied for different drives and by incomes of numerous terms

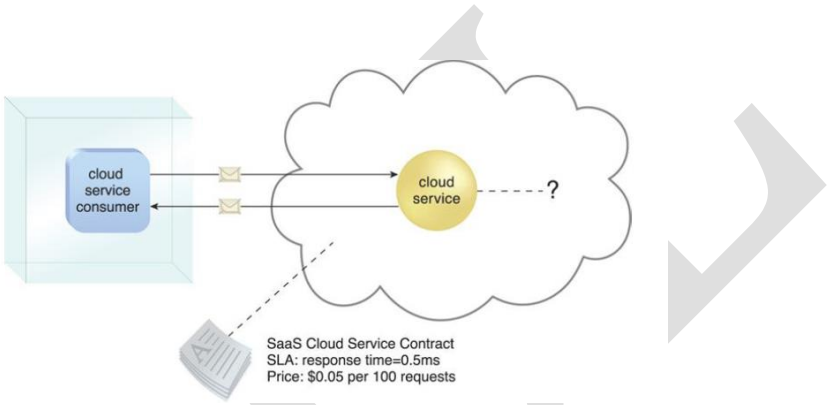


Figure 2.1.10. The cloud service customer is given access the cloud agreement, but to not any fundamental IT resources or application details.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

A cloud purchaser is normally allowed constrained authoritative command above a SaaS practice. it's most normally provisioned by the cloud provider, yet it are frequently formally claimed by whichever element expect the cloud administration proprietor job. for example , an enterprise going about as a cloud buyer while utilizing and managing a PaaS domain can assemble a cloud administration that it chooses to convey in that equivalent condition as a SaaS offering. An identical association at that point adequately accept the cloud supplier job in bright of the fact that the SaaS-based cloud administration is framed accessible to different associations that go about as cloud purchasers when utilizing that cloud administration.

2.1.7. Cloud Deployment Models

A cloud procedure classical says to a selected kind of cloud state, basically documented by proprietorship, scope, and access. There are four steady cloud sending models:
• Public cloud
• Community cloud
• Private cloud
• Hybrid cloud the resulting areas depict each.

2.1.7.1. Open Clouds

An open cloud might be a freely accessible cloud environment claimed by an outsider cloud provider. The IT properties on exposed hazes are commonly provisioned through the recently characterized cloud transportation replicas and are typically offered to cloud purchasers at a worth or are marketed by means of different ways. a cloud breadwinner is accountable for the arrangement and on-going support of the overall population cloud and its IT properties. A significant number of the circumstances and designs investigated in forthcoming sections include open mists and accordingly the association among the providers and buyers of IT assets through open mists. Figure 2.1.10. shows a halfway perspective on the general open cloud scene, featuring some of the main sellers inside the commercial center.

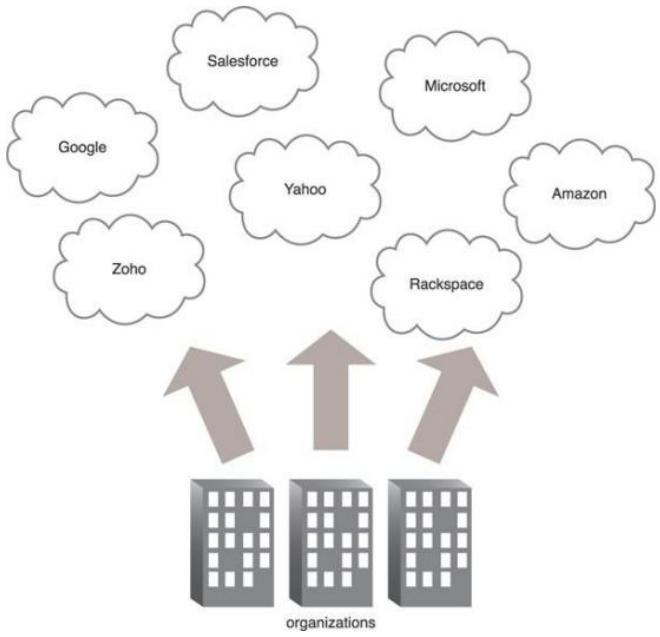
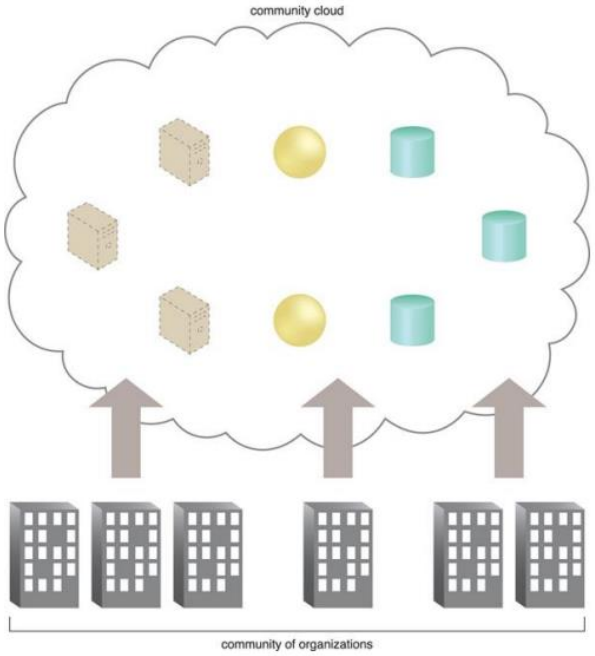


Figure 2.1.11. Organizations act as cloud consumers when accessing cloud services and IT resources made available by different cloud providers.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

2.1.7.2. Network Clouds

A people group cloud resembles to a network cloud aside from that its entrance is limited to a specific network of cloud buyers. The open cloud might be together controlled by the open partners or by an outsider cloud supplier that provisions an open cloud with restricted access. The partner cloud buyers of people in general commonly share the responsibility for characterizing and developing the network cloud (Figure 2.1.11.).



(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

2.1.7.2.1. Private Clouds

An individual cloud is claimed by one association. Private mists empower a partnership to utilize distributed computing innovation as a method of concentrating access thereto assets by various parts, areas, or divisions of the association. At the point when an individual cloud exists as a controlled situation, the issues portrayed inside the Risks and Challenges segment from Chapter 2.1 don't will in general use

The utilization of an individual cloud can change how authoritative and believe limits are characterized and applied. the specific organization of an individual cloud condition could likewise be regulated by inside or re-appropriated staff.

With an individual cloud, an identical association is actually both the cloud purchaser and cloud supplier (Figure 2.1.12.1). to separate these jobs:

• an unmistakable authoritative segment normally accept the responsibility for provisioning the cloud (and thusly expect the cloud supplier job)

• Divisions requesting access to the private cloud expect the cloud customer job.

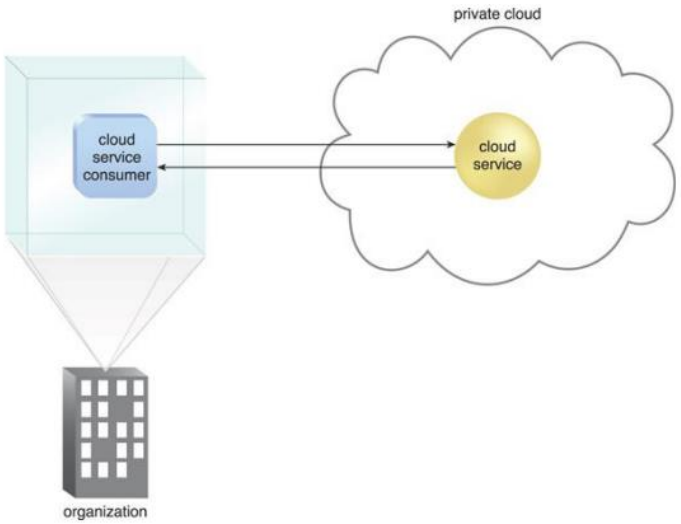


Figure 2.1.12.1. A cloud service consumer within the organization's on-premise environment accesses a cloud service hosted on an equivalent organization's private cloud via a virtual private network.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

It is critical to utilize the affinities "on-reason" and "cloud-based" accurately inside the setting of an individual cloud. but the private cloud may genuinely dwell on the association's premises, IT assets it armed forces are as yet estimated "cloud-based" insofar as they're made remotely open to cloud shoppers. IT assets facilitated outside of the private cloud by the segments going about as cloud clients are along these lines considered "on-premise" regarding the private cloud-based IT assets.

2.1.7.4. Half and half Clouds

A half and half cloud may be a cloud environment included of at least two differing cloud arrangement models. for example , a cloud client may like to carry cloud managements making touchy information to an individual cloud and extra, fewer delicate cloud administrations to an open cloud. The aftereffects of this blend might be a half and half organization model (Figure 2.1.12.1).

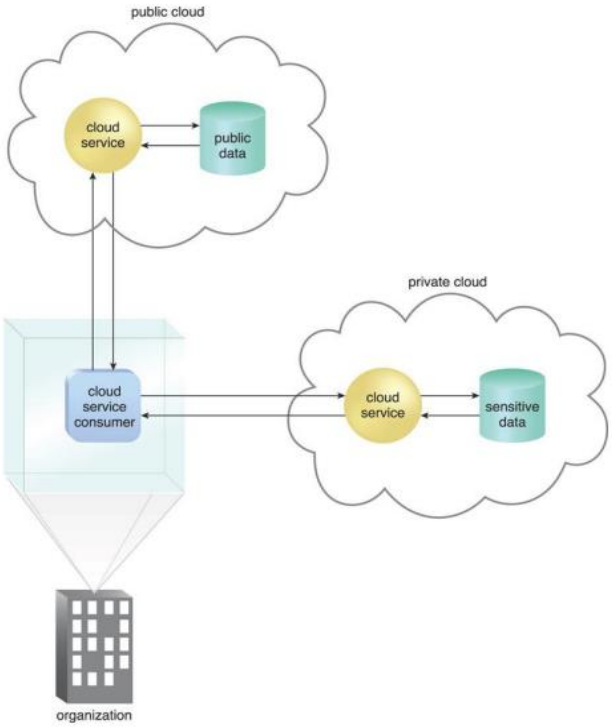


Figure 2.1.14. a establishment employing a hybrid cloud architecture that uses both a individual

and public cloud.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

Mixture mien structures are regularly compound and testing to make and proceed with gratitude to the conceivable contrast in cloud airs and in this way, the undeniable actuality that organization accountabilities are traditionally part between the private cloud supplier association and the open cloud supplier.

2.1.8. Financial matters of the cloud

The principle drivers of distributed computing are economy of scale and simplicity of programming conveyance and its activity. Indeed, the most significant favorable position of this wonder is monetary: the pay-more only as costs arise model offered by cloud suppliers. particularly, distributed computing permits:

• Reducing the capital expenses related with the IT framework

• Eliminating the devaluation or lifetime costs identified with IT capital resources

• Replacing programming permitting with memberships

• Cutting the upkeep and regulatory expenses of IT assets

An expense of capital is that the expense happened in buying an advantage that is valuable inside the creation of items or the rendering of administrations. Capital expenses are one-time

costs that are commonly paid forthright which will contribute over the future to get benefit. The IT foundation and along these lines the product is capital resources since endeavors expect them to lead their business. at the present, it doesn't make a difference whether the foremost business of an endeavor is said there to in bright of the fact that the business will surely have an IT office that is wont to robotize a considerable lot of the exercises that are performed inside the venture: finance, client relationship the board, undertaking asset arranging, following and stock of items, and others. Subsequently, IT assets comprise an expense of capital for any very endeavor. it's acceptable practice to embrace to remain capital costs low since they present costs which will create benefit after some time; very that, since they're identified with material things they're dependent upon devaluation above the extensive run, which inside the end diminishes the benefit of the undertaking in light of the fact that such expenses are legitimately deducted from the venture incomes. inside the instance of IT capital costs, the deterioration costs are spoken to by the less valuable of the equipment after some time and in this manner the maturing of programming items that require to get supplanted on the grounds that new highlights are required.

Before distributed computing diffused inside the venture, the financial plan spent consequently framework and programming comprised a major cost for medium-sized and gigantic endeavors. Numerous undertakings own a little or medium-sized datacenter that presents a few operational expenses as far as upkeep, power, and cooling. Extra operational expenses are happened in keeping up an IT division and an IT bolster focus. In addition, different expenses are activated by the securing of likely costly programming. With distributed computing, these expenses are altogether decreased or simply vanish reliably with their infiltration. one of the advantages presented by the distributed computing model is that it moves the capital expenses recently designated to the securing of equipment and programming into operational expenses drafted by leasing the foundation and paying memberships for the usage of the product.

These expenses are regularly better controlled predictable with the business needs and flourishing of the endeavor. Distributed computing likewise presents decreases in authoritative and support costs. the amount of cost reserve funds that distributed computing can present inside an undertaking is said to the particular situation during which cloud administrations are utilized and the manner in which they add to get a benefit for the venture. inside the instance of a little startup, it's conceivable to totally use the cloud for a few angles, for example,

•IT foundation

• Software improvement

• CRM and ERP

For this situation, it's conceivable to totally dispose of capital expenses in light of the fact that there are no underlying IT resources. things are entirely unexpected inside the instance of undertakings that have just got a generous measure of IT resources. during this case, distributed computing, particularly IaaS-based arrangements, can help oversee impromptu capital costs that are created by the prerequisites of the attempt privileged the current instant. during this case, by utilizing distributed computing, these expenses are frequently turning out

to be operational costs that keep going as long as there's a prerequisite for them. for example, IT foundation renting helps all the more productively oversee top burdens without actuating capital costs. As soon in light of the fact that the expanded burden doesn't legitimize the use of extra assets, these are regularly discharged and consequently the costs identified with them vanish. this is frequently the premier embraced model of distributed computing in light of the fact that numerous undertakings have just got IT offices. an elective decision is to frame a moderate progress toward cloud-based arrangements while the capital IT resources get deteriorated and wish to get supplanted. Between these two cases, there's a decent kind of situation during which distributed computing may be of help in producing benefits for endeavors. Regarding the valuing models presented by distributed computing, we will recognize three unique methodologies that are received by the suppliers.

Reference

Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini

Unit 2

Chapter 2

2.1.Fundamental Cloud Security

Data security is a compound group of procedures, advancements, guidelines, and exhibitions that cooperatively guard the genuineness of and access to PC frameworks and information. IT security occasions expect to ensure against dangers and obstruction that ascent from both noxious purpose and accidental client mistake.

2.1.1. Secrecy

Secrecy is the attribute of something being made available just to approved gatherings (Figure 2.1). Inside cloud situations, secrecy basically relates to limiting access to information in travel and capacity.

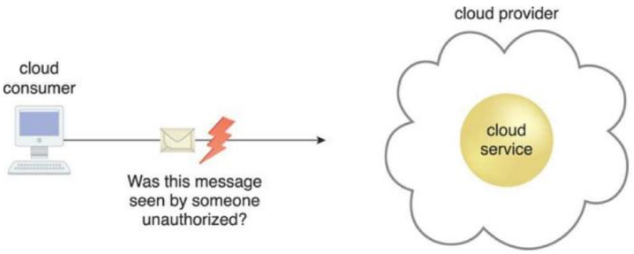


Figure2.1.: The message issued by the cloud consumer to the cloud service is considered confidential only if it is not accessed or read by an unauthorized party.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

2.1.2.Integrity

Honesty is the quality of not hosting been changed by an unapproved gathering (Figure 2.2). A important matter that worries information respectability in the cloud is whether a cloud buyer can be ensured that the information it transmits to a cloud administration coordinates the information got by that cloud administration. Respectability can stretch out to how info is set away, prepared, and recovered by cloud administrations and cloud-based IT assets.

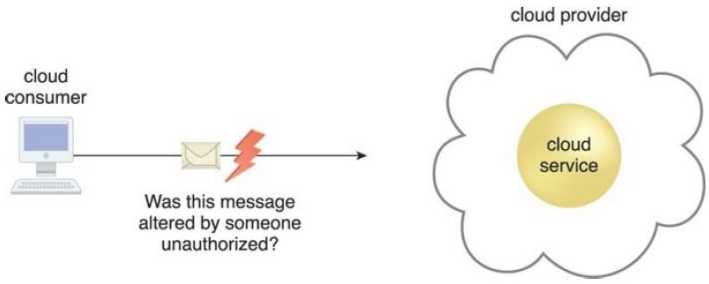


Figure 2.2.: The message issued by the cloud consumer to the cloud service is considered to have integrity if it has not been altered.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

2.1.3. Realness

Realness is the quality of something having been given by an approved source. This knowledge includes non-revocation, which is the letdown of a meeting to reject or challenge the authorization of a association. Verification in non-repudiable connections gives evidence that these associations are remarkably connected to an approved source. Now take example, a consumer will most expected be unable to get to a non-reputable article later its receiving deprived of similarly generating a record of this entry.

2.1.2. Accessibility

Accessibility is the attribute of being open and usable during a predefined timeframe. In average cloud situations, the accessibility of cloud administrations can be an obligation that is shared by the cloud supplier and the cloud bearer. The accessibility of a cloud-based arrangement that reaches out to cloud administration customers is additionally shared by the cloud purchaser.

2.1.5. Danger

A danger is a potential security infringement that can move guards trying to break protection and additionally cause hurt. Both physically and naturally incited dangers are intended to misuse referred to shortcomings, likewise alluded to as vulnerabilities. A danger that is done outcomes in an assault.

2.1.6. Powerlessness

A defenselessness is a shortcoming that can be misused either in bright of the detail that it is ensured by inadequate security controls, or on the grounds that present safety panels are overwhelmed by an assault. IT asset vulnerabilities can have a scope of causes, including setup inadequacies, security strategy shortcomings, client mistakes, equipment or firmware defects, programming bugs, and helpless security design.

2.1.7. Hazard

Hazard is the chance of misfortune or mischief emerging from playing out an action. Hazard is regularly estimated by its danger level and the quantity of conceivable or known vulnerabilities. Two measurements that can be utilized to decide chance for an IT asset are:

• The likelihood of a danger happening to abuse vulnerabilities in the IT asset

• The desire for misfortune upon the IT asset being undermined

Insights about hazard the board are shrouded later in this section.

2.1.8. Security Controls

Security controls are countermeasures used to forestall or react to security dangers and to decrease or maintain a strategic distance from hazard. Subtleties on the greatest capable technique to use security countermeasures are ordinarily laid out in the security strategy, which contains a lot of rules and works on determining how to execute a framework, administration, or security plan for greatest assurance of touchy and basic IT assets.

2.1.9. Security Mechanisms

Countermeasures are commonly depicted as far as security systems, which are parts containing a guarded structure that ensures IT assets, data, and administrations.

2.1.10. Security Policies

A security strategy sets up a lot of security rules and guidelines. Regularly, security strategies will additionally characterize how these guidelines and guidelines are executed and implemented. For instance, the situating and use of security controls and components can be controlled by security arrangements

2.2.1. Danger Agents

A danger specialist is an element that represents a danger since it is fit for doing an assault. Cloud security dangers can begin either inside or remotely, from people or programming programs. Comparing danger operators are depicted in the up and coming areas. Figure 2.3 shows the job a danger operator accept comparable to vulnerabilities, dangers, and dangers, and the shields built up by security approaches and security systems.

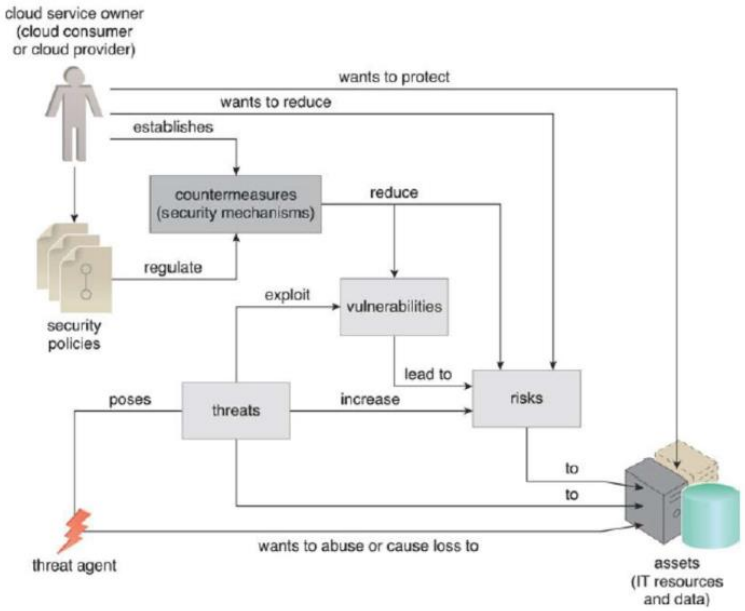


Figure 2.3.: How security policies mechanisms are used to counter threats vulnerabilities, and risks caused by threat agents.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

2.2.2. Unknown Attacker

An unknown assailant is a non-believed cloud administration buyer without authorizations in the cloud. It regularly exists as an outer programming program that dispatches organize level assaults through open systems. At the point when mysterious aggressors have restricted data on security approaches and protections, it can restrain their capacity to detail successful assaults. In this manner, unknown assailants frequently resort to submitting acts like bypassing client records or taking client qualifications, while utilizing strategies that either guarantee namelessness or require considerable assets for arraignment.

Figure 2.2. The notation used for an anonymous attacker.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

2.2.2. Mysterious Attacker

A mysterious assailant is a non-believed cloud administration shopper without authorizations in the cloud. It ordinarily exists as an outer programming program that dispatches arrange level assaults through open systems. At the point when mysterious aggressors have restricted data on security arrangements and barriers, it can hinder their capacity to plan powerful assaults. Accordingly, mysterious aggressors regularly resort to submitting acts like bypassing client records or taking client certifications, while utilizing strategies that either guarantee namelessness or require generous assets for indictment.

Figure 2.5. The notation used for a malicious service agent.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

2.2.2. Confided in Attacker

A believed assailant shares IT assets in a similar cloud condition as the cloud purchaser and endeavors to misuse real accreditations to target cloud suppliers and the cloud inhabitants with we celebrate IT assets (Figure 2.6). In contrast to mysterious aggressors (which are non-believed), believed assailants as a rule dispatch their assaults from inside a cloud's trust limits by manhandling genuine qualifications or by means of the selection of touchy and private data. Confided in aggressors (otherwise called vindictive inhabitants) can utilize cloud-based IT assets for a wide scope of abuses, including the hacking of feeble validation forms, the breaking of encryption, the spamming of email accounts, or to dispatch basic assaults, for example, forswearing of administration crusades.

Figure 2.6. The notation that is used for a trusted attacker.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

2.2.5. Noxious Insider

Noxious insiders are human danger specialists following up for the benefit of or corresponding to the cloud supplier. They are ordinarily current or previous workers or outsiders with access to the cloud supplier's premises. This sort of danger specialist conveys enormous harm potential, as the malignant insider may have regulatory benefits for getting to cloud buyer IT assets.

Figure 2.7. The notation used for an attack originating from a workstation. The human symbol is optional.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

2.3. Cloud Security Threats

This segment presents a few normal dangers and vulnerabilities in cloud-based situations and portrays the jobs of the previously mentioned danger operators.

2.3.1. Traffic Eavesdropping

Traffic listening in happens when information being moved to or inside a cloud (normally from the cloud customer to the cloud supplier) is inactively captured by a vindictive assistance specialist for ill-conceived data gathering purposes (Figure 2.8). The point of this assault is to straightforwardly bargain the secrecy of the information and, conceivably, the classification of the connection between the cloud buyer and cloud supplier. Due to the uninvolved idea of the assault, it can all the more effectively go undetected for broadened timeframes.

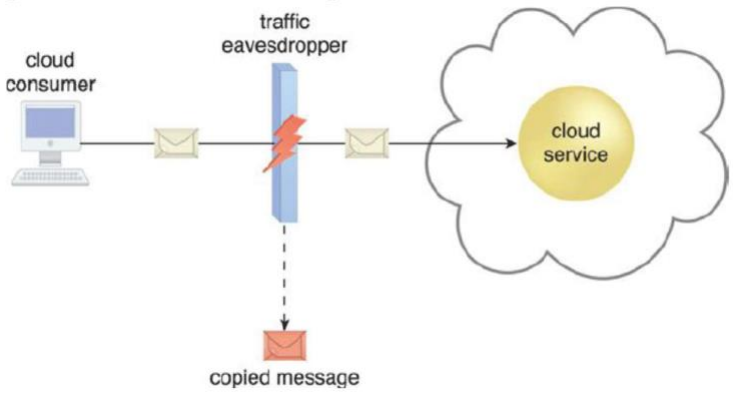


Figure 2.8.: An externally positioned malicious service agents carries out a traffic eavesdropping attack by intercepting a message sent by the cloud service consumer to the cloud service. The service agent makes an unauthorized copy of the message before it is sent along its original path to the cloud service.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

2.3.2. Malevolent Intermediary

The malevolent mediator danger emerges when messages are caught and changed by a pernicious help specialist, in this manner conceivably trading off the message's classification and additionally uprightness. It might likewise embed destructive information into the message before sending it to its goal. Figure 2.9. represents a typical case of the malevolent mediator assault.

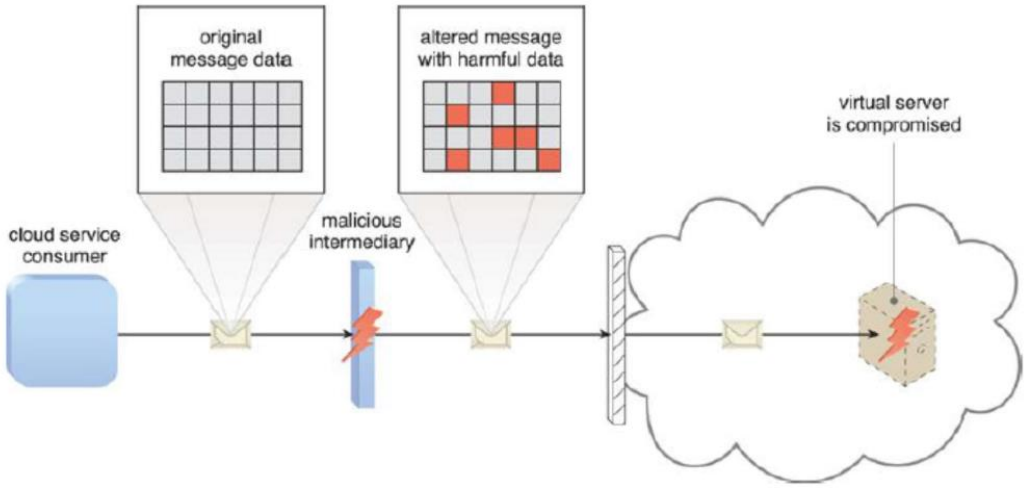


Figure 2.9.: The malicious service agent intercepts and modifies a message sent by a cloud service consumer to a cloud service(not shown) being hosted on a virtual server. Because harmful data is packaged into the message, the virtual server is compromised.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

2.3.3. Forswearing of Service

The goal of the disavowal of administration (DoS) assault is to over-burden IT assets to where they can't work appropriately. This type of assault is regularly propelled in one of the accompanying ways:

• The outstanding task at hand on cloud administrations is misleadingly expanded with impersonation messages or rehashed correspondence demands.

• The system is over-burden with traffic to lessen its responsiveness and challenged person its presentation.

• Multiple cloud administration demands are sent, every one of which is intended to expend unreasonable memory and handling assets.

Effective DoS assaults produce server debasement as well as disappointment, as delineated in Figure 2.10.:

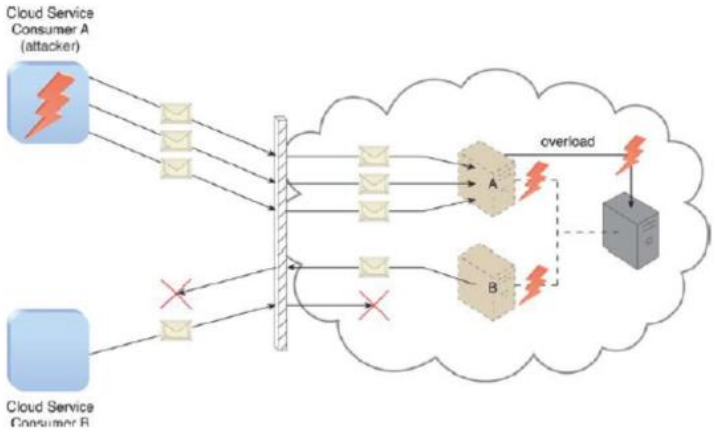


Figure 2.10.: Cloud service consumer A sends multiple message to a cloud service(not shown) hosted on virtual server A. This overloads the capacity of the underlying physical server, which causes outages with virtual servers A and B. As a result legitimate cloud service consumers, such as cloud service consumer B. become unable to communicate with any cloud services hosted on virtual servers A and B.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

2.3.3. Refusal of Service

The target of the refusal of administration (DoS) assault is to over-burden IT assets to where they can't work appropriately. This type of assault is normally propelled in one of the accompanying ways:

• The remaining task at hand on cloud administrations is falsely expanded with impersonation messages or rehashed correspondence demands.

• The system is over-burden with traffic to decrease its responsiveness and handicapped person its exhibition.

• Multiple cloud administration demands are sent, every one of which is intended to expend over the top memory and preparing assets.

Fruitful DoS assaults produce server corruption as well as disappointment, as represented in Figure 2.10.:

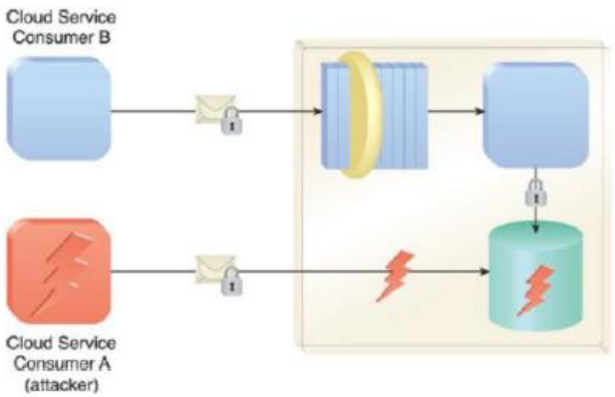


Figure 2.11. : Cloud service consumer A gains access to a database that was implemented under the assumption that it would only be accessed through a web service with a published service contract (as per cloud service consumer B)

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

A variety of this assault, known as feeble confirmation, can result when frail passwords or shared records are utilized to secure IT assets. Inside cloud situations, these kinds of assaults can prompt noteworthy effects relying upon the scope of IT assets and the scope of access to those IT assets the assailant gains.

2.3.5. Virtualization Attack

Virtualization furnishes different cloud shoppers with access to IT assets that share basic equipment however are coherently disconnected from one another. Since cloud suppliers award cloud buyers regulatory access to virtualized IT assets, (for example, virtual servers), there is an inalienable hazard that cloud purchasers could manhandle this entrance to assault the basic physical IT assets. A virtualization assault misuses vulnerabilities in the virtualization stage to risk its secrecy, honesty, as well as accessibility. This danger is outlined in Figure 2.12, where a believed aggressor effectively gets to a virtual server to bargain its hidden physical server. With open mists, where a solitary physical IT asset might be giving virtualized IT assets to various cloud buyers, such an assault can have noteworthy repercussions.

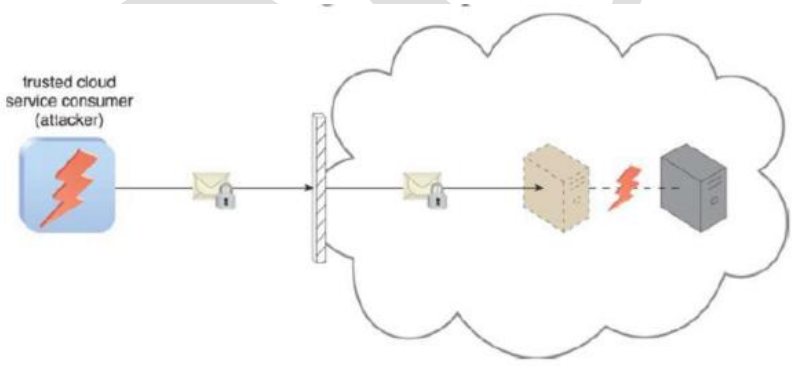


Figure 2.12.: An authorized cloud service consumer carries out a virtualization attack by abusing its administrative access to a virtual server to exploit the underlying hardware.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

2.3.6. Covering Trust Boundaries

In the event that physical IT assets inside a cloud are shared by various cloud administration shoppers, these cloud administration purchasers have covering trust limits. Malevolent cloud administration customers can target imparted IT assets to the aim of bargaining cloud shoppers or other IT assets that share a similar trust limit. The outcome is that a few or the entirety of the other cloud administration shoppers could be affected by the assault as well as the assailant could utilize virtual IT assets against others that happen to likewise have a similar trust limit.

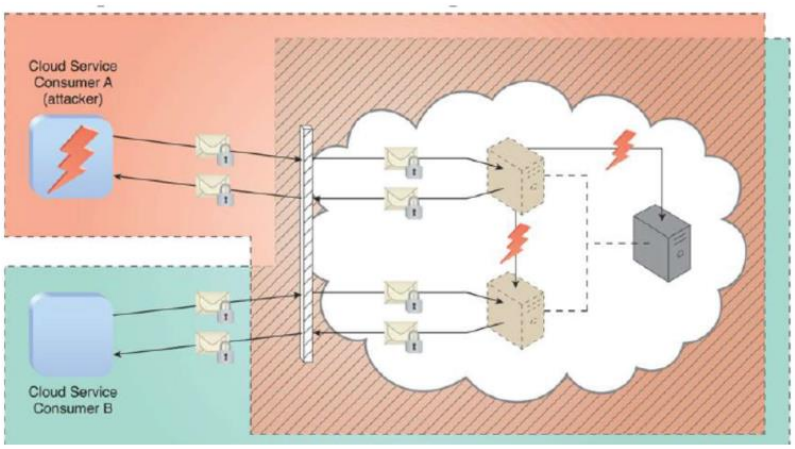


Figure 2.13 illustrates an example in which two cloud service consumers share virtual servers hosted by the same physical server and, resultantly, their respective trust boundaries overlap.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

2.3.7. Hazard Management

While evaluating the expected effects and difficulties relating to cloud reception, cloud shoppers are urged to play out a conventional hazard appraisal as a major aspect of a hazard the executives procedure. A consistently executed procedure used to upgrade key and strategic security, chance administration is involved a lot of facilitated exercises for supervising and controlling dangers. The fundamental exercises are commonly characterized as hazard appraisal, chance treatment, and hazard control (Figure 2.12).

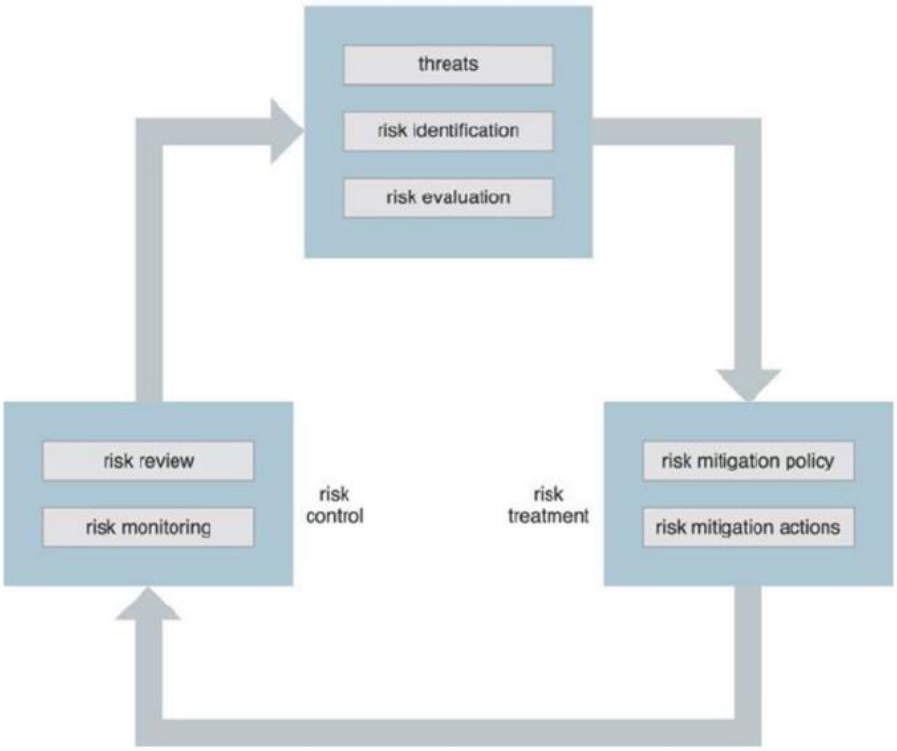


Figure 2.12.: The on-going risk management process, which can be initiated from any of the three stages.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

• **Risk Treatment** – Mitigation arrangements and plans are structured during the hazard treatment stage with the expectation of effectively rewarding the dangers that were found during hazard evaluation. A few dangers can be dispensed with, others can be moderated, while others can be managed through re-appropriating or even joined into the protection as well as working misfortune financial plans. The cloud supplier itself may consent to accept accountability as a feature of its legally binding commitments.

• **Risk Control –** The hazard control stage is identified with chance checking, a three-advance procedure that is included studying related occasions, surveying these occasions to decide the adequacy of past appraisals and medicines, and recognizing any strategy modification needs. Contingent upon the idea of the checking required, this stage might be done or shared by the cloud supplier.

2.2. Mechanical Platforms and New Developments:

Advancement of a distributed computing application occurs by utilizing stages and structures that give various sorts of administrations, from the exposed metal foundation to adaptable applications filling explicit needs.

2.5. Amazon web administrations (AWS)

One of the utmost much-admired and enormous traffic sites is the Amazon.com which offers a tremendous determination of items utilizing framework based web administration. This organization was begun in the year 2006 with the accessible stage on web-administration for designers on a use premise model. This organization brings the best case of web-administration accomplished through the administration situated engineering.

Amazon Web Service is a valuable of Amazon.com. Amazon has made it conceivable to create private virtual servers that can run overall through 'Equipment Virtualization' on Xen hypervisor. These servers can be provisioned with various kinds of utilization programming that client may foresee alongside a scope of help benefits that makes distributed computing applications conceivable as well as make them solid to withstand calculation.

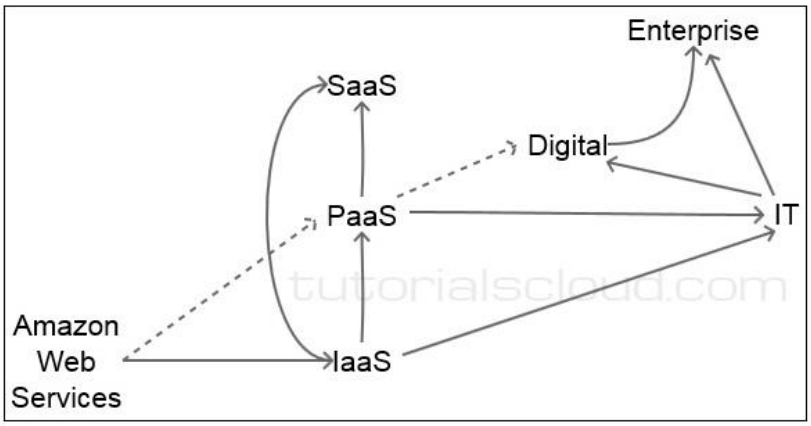


Figure 2.15.

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

In view of SOA standard, and SOAP, REST and HTTP move conventions, moreover open - source and business OS, program based programming and application servers are running by Amazon Web Service. AWS offers different set-ups of Cloud registering innovation that makes up an on-request computational stage. These administrations get worked from twelve distinctive geological areas and among them, the most popular is the Amazon's Elastic Compute Cloud (EC2) and Amazon's Simple Storage Service (S3).

2.5.1. Recommendation of AWS

AWS has a tremendous offer. Just clients need to pay that they use, which can spare a lot of cash. AWS has in extra of seventy administrations including capacity, figure, database, organizing, application administration, versatile, the executives, engineer's devices and IoT.

2.5.2. Downplaying Amazon Web Services

It is the world's biggest online retailer. Before Amazon.com, the world's greatest retailer was Wal - Mart. As per the yearly report of the year 2009, the net offer of Amazon is $22.51 billion. It has an immense business, and for this, it has fabricated a colossal system of IT frameworks for supporting. AWS basically takes Amazon.com system's truly beneficial business bringing a huge measure of income.

AWS has an enormous power and effect in cloud innovation, giving the biggest Infrastructure as a Service (IaaS) commercial center.

2.5.3. Segment and Web Services of AWS

The Amazon's web administrations have the accompanying segments:

- Amazon Elastic Compute Cloud: (EC2; http://aws.amazon.com/ec2/) is the incorporated use of AWS which encourages the administration and use of virtual private servers that can run on Windows and Linux-based stages over Xen Hypervisor. Various devices are utilized to help Amazon's web administrations. These are:

o Amazon Simple Queue Service is a message line and exchange framework for dispersed Internet-based applications.
o Amazon Simple Notification Service is utilized to distribute message from an application.
o Amazon CloudWatch is utilized for observing EC2 Cloud which underpins by giving console or order line perspective on assets in use.
o Elastic Load Balancing is utilized to identify whether an occurrence is falling flat or check whether the traffic is sound or not.
• Amazon's Simple Storage Service: is an online stockpiling and reinforcement framework which has rapid information move strategy called AWS Import/Export.
Additional web - services mechanisms are:

- Amazon's Elastic Block Store
- Amazon's Simple Database (DB)
- Amazon's Relational Database Service
- Amazon Cloudfront

A large number of services and utilities also support Amazon partners, i.e., the AWS infrastructure itself. These are:

- Alexa Web Information Service
- Amazon Associates Web Services (A2S)
- Amazon DevPay
- Elastic Map-Reduce
- Amazon's Mechanical Turk
- AWS Multi-factor Authentication
- Amazon's Flexible payment Service (FPS)
- Amazon's Fulfillment Web-Service (FWS)
- Amazon Virtual Private Cloud

2.5.2. Flexible Cloud Compute

It is a virtual server stage permitting clients o make and run virtual machines on Amazon Server firm. The Amazon Machine Images (AMI) is used by EC2 to communication and run server samples for running working frameworks like: Linux (Red-Hat), Windows, and so on various servers. As the name proposes, we can include or take away flexibly as and when required; reproduce and load-adjusted servers. We can likewise find our servers in various zones all through the globe to give adaptation to non-critical failure.

The term 'versatile' characterizes the capacity to resize your ability rapidly varying. Executing a help may require the accompanying segments:

- Application server (having large RAM allocation)
- A load balancer
- Database server
- Firewall and network switches
- Additional rack capacity

2.6. Google App Engine

Google App Engine permits us to run (have) your own Web applications on Google's foundation. Be that as it may, in no method, figure or form is this a "lease a bit of a server" facilitating administration. With App Engine, your application isn't facilitated on a solitary server. There are no servers to keep up: You simply transfer your application, and it's prepared to serve your clients. Similarly as adjusting a Google search solicitation may include handfuls, or even many Google servers, all completely covered up and fulfilled in a small amount of a second, Google App Engine applications run a similar way, on a similar framework. This is the novel part of Google's methodology. Truly, you surrender some control to Google, yet you are compensated by being absolutely liberated from the foundation, limit the executives, and burden adjusting assignments that undertaking normally need to oversee, independent of whether they are self-facilitating or facilitating on another person's PaaS or IaaS.

You can decide to impart your application to the world, or limit access to individuals from your association. Google App Engine bolsters applications written in a few program design vernaculars:

With App Engine's Java runtime condition, you can manufacture your application utilizing standard Java inventions, counting the JVM, Java servlets, and the Java program design language—or some other language utilizing a JVM-based mediator or compiler, for example, JavaScript or Ruby. Application Engine likewise includes a committed Python runtime condition, which incorporates a quick Python translator and the Python standard library. The Java and Python runtime situations are worked to guarantee that your application runs rapidly, safely, and without impedance from different applications on the framework.

Likewise with most cloud-facilitating administrations, with App Engine, you just compensation for what you use. Google demands no set-up costs and no repetitive charges. Like Amazon's AWS, assets, for example, stockpiling and data transfer capacity are estimated by the gigabyte.
Application Engine costs nothing to begin. All applications can utilize around 500 MB of capacity and sufficient CPU and data transmission to help a proficient application portion round 5 million site visits a month, totally free. At the point when you empower charging for your application, your free cutoff points are raised, and you just compensation for assets you use over the free levels.

Application designers approach tireless capacity innovations, for example, the Google File System (GFS) and Bigtable, a circulated stockpiling framework for unstructured information. The Java rendition underpins offbeat nonblocking inquiries utilizing the Twig Object Datastore interface. This offers an option in contrast to utilizing strings for equal information preparing.

"With Google App Engine, designers can compose Web applications dependent on a similar structure hinders that Google utilizes," Kevin Gibbs, Google's specialized lead for the undertaking, wrote in The Official Google Blog"Google. Twig is an item industriousness interface based on Google App Engine's low-level datastore which defeats a considerable lot of JDO-GAEs impediments, including full help for legacy, polymorphism, and conventional sorts. You can without much of a stretch design, change or broaden Twig's conduct by actualizing your own systems or superseding augmentation focuses in unadulterated Java

code. Application Engine bundles those structure squares and gives access to adaptable foundation that we expectation will make it simpler for designers to scale their applications naturally as they develop."

Google App Engine has showed up when an expanding number of tech organizations are moving their tasks to the cloud; it places Google solidly in rivalry with Amazon's Elastic Cloud Computing (EC2) and Simple Storage Service (S3) contributions.

Google says its vision with Google App Engine is to offer designers a progressively all encompassing, start to finish answer for building and scaling applications on the web. Its servers are designed to adjust the heap of traffic to engineers' applications, scaling to satisfy the need of a flood of traffic. Application Engine likewise incorporates APIs for client validation to permit designers to sign on for administrations, and for email, to oversee correspondences.

InternetNews.com detailed,

Through its underlying review, Google's App Engine will be accessible allowed to the initial 10,000 designers who sign up, with plans to extend that number in the future.During that period, clients will be constrained to 500MB of capacity, 10GB of every day transfer speed and 5 million day by day site hits, the organization said. Engineers will have the option to enroll up to three applications.

2.7. Microsoft Azure

it gives a wide assortment of administrations which cloud clients can use without buying your own equipment. It empowers fast improvement of arrangements and giving the assets to that condition. Without the need to stress over gathering of physical foundation, Azure's figure, Azure's process, system, stockpiling and application administrations permit clients to concentrate on building incredible arrangements.

Azure Services

Azure Services remembers different administrations for its cloud innovation. These are:

1.Compute Services: This holds MS Azure administrations, for example, Azure VM, Azure Website, Mobile administrations and so forth.

2.Data Services: It incorporates MS Azure Storage, Azure SQL database, and so on.

3.Application Services: It Includes those administrations that causes clients to fabricate and work applications, for example, Azure Active Directory, Service transport for associating appropriated frameworks, preparing enormous information and so forth.

2.Network Services: It incorporates Virtual system, content conveyance system and Traffic administrator of Azure.
There are other services such as:

- BizTalk
- Big Compute
- Identity

- Messaging
- Media
- CDN etc...

2.7.1. More on MS Cloud

The beginning zone for Microsoft's Cloud innovation endeavors might be found at Microsoft.com/cloud. It has an immense scope of cloud innovation items and some driving Web-utilizations of the business. Microsoft Messenger turned into the market chief after America Online Instant Messenger (AIM). Progressively with the ascent of e-office and advertising field, Microsoft considers its to be as giving best Web understanding for various kinds of gadgets, for example, PCs, work areas, PCs, tablets, advanced lockups, and so out.

2.7.2. Purplish blue Virtual Machines

It is one of the unified highlights of IaaS ability of MS Azure along with virtual system. Purplish blue's VM support the improvement of Windows Server (or Linux VM) in MS Azure's datacenter; where you have unlimited oversight over the Virtual machine's setup. Sky blue's VM has three potential states:
•Running
•Stopped
•Stopped (De-designated)

The VM gets the stop (de-assigned) state as a matter of course when it is halted in the Azure Management Portal. On the off chance that we need to keep it halted just as apportioned we need to utilize PowerShell cmdlet with the accompanying order:

```
> Stop-AzureVM -Name "az-essential" -ServiceName "az-
essential" -StayProvisioned
```

2.7.3. Element of Microsoft Azure

There are 6 main elements that form Windows Azure. These are:

- Compute
- Storage
- Application
- Fabric
- VM (Virtual Machines)
- Config (Configuration)

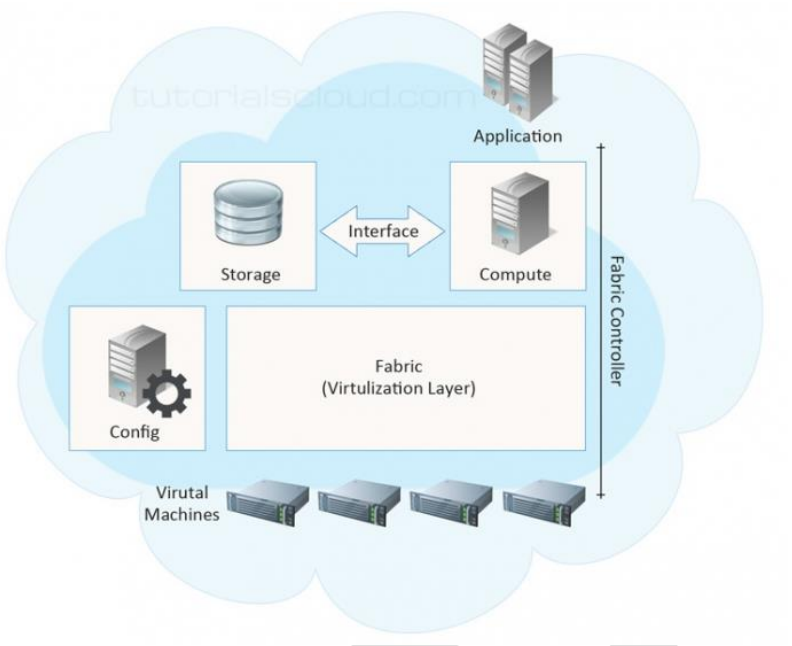


Figure 2.16. - Elements of Microsoft Azure:

(Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini)

2.7.2. Access Control of MS Cloud

It permits an application to believe the character of another application and this strategy can meet up with personality suppliers, for example, ADFS to make conveyed frameworks dependent on SOA. The means required for Access Control are:

•The customer/client sends demand for verification from AC (Access Control)
• Access Control creates a token dependent on put away guidelines for server application
• The token is marked and come back to customer application
• The customer present the got token to the administration application
• The check of the mark is done at long last and uses a token to pick whether cloud application is permitted or not.

Reference : Cloud Computing(Concepts, Technology & Architecture) by Thomas Erl,Zaigham Mahmood, and Ricardo Puttini

CHAPTER-1
SPECIALIZED CLOUD MECHANISMS

Structure:

3.1.1 Objective:

Collective to offer different and tradition architecture. This mechanism is a service agent that monitors and tracks communications between cloud service consumers and cloud services for dynamic scaling purposes. This way, the cloud consumer can choose to adjust its current IT resource allocation.

3.1.2 Introduction:

A typical cloud technology architecture contains numerous moving parts to address distinct usage requirements of IT resources and solutions. Each mechanism covered in this chapter fulfills a specific runtime function in support of one or more cloud characteristics.

3.1.3 Automated Scaling Listener:

The automated scaling listener mechanism is a service agent that monitors and tracks communications between cloud service consumers and cloud services for dynamic scaling purposes. Automated scaling listeners are deployed within the cloud, typically near the firewall, from where they automatically track workload status information.

Workloads can be determined by the volume of cloud consumer-generated requests or via back-end processing demands triggered by certain types of requests. For example, a small amount of incoming data can result in a large amount of processing.

Automated scaling listeners can provide different types of responses to workload fluctuation conditions, such as:

- Automatically scaling IT resources out or in based on parameters previously defined by the cloud consumer (commonly referred to as **auto-scaling**).
- Automatic notification of the cloud consumer when workloads exceed current thresholds or fall below allocated resources. This way, the cloud consumer can choose to adjust its current IT resource allocation. **(auto-notification**)

Different cloud provider vendors have different names for service agents that act as automated scaling listeners.

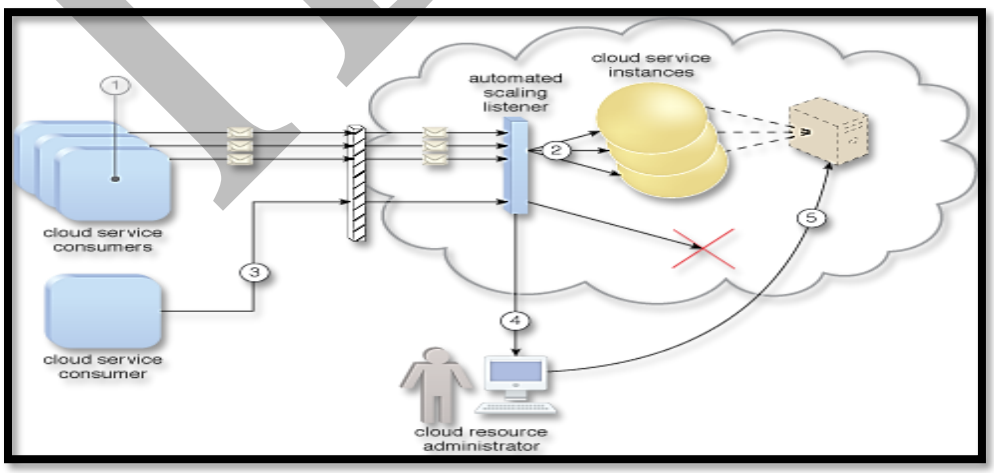


Fig 3.1.3

Three cloud service consumers attempt to access one cloud service simultaneously **(1).** The automated scaling listener scales out and initiates the creation of three redundant instances of the service **(2).** A fourth cloud service consumer attempts to use the cloud service **(3).** Programmed to allow up to only three instances of the cloud service, the automated scaling listener rejects the fourth attempt and notifies the cloud consumer that the requested workload limit has been exceeded **(4).** The cloud consumer's cloud resource administrator accesses the remote administration environment to adjust the provisioning setup and increase the redundant instance limit **(3.1).**

3.1.3.1 Case of DTGOV

The virtualization platform is configured to automatically scale a virtual server at runtime, as follows:

A. **Scaling-Down –** The virtual server continues residing on the same physical host server while being scaled down to a lower performance configuration.
B. **Scaling-Up –** The virtual server's capacity is doubled on its original physical host server. The VIM may also live migrate the virtual server to another physical server if the original host server is overcommitted. Migration is automatically performed at runtime and does not require the virtual server to shut down.
    1. A cloud consumer creates and starts a virtual server with 8 virtual processor cores and 16 GB of virtual RAM (1).
    2. The VIM creates the virtual server at the cloud service consumer's request, and the corresponding virtual machine is allocated to Physical Server 1 to join 3 other active virtual machines (2).
    3. Cloud consumer demand causes the virtual server usage to increase by over 80% of the CPU capacity for 60 consecutive seconds (3).
    4. The automated scaling listener running at the hypervisor detects the need to scale up and commands the VIM accordingly (4).

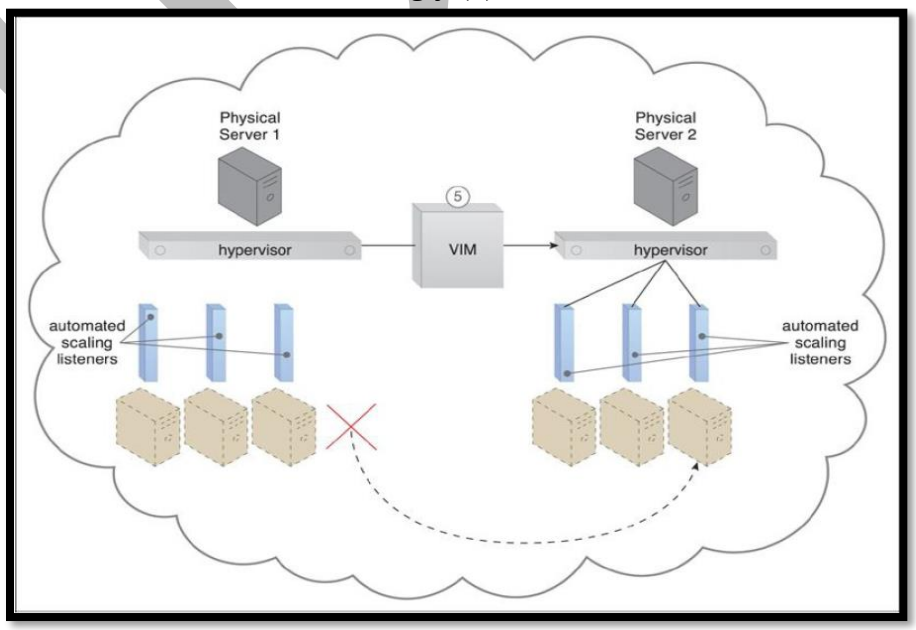


Fig 3.1.3.1

In Fig 3.1.3.1 the VIM determines that scaling up the virtual server on Physical Server 1 is not possible and proceeds to live migrate it to Physical Server 2.

5. The virtual server's CPU/RAM usage remains below 13.1% capacity for 60 consecutive seconds (6).
6. The automated scaling listener detects the need to scale down and commands the VIM (7), which scales down the virtual server (8) while it remains active on Physical Server 2.

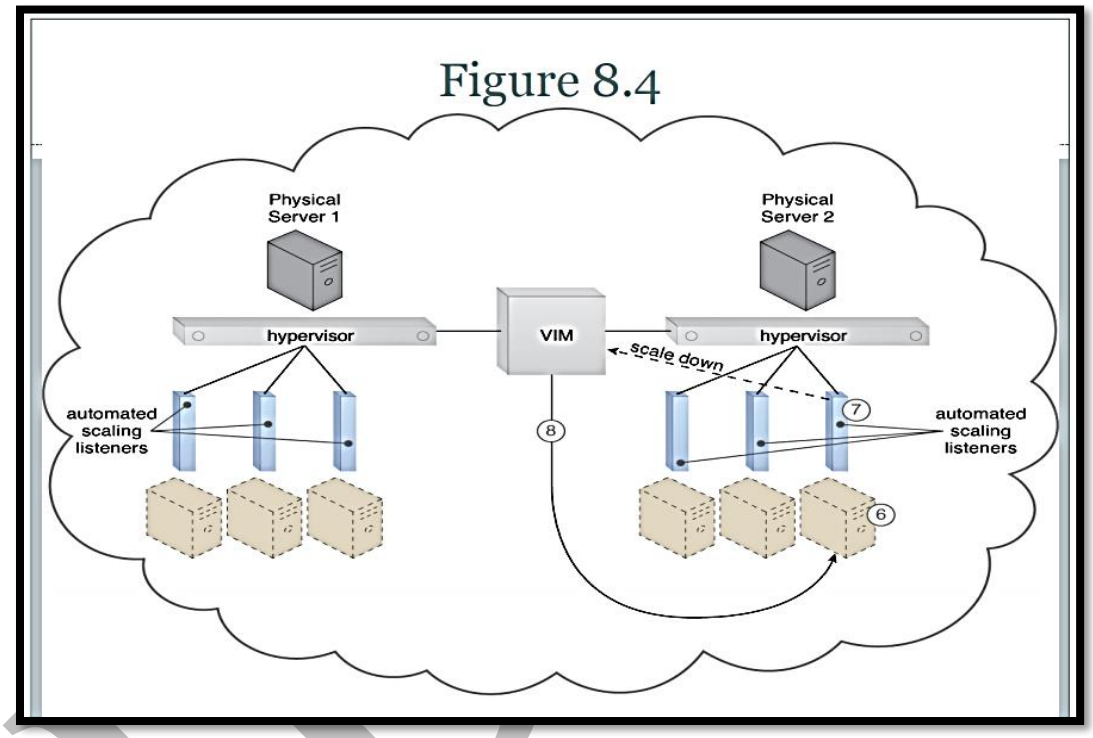


Fig 3.1.3.2

3.1.4 Load Balancer

Cloud **load-balancing** is the process of distributing workloads & computing resources within a cloud technology's environment. It also helps organizations & enterprises to manage workload demands by allocating resources among multiple systems or servers. Cloud load balancing also involves hosting the distribution of workload traffic that resides over the internet.

High performance level of tasks can be achieved using this load balancing technique on a lower cost than traditional on-premises load-balancing technology. In addition to workload and traffic distribution, cloud technology's load balancing can also provide health check for Cloud applications.

The **load balancer** mechanism is a runtime agent with logic fundamentally based on this premise. Load balancers can perform a range of specialized runtime workload distribution functions that include:

- **Asymmetric Distribution** – larger workloads are issued to IT resources with higher processing capacities

- **Workload Prioritization** – workloads are scheduled, queued, discarded, and distributed workloads according to their priority levels
- **Content-Aware Distribution** – requests are distributed to different IT resources as dictated by the request content

The **common objectives** of using load balancers are:

- To maintain system firmness.
- To improve system performance.
- To protect against system failures.

3.1.4.1 How does load balancing work?

Here, load refers to not only the website traffic but also includes CPU load, network load and memory capacity of each server. A load balancing technique makes sure that each system in the network has same amount of work at any instant of time. This means neither any of them is excessively over-loaded, nor under-utilized. [1]

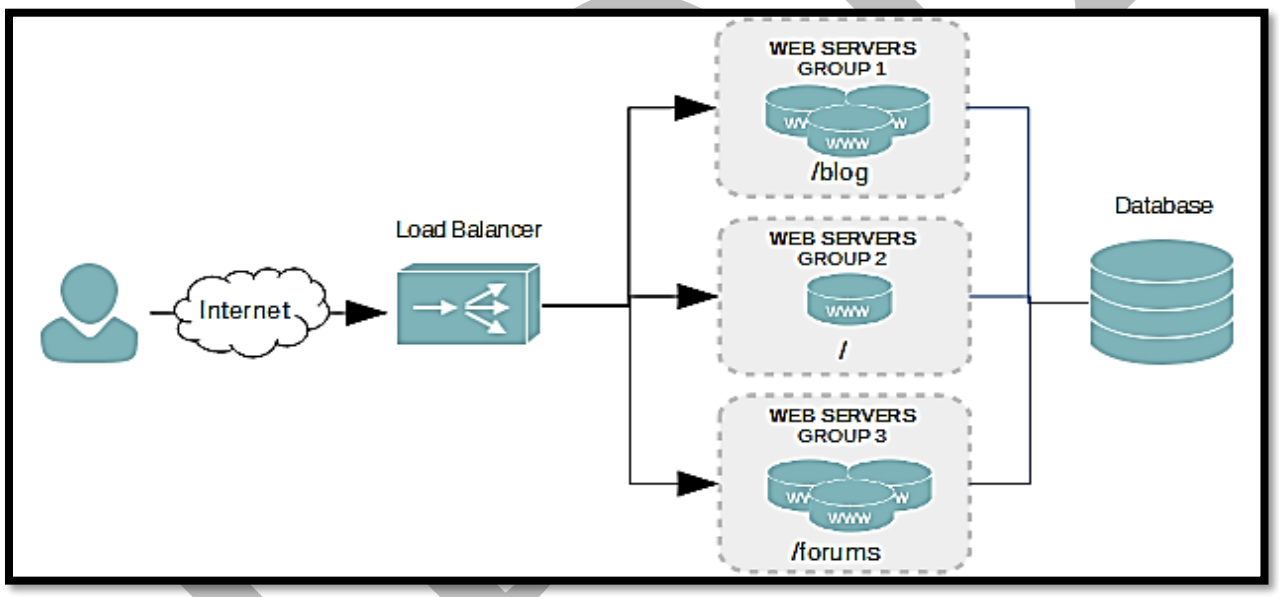


Fig 3.1.4.1

The load balancer mechanisms can exist as a:

- Multi-layer network switch
- Dedicated hardware appliance
- Dedicated software-based system
- Service agent

3.1.3.1 SLA Monitor:

The SLA management system mechanism represents a range of commercially available cloud management products that provide features pertaining to the administration, collection, storage, reporting, and runtime notification of SLA data.

An SLA management system deployment will generally include a repository used to store and retrieve collected SLA data based on pre-defined metrics and reporting parameters. It will further

rely on one or more SLA monitor mechanisms to collect the SLA data that can then be made available in near-real time to usage and administration portals to provide ongoing feedback regarding active cloud services **(Figure 3.1.3.1).** The metrics monitored for individual cloud services are aligned with the SLA guarantees in corresponding cloud provisioning contracts.

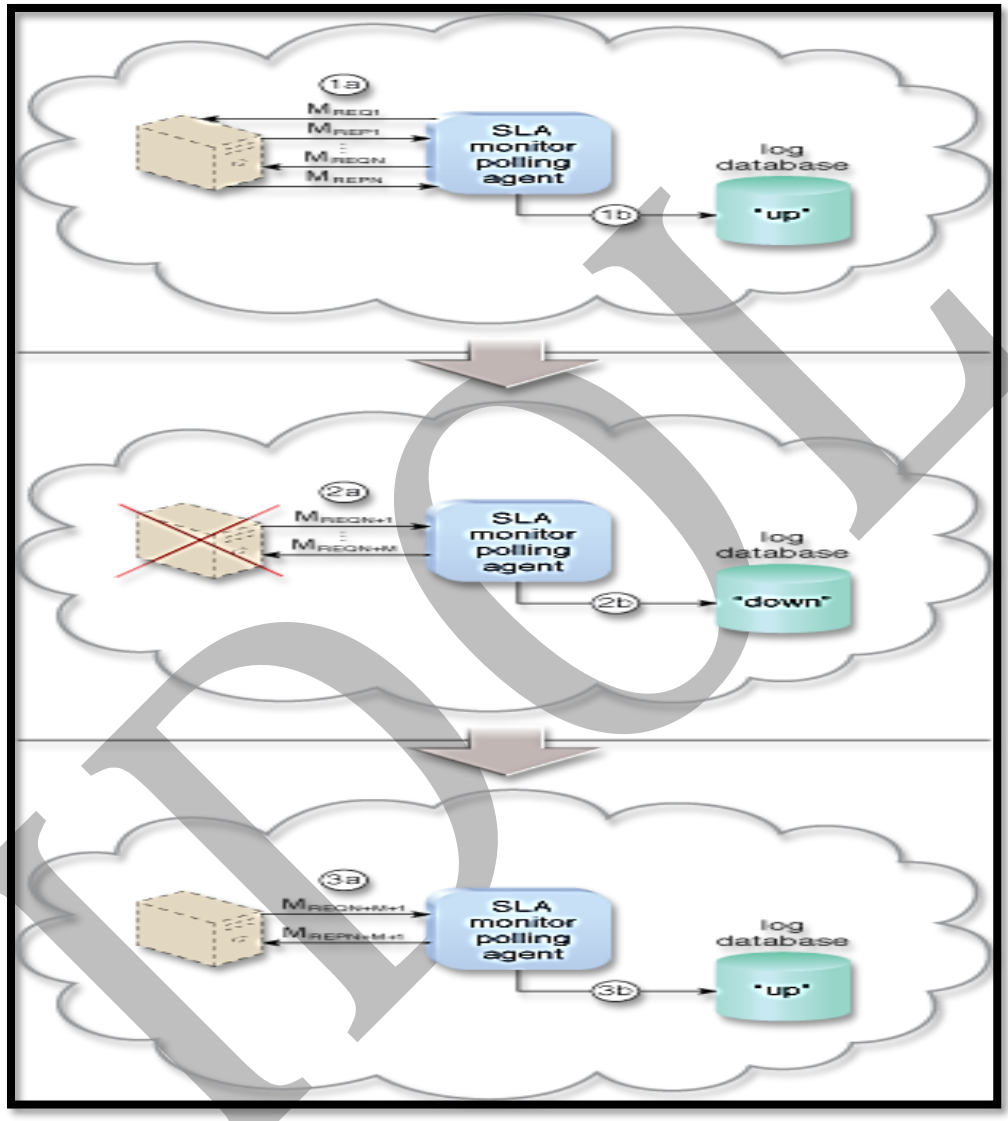


Fig 3.1.3.1

Figure 3.1.3.1 – A cloud service consumer interacts with a cloud service (1). An SLA monitor intercepts the exchanged messages, evaluates the interaction, and collects relevant runtime data in relation to quality-of-service guarantees defined in the cloud service's SLA (2A). The data collected is stored in a repository (2B) that is part of the SLA management system (3). Queries can be issued and reports can be generated for an external cloud resource administrator via a usage and administration portal (4) or for an internal cloud resource administrator via the SLA management system's native user-interface (3.1).

3.1.6 Pay-per-use monitor:

The pay-per-use monitor mechanism measures cloud-based IT resource usage in accordance with predefined pricing parameters and generates usage logs for fee calculations and billing purposes. Some typical monitoring variables are:

- request/response message quantity
- transmitted data volume
- bandwidth consumption

The data collected by the pay-per-use monitor is processed by a billing management system that calculates the payment fees.

Figure 3.1.6 shows a pay-per-use monitor implemented as a resource agent used to determine the usage period of a virtual server.

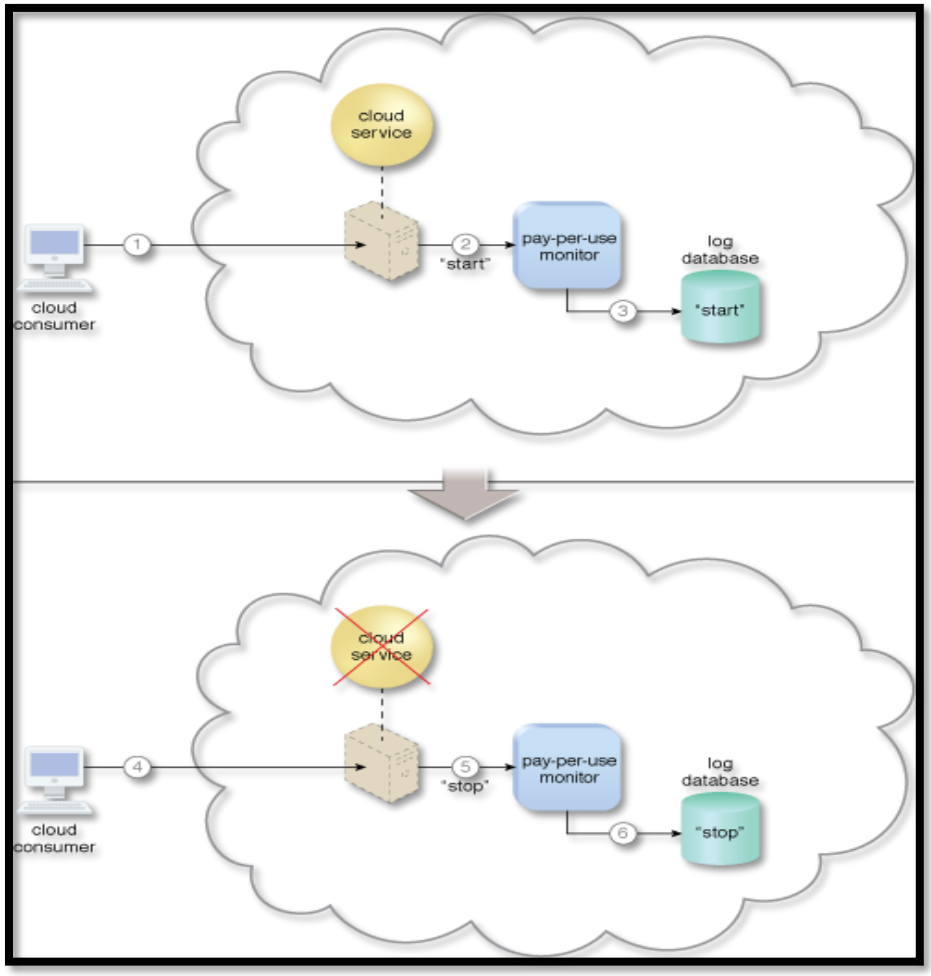


Fig 3.1.6.1

Figure 3.1.6.1 – A cloud consumer requests the creation of a new instance of a cloud service (1). The IT resource is instantiated and they pay-per-use monitor mechanism receives a "start" event notification from the resource software (2). The pay-per-use monitor stores the value timestamp in the log database (3). The cloud consumer later requests that the cloud service instance be stopped

(4). The pay-per-use monitor receives a "stop" event notification from the resource software (3.1) and stores the value timestamp in the log database (6).

Figure3.1.6 illustrates the pay-per-use monitor designed as a monitoring agent that transparently intercepts and analyzes runtime communication with a cloud service.

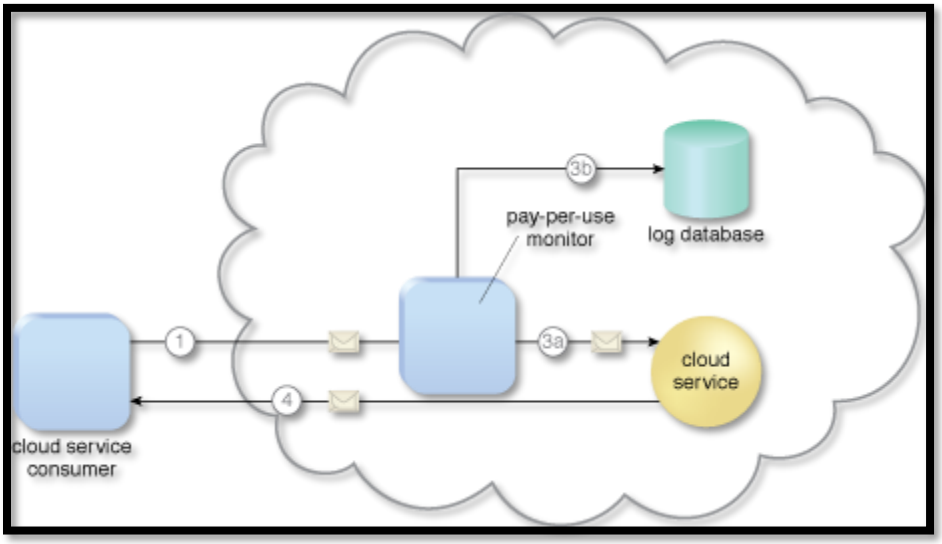


Fig 3.1.6.2

Figure 3.1.6.2 – A cloud service consumer sends a request message to the cloud service (1). The pay-per-use monitor intercepts the message (2), forwards it to the cloud service (3a), and stores the usage information in accordance with its monitoring metrics (3b). The cloud service forwards the response messages back to the cloud service consumer to provide the requested service (4).

3.1.7 Audit monitor:

The audit monitor mechanism is used to collect audit tracking data for networks and IT resources in support of, or dictated by, regulatory and contractual obligations. The figure depicts an audit monitor implemented as a monitoring agent that intercepts "login" requests and stores the requestor's security credentials, as well as both failed and successful login attempts, in a log database for future audit reporting purposes.

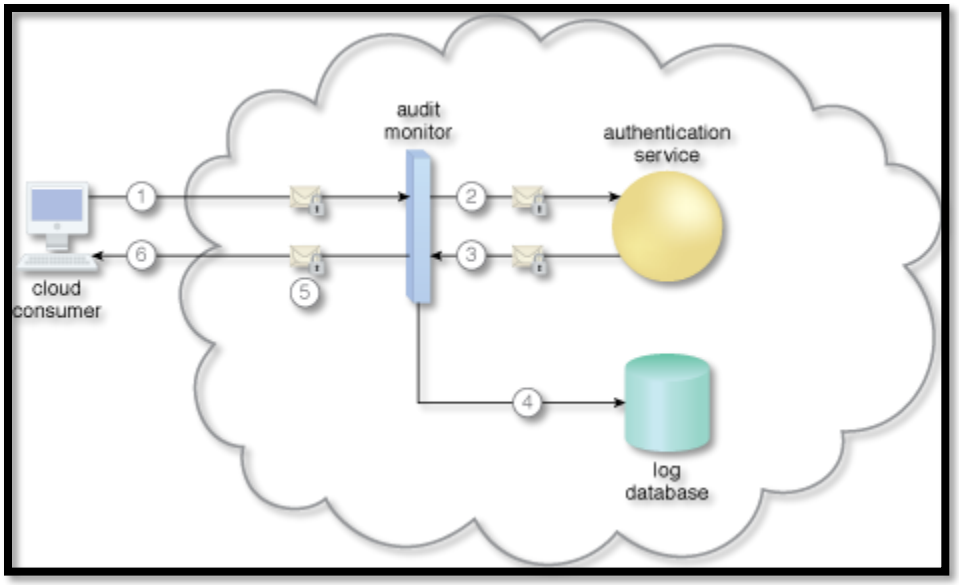


Fig 3.1.7

A cloud service consumer requests access to a cloud service by sending a login request message with security credentials (1). The audit monitor intercepts the message (2) and forwards it to the authentication service (3). The authentication service processes the security credentials. A response message is generated for the cloud service consumer, in addition to the results from the login attempt (4). The audit monitor intercepts the response message and stores the entire collected login event details in the log database, as per the organization's audit policy requirements (3.1). Access has been granted, and a response is sent back to the cloud service consumer (6).

3.1.8 Failover System:

The failover system mechanism is used to increase the reliability and availability of IT resources by using established clustering technology to provide redundant implementations. A failover system is configured to automatically switch over to a redundant or standby IT resource instance whenever the currently active IT resource becomes unavailable.

Failover systems are commonly used for mission-critical programs or for reusable services that can introduce a single point of failure for multiple applications. A failover system can span more than one geographical region so that each location hosts one or more redundant implementations of the same IT resource.

This mechanism may rely on the resource replication mechanism to supply the redundant IT resource instances, which are actively monitored for the detection of errors and unavailability conditions.

3.1.8.1 Failover systems come in two basic configurations:

A. Active-Active

In an active-active configuration, redundant implementations of the IT resource actively serve the workload synchronously (Figure 3.1.8.1). Load balancing among active instances is required. When a failure is detected, the failed instance is removed from the load balancing scheduler (Figure 3.1.8.2). Whichever IT resource remains operational when a failure is detected takes over the processing (Figure 3.1.8.3).

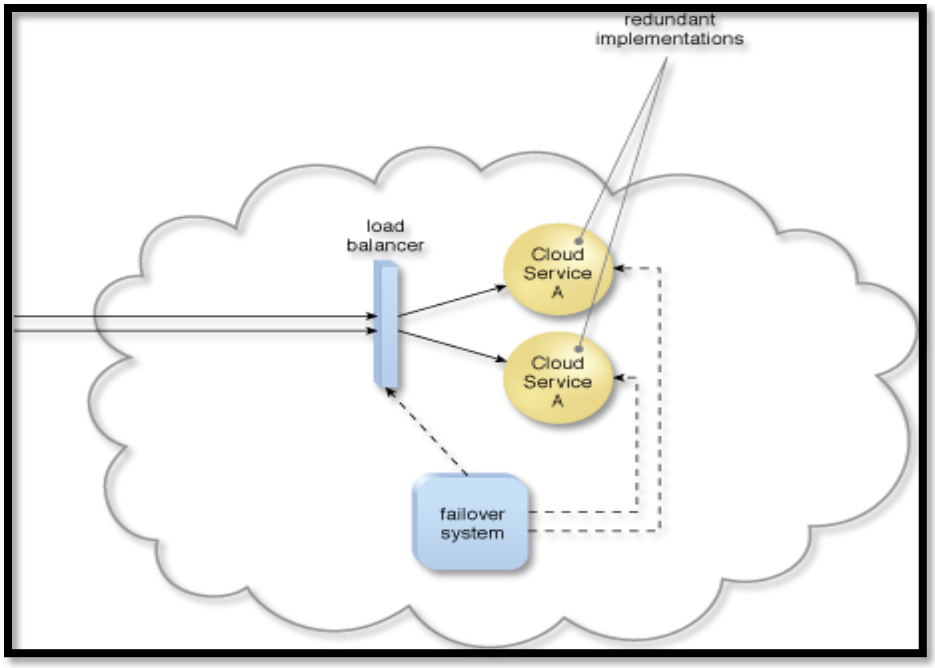


Fig 3.1.8.1

Figure 3.1.8.1 – The failover system monitors the operational status of Cloud Service A.

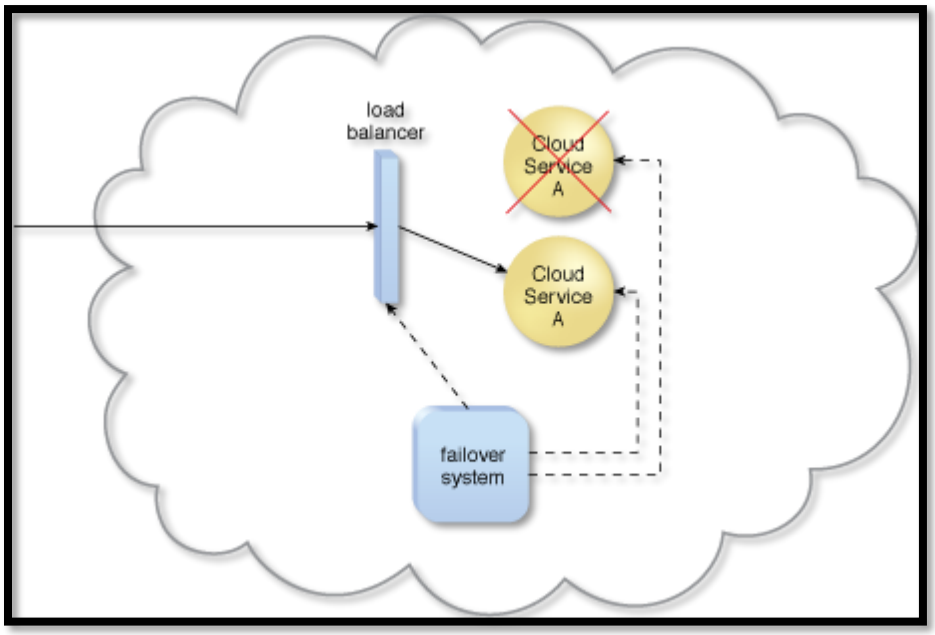


Fig 3.1.8.2

Figure 3.1.8.2 – When a failure is detected in one Cloud Service A implementation, the failover system commands the load balancer to switch over the workload to the redundant Cloud Service A implementation.

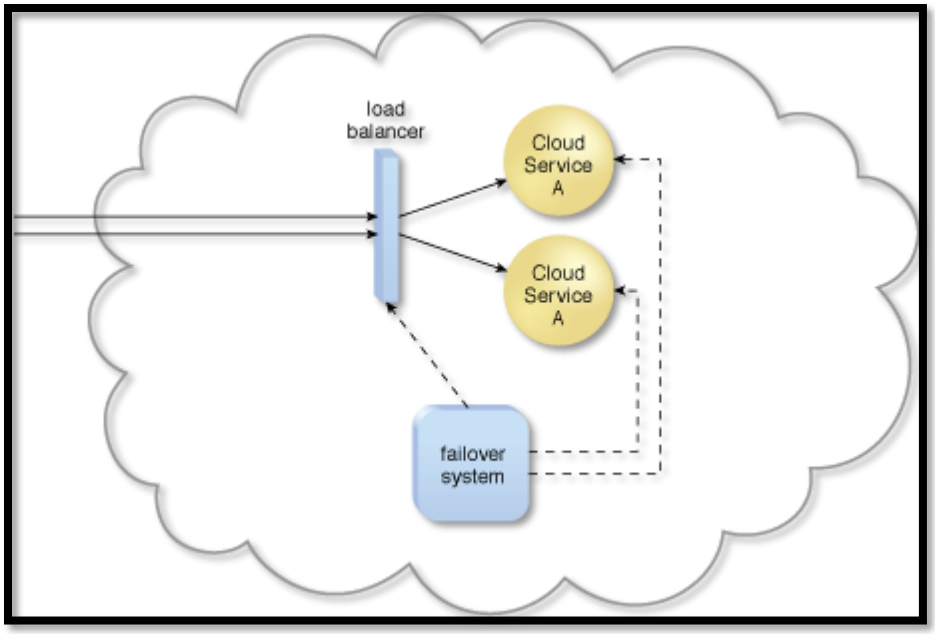


Fig 3.1.8.3

Figure 3.1.8.3 – The failed Cloud Service A implementation is recovered or replicated into an operational cloud service. The failover system now commands the load balancer to distribute the workload again.

B. Active-Passive

In an active-passive configuration, a standby or inactive implementation is activated to take over the processing from the IT resource that becomes unavailable, and the corresponding workload is redirected to the instance taking over the operation (Figures 4 to 3.1).

Some failover systems are designed to redirect workloads to active IT resources that rely on specialized load balancers that detect failure conditions and exclude failed IT resource instances from the workload distribution. This type of failover system is suitable for IT resources that do not require execution state management and provide stateless processing capabilities. In technology architectures that are typically based on clustering and virtualization technologies, the redundant or standby IT resource implementations are also required to share their state and execution context. A complex task that was executed on a failed IT resource can remain operational in one if it's redundant implementations.

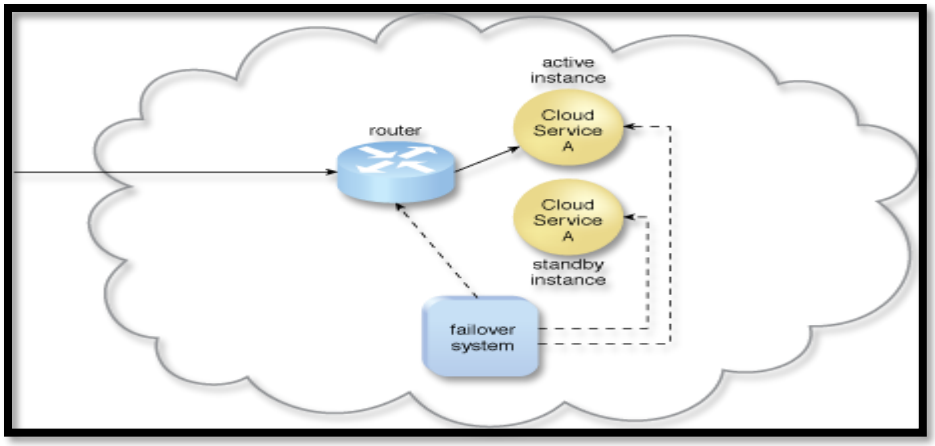


Fig 3.1.8.4

Figure 3.1.8.4 – The failover system monitors the operational status of Cloud Service A. The Cloud Service A implementation acting as the active instance is receiving cloud service consumer requests.

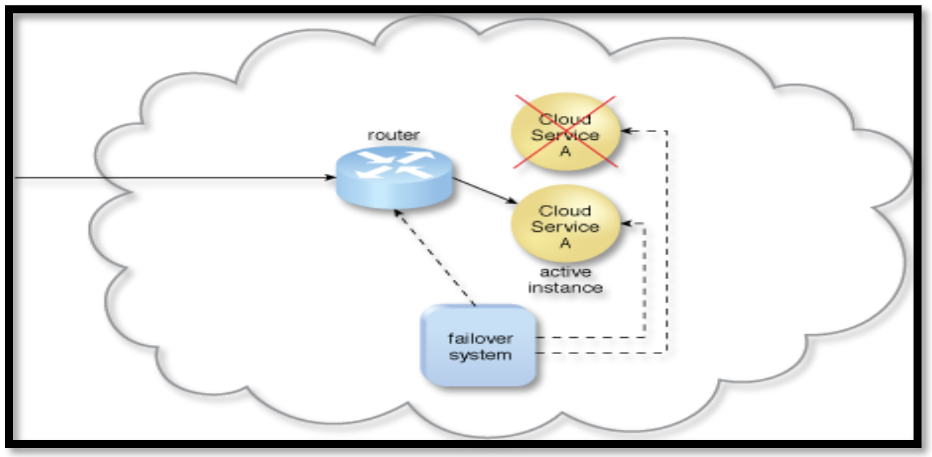


Fig 3.1.8.3.1

Figure 3.1.8.3.1 – The Cloud Service An implementation acting as the active instance encounters a failure that is detected by the failover system, which subsequently activates the inactive Cloud Service A implementation and redirects the workload toward it. The newly invoked Cloud Service A implementation now assumes the role of active instance.

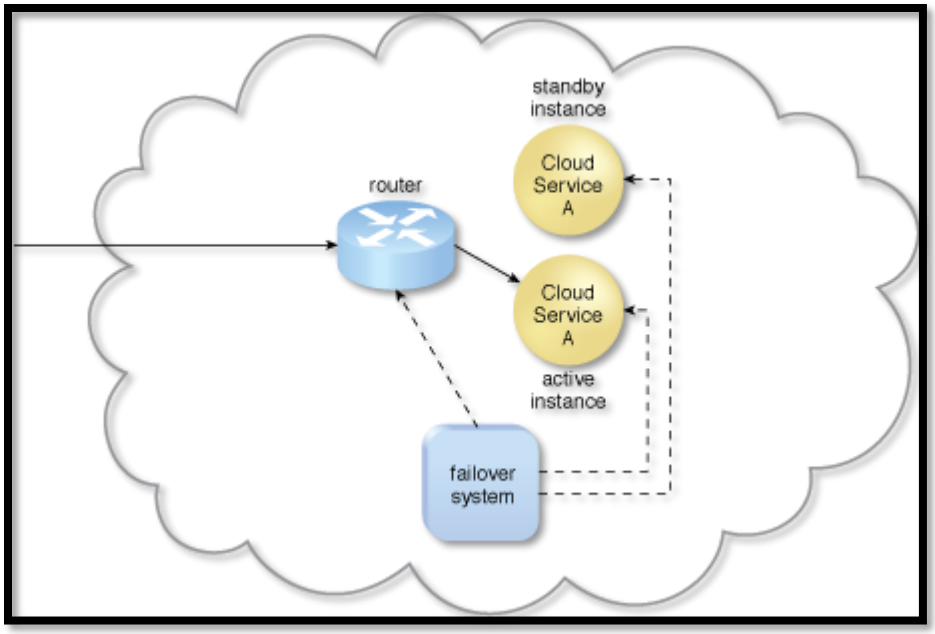


Fig 3.1.8.6

Figure 3.1.8.6 – The failed Cloud Service A implementation is recovered or replicated into an operational cloud service, and is now positioned as the standby instance, while the previously invoked Cloud Service A continues to serve as the active instance.

3.1.9 Hypervisor

A hypervisor is a hardware virtualization technique that allows multiple guest operating systems (OS) to run on a single host system at the same time. The guest OS shares the hardware of the host computer, such that each OS appears to have its own processor, memory and other hardware resources.

A hypervisor is also known as a virtual machine manager (VMM).

The hypervisor isolates the operating systems from the primary host machine. The job of a hypervisor is to cater to the needs of a guest operating system and to manage it efficiently. Each virtual machine is independent and do not interfere with each another although they run on the same host machine.[2]

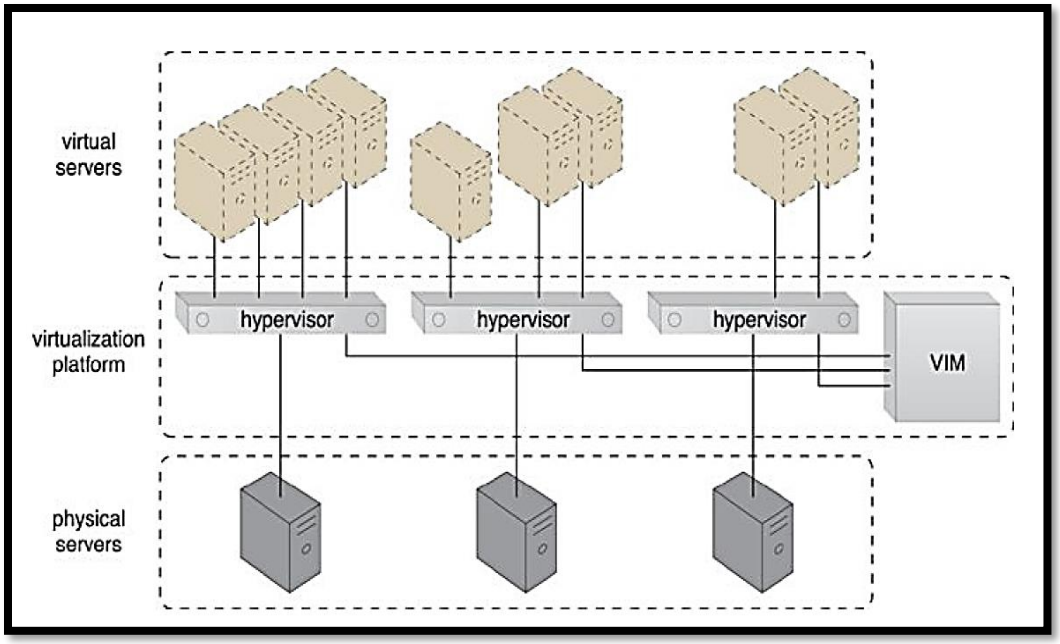


Fig 3.1.9

3.1.9.1 Hypervisors Are Divided Into Two Types:

A. Type one is the bare-metal hypervisor that are deployed directly over the host's system hardware without any underlying operating systems or software. Some examples of the type 1 hypervisors are Microsoft Hyper-V hypervisor, VMware ESXi, Citrix XenServer.

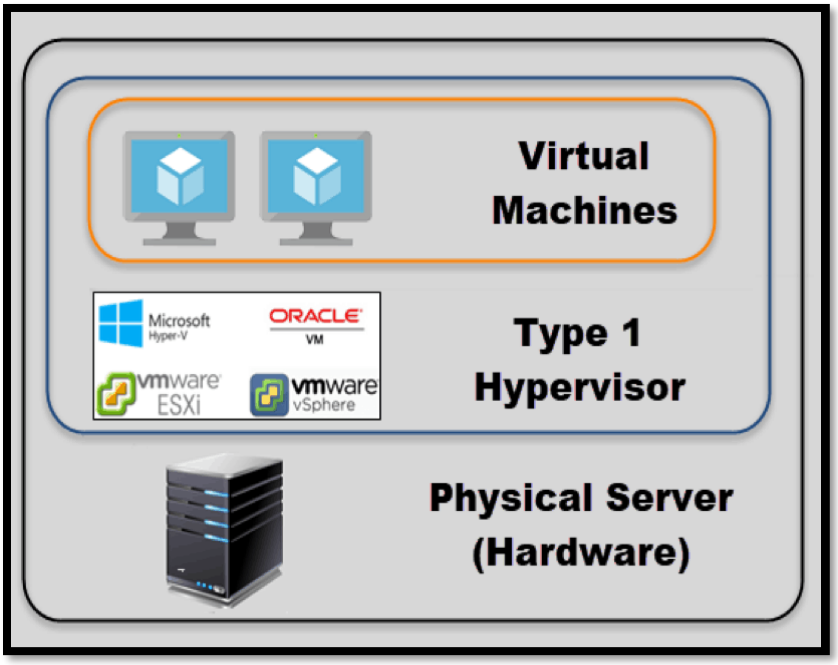


Fig 3.1.9.1.A

B. Type two is a hosted hypervisor that runs as a software layer within a physical operating system. The hypervisor runs as a separate second layer over the hardware while the operating system runs as a third layer. The hosted hypervisors include Parallels Desktop and VMware Player.

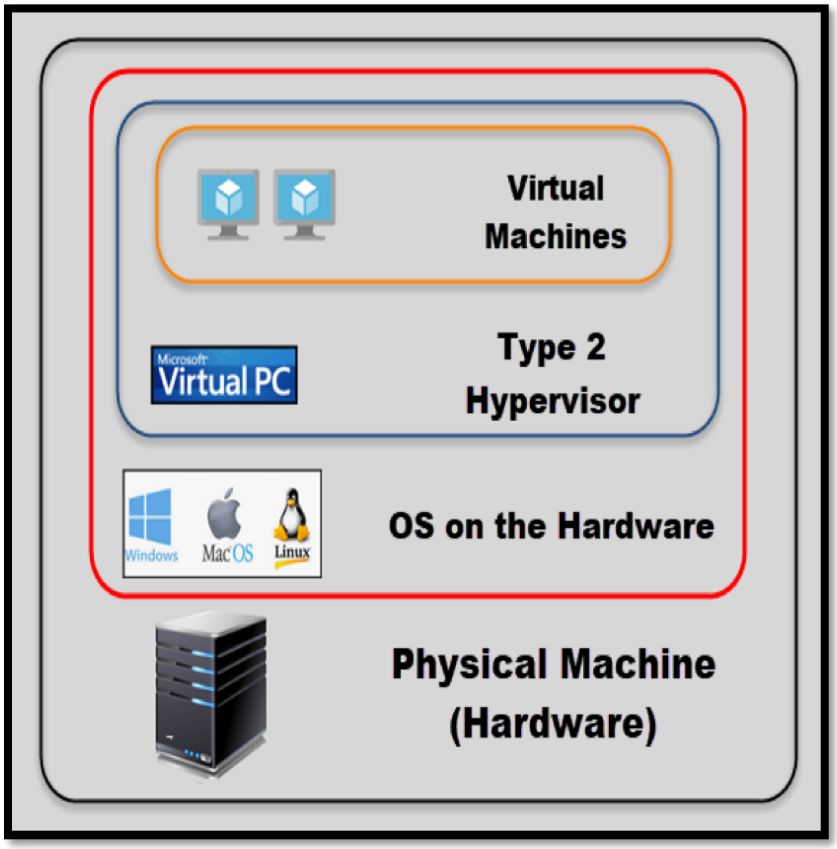


Fig 3.1.9.1.B

3.1.10 Resource Cluster

Cloud-based IT resources that are geographically diverse can be logically combined into groups to improve their allocation and use. The resource cluster mechanism is used to group multiple IT resource instances so that they can be operated as a single IT resource. This increases the combined computing capacity, load balancing, and availability of the clustered IT resources.

Resource cluster architectures rely on high-speed dedicated network connections, or cluster nodes, between IT resource instances to communicate about workload distribution, task scheduling, data sharing, and system synchronization. A cluster management platform that is running as distributed middleware in all of the cluster nodes is usually responsible for these activities. This platform implements a coordination function that allows distributed IT resources to appear as one IT resource, and also executes IT resources inside the cluster.

5.10.1 Common resource cluster types include:

A. Server Cluster – Physical or virtual servers are clustered to increase performance and availability. Hypervisors running on different physical servers can be configured to share virtual server execution state (such as memory pages and processor register state) in order to establish clustered virtual servers. In such configurations, which usually require physical servers to have access to shared storage, virtual servers are able to live-migrate from one to another. In this process, the virtualization platform suspends the execution of a given

virtual server at one physical server and resumes it on another physical server. The process is transparent to the virtual server operating system and can be used to increase scalability by live-migrating a virtual server that is running at an overloaded physical server to another physical server that has suitable capacity.

B. **Database Cluster –** Designed to improve data availability, this high-availability resource cluster has a synchronization feature that maintains the consistency of data being stored at different storage devices used in the cluster. The redundant capacity is usually based on an active-active or active-passive failover system committed to maintaining the synchronization conditions.

C. **Large Dataset Cluster** – Data partitioning and distribution is implemented so that the target datasets can be efficiently partitioned without compromising data integrity or computing accuracy. Each cluster node processes workloads without communicating with other nodes as much as in other cluster types.

Many resource clusters require cluster nodes to have almost identical computing capacity and characteristics in order to simplify the design of and maintain consistency within the resource cluster architecture. The cluster nodes in high-availability cluster architectures need to access and share common storage IT resources. This can require two layers of communication between the nodes—one for accessing the storage device and another to execute IT resource orchestration (Figure 3.1.10.1). Some resource clusters are designed with more loosely coupled IT resources that only require the network layer (Figure 2).

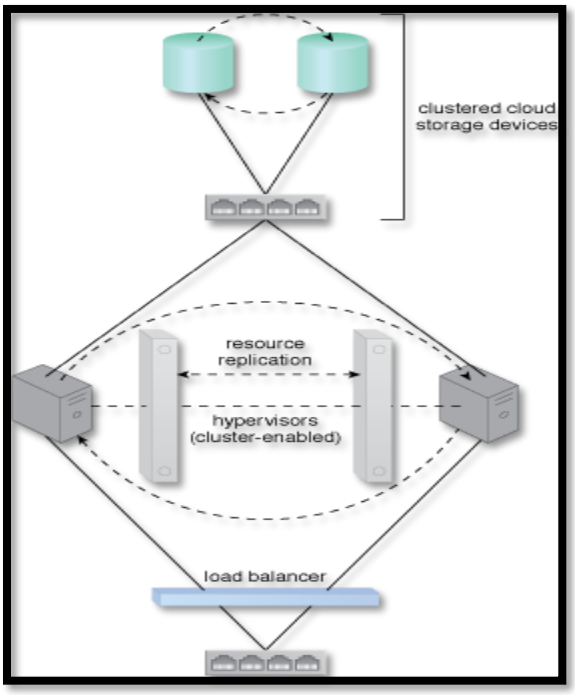


Fig 3.1.10.1

Figure 3.1.10.1 – Load balancing and resource replication are implemented through a cluster enabled hypervisor. A dedicated storage area network is used to connect the clustered storage and the clustered servers, which are able to share common cloud storage devices. This simplifies the storage replication process, which is independently carried out at the storage cluster.

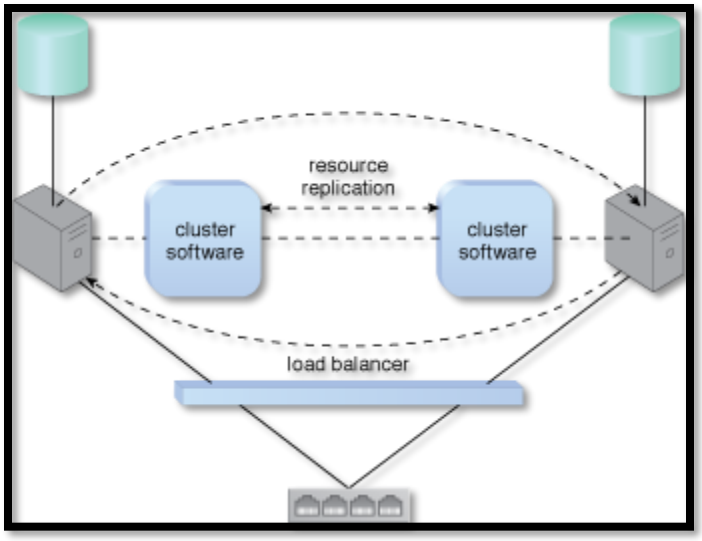


Fig 3.1.10.2

Figure 3.1.10.2 – A loosely coupled server cluster that incorporates a load balancer. There is no shared storage. Resource replication is used to replicate cloud storage devices through the network by the cluster software.

3.1.10.2 There are two basic types of resource clusters:

A. **Load Balanced Cluster –** This resource cluster specializes in distributing workloads among cluster nodes to increase IT resource capacity while preserving the centralization of IT resource management. It usually implements a load balancer mechanism that is either embedded within the cluster management platform or set up as a separate IT resource.

B. **High-Availability (HA) Cluster –** A high-availability cluster maintains system availability in the event of multiple node failures, and has redundant implementations of most of all of the clustered IT resources. It implements a failover system mechanism that monitors failure conditions and automatically redirects the workload away from any failed nodes.

3.1.11 Multi-Device Broker

An individual cloud service may need to be accessed by different types of cloud service consumers, some of which may be incompatible with the cloud service's published service contract. Disparate cloud service consumers may be differentiated by their hosting hardware devices and/or may have different types of communication requirements. To overcome incompatibilities between a cloud service and a disparate cloud service consumer, mapping logic needs to be created to transform (or convert) information that is exchanged at runtime.

The multi-device broker mechanism is used to facilitate runtime data transformation so as to make a cloud service accessible by a wider range of cloud service consumer programs and devices (Figure 1).

Multi-device brokers commonly exist as or incorporate gateway components, such as:

- XML Gateway – transmits and validates XML data
- Cloud Storage Gateway – transforms cloud storage protocols and encodes storage devices to facilitate data transfer and storage
- Mobile Device Gateway – transforms the communication protocols used by mobile devices

The levels at which transformation logic can be created include:

- transport protocols
- messaging protocols
- storage device protocols
- data schemas/data models

For example, a multi-device broker may contain mapping logic that coverts both transport and messaging protocols for a cloud service consumer accessing a cloud service with a mobile device.

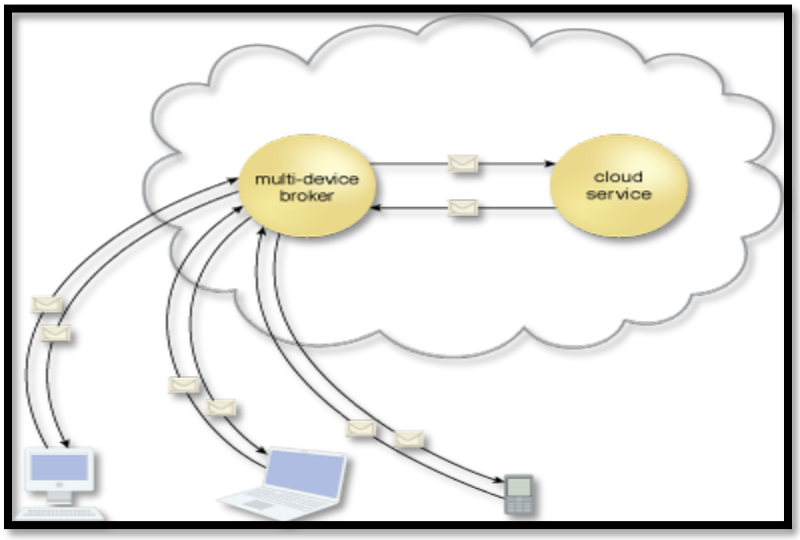


Fig 3.1.11

Figure 3.1.11 – A multi-device broker contains the mapping logic necessary to transform data exchanges between a cloud service and different types of cloud service consumer devices.

3.1.12 State Management Database

A state management database is a storage device that is used to temporarily persist state data for software programs. As an alternative to caching state data in memory, software programs can offload state data to the database in order to reduce the amount of run-time memory they consume (Figures 3.1.12.1 and 3.1.12.2). By doing so, the software programs and the surrounding infrastructure are more scalable. State management databases are commonly used by cloud services, especially those involved in long-running runtime activities.

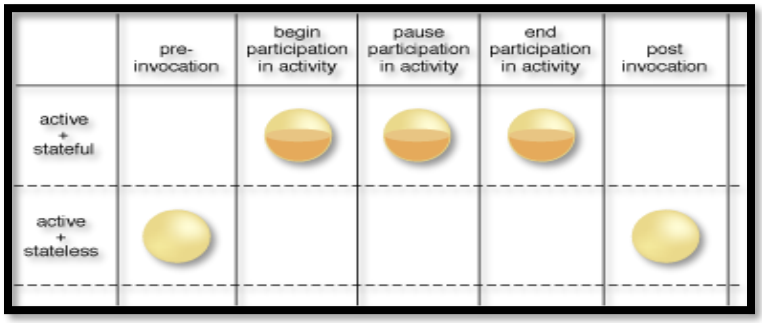


Fig 3.1.12.1

Figure 3.1.12.1 – During the lifespan of a cloud service instance, it may be required to remain stateful and keep state data cached in memory, even when idle.

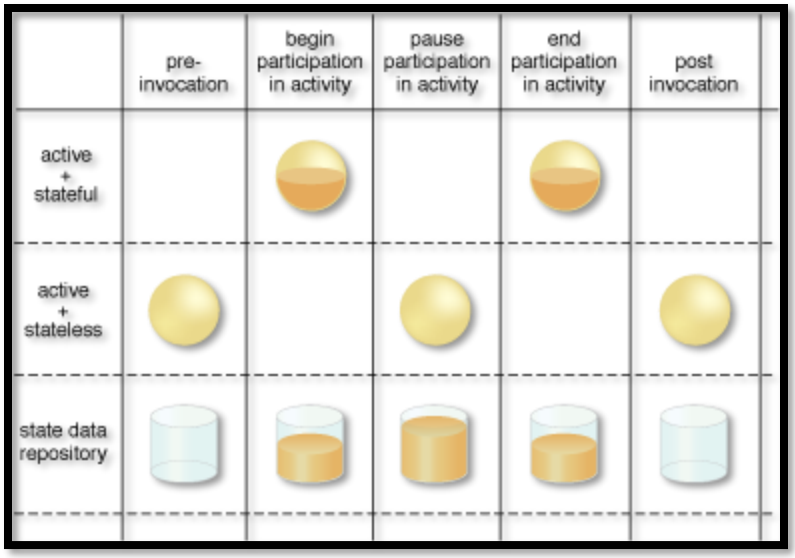


Fig 3.1.12.2

Figure 3.1.12.2 – By deferring state data to a state repository, the cloud service is able to transition to a stateless condition (or a partially stateless condition), thereby temporarily freeing system resources.

3.1.13 Sample Question Exercise

1. Explain Automated Scaling Listener.

2. Discuss the case study on DTGOV. Explain in detail.

3. What is load balancing work? How does load balancing work?

4. Explain SLA monitor in detail.

3.1. Explain Pay-per-use monitor in detail.

6. Explain Audit monitor in detail.

7. What is Failover System? Explain its types in details.

8. What is Hypervisor? Explain its types in details.

9. What is resource cluster?

10. Explain common type cluster and basic resource cluster type.

11. Explain in detail Multidevice broker.

12. Explain in detail State management database.

REFERENCES:

1. https://www.znetlive.com/blog/what-is-load-balancing-in-cloud-computing-and-its-advantages/
2. https://www.cloudoye.com/kb/general/what-is-hypervisor-in-cloud-computing-and-its-types
3. https://patterns.arcitura.com/cloud-computing-patterns/mechanisms/state_management_database

UNIT-III
CHAPTER-2
CLOUD MANAGEMENT MECHANISMS & CLOUD SECURITY MECHANISMS

Unit Structure:

3.2.1 Objective:

To understand the security issues and to identify the appropriate security techniques those are being used in the current world of Cloud Computing To identify the security challenges those are expected in the future of Cloud Computing. To suggest some counter measures for the future challenges to be faced in Cloud Computing.

3.2.2 Introduction:

The cloud management mechanisms are measures to be taken to ensure that there are security mechanisms of cloud solutions in place to deal with the security attacks and threats. Cloud management mechanisms can help to facilitate the control, management and the evolutions of cloud technology and IT resources that form part of the cloud platforms and solutions. As cloud-based IT resources must be configured, set-up, maintained, and monitored, there are systems and mechanisms that should be in place to managed this tasks.

These mechanisms of how these systems are managed are discussed and they typically provide an integrated APIs that can offer individual products, custom applications, which can be combined into various product suites or multi-function applications.

3.2.3 Remote Administration System:

The remote administration system mechanism provides tools and user-interfaces for external cloud resource administrators to configure and administer cloud-based IT resources.

A remote administration system can establish a portal for access to administration and management features of various underlying systems, including the resource management, SLA management, and billing management systems (Figure 1).

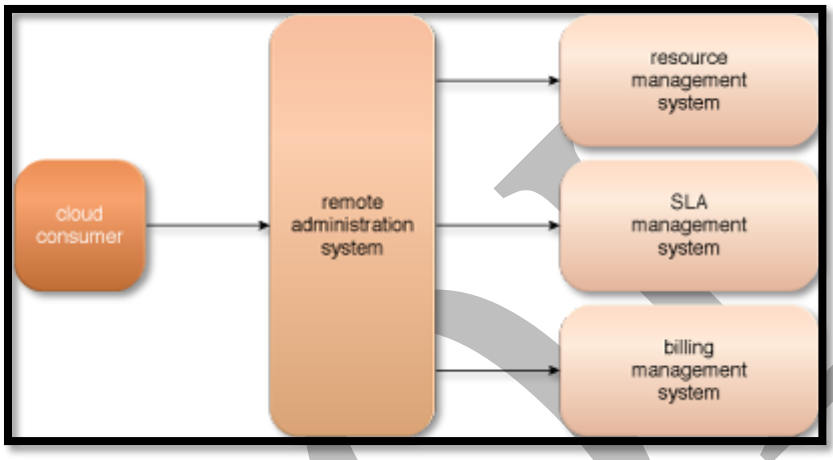


Fig 3.2.3.1

Figure 3.2.3.1 – The remote administration system abstracts underlying management systems to expose and centralize administration controls to external cloud resource administrators. The system provides a customizable user console, while programmatically interfacing with underlying management systems via their APIs.

The tools and APIs provided by a remote administration system are generally used by the cloud provider to develop and customize online portals that provide cloud consumers with a variety of administrative controls.

The following are the **two primary types of portals** that are created with the remote administration system:

- **Usage and Administration Portal** – A general purpose portal that centralized management controls to different cloud-based IT resources and can further provide IT resource usage reports.
- **Self-Service Portal** – This is essentially a shopping portal that allows cloud consumers to search an up-to-date list of cloud services and IT resources that are available from a cloud provider (usually for lease). The cloud consumer submits its chosen items to the cloud provider for provisioning.

Figure 3.2.3.2 illustrates a scenario involving a remote administration system and both usage and administration and self-service portals.

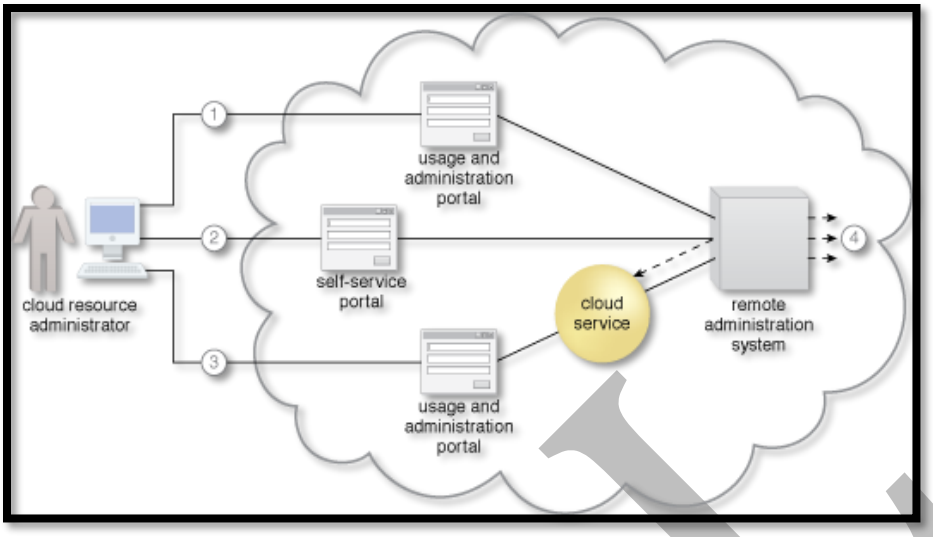


Fig 3.2.3.2

Figure 3.2.3.2 – A cloud resource administrator uses the usage and administration portal to configure an already leased virtual server (not shown) to prepare it for hosting (1). The cloud resource administrator then uses the self-service portal to select and request the provisioning of a new cloud service (2). The cloud resource administrator then accesses the usage and administration portal again to configure the newly provisioned cloud service that is hosted on the virtual server (3). Throughout these steps, the remote administration system interacts with the necessary management systems to perform the requested actions (4).

Depending on:

- the type of cloud product or cloud delivery model the cloud consumer is leasing or using from the cloud provider,
- the level of access control granted by the cloud provider to the cloud consumer, and
- further depending on which underlying management systems the remote administration system interfaces with,

Tasks that can commonly be performed by cloud consumers via a remote administration console include:

- configuring and setting up cloud services
- provisioning and releasing IT resource for on-demand cloud services
- monitoring cloud service status, usage, and performance
- monitoring QoS and SLA fulfillment
- managing leasing costs and usage fees
- managing user accounts, security credentials, authorization, and access control
- tracking internal and external access to leased services
- planning and assessing IT resource provisioning
- capacity planning

While the user-interface provided by the remote administration system will tend to be proprietary to the cloud provider, there is a preference among cloud consumers to work with remote administration systems that offer standardized APIs. This allows a cloud consumer to invest in the creation of its own front-end with the foreknowledge that it can reuse this console if it decides to move to another cloud provider that supports the same standardized API. Additionally, the cloud consumer would be able to further leverage standardized APIs if it is interested in leasing and centrally administering IT resources from multiple cloud providers and/or IT resources residing in cloud and on-premise environments.

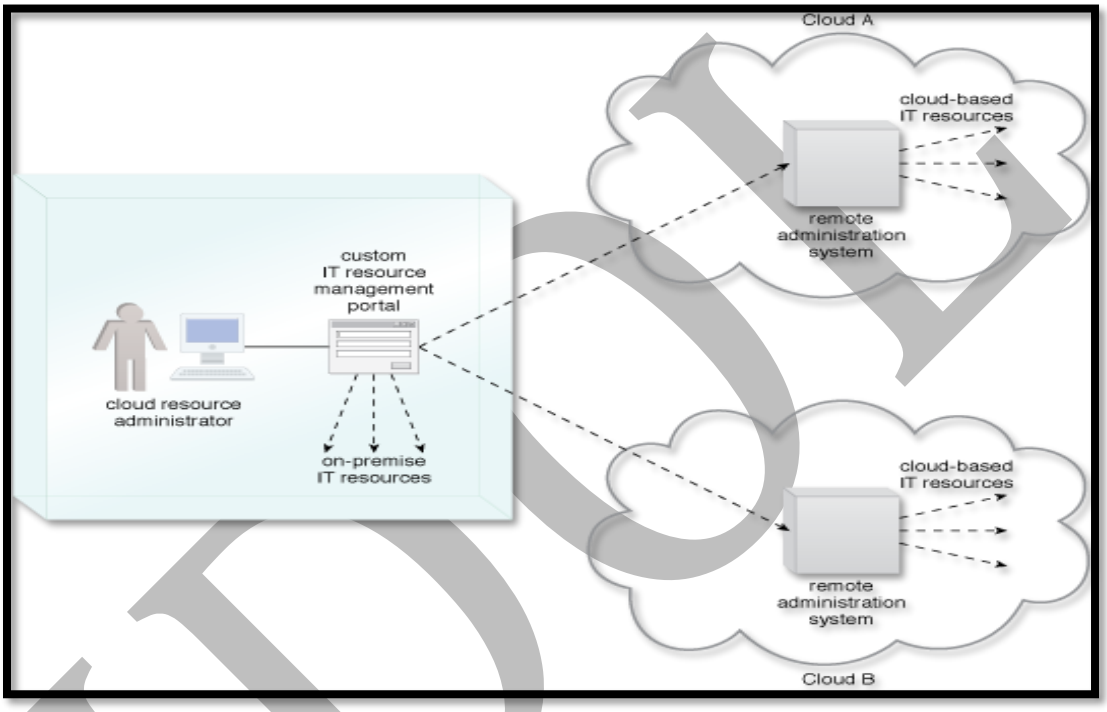


Fig 3.2.3.4

Figure 3.2.3.4 – Standardized APIs published by remote administration systems from different clouds enable a cloud consumer to develop a custom portal that centralizes a single IT resource management portal for both cloud-based and on-premise IT resources.

3.2.4 Resource Management System:

The resource management system mechanism helps coordinate IT resources in response to management actions performed by both cloud consumers and cloud providers (Figure 1). Core to this system is the virtual infrastructure manager (VIM) that coordinates the server hardware so that virtual server instances can be created from the most expedient underlying physical server. VIM is a commercial product that can be used to manage a range of virtual IT resources across multiple physical servers. For example, VIM can create and manage multiple instances of a hypervisor across different physical servers or allocate a virtual server on one physical server to another (or to a resource pool).

Tasks that are typically automated and implemented through the resource management system include:

- managing virtual IT resource templates that are used to create pre-built instances, such as virtual server images
- allocating and releasing virtual IT resources into the available physical infrastructure in response to the starting, pausing, resuming, and termination of virtual IT resource instances
- coordinating IT resources in relation to the involvement of other mechanisms, such as resource replication, load balancer, and failover system
- enforcing usage and security policies throughout the lifecycle of cloud service instances
- monitoring operational conditions of IT resources

Resource management system functions can be accessed by cloud resource administrators employed by the cloud provider or cloud consumer. Those working on behalf of a cloud provider will often be able to directly access the resource management system's native console.

Resource management systems typically expose APIs that allow cloud providers to build remote administration system portals that can be customized to selectively offer resource management controls to external cloud resource administrators acting on behalf of cloud consumer organizations via usage and administration portals.

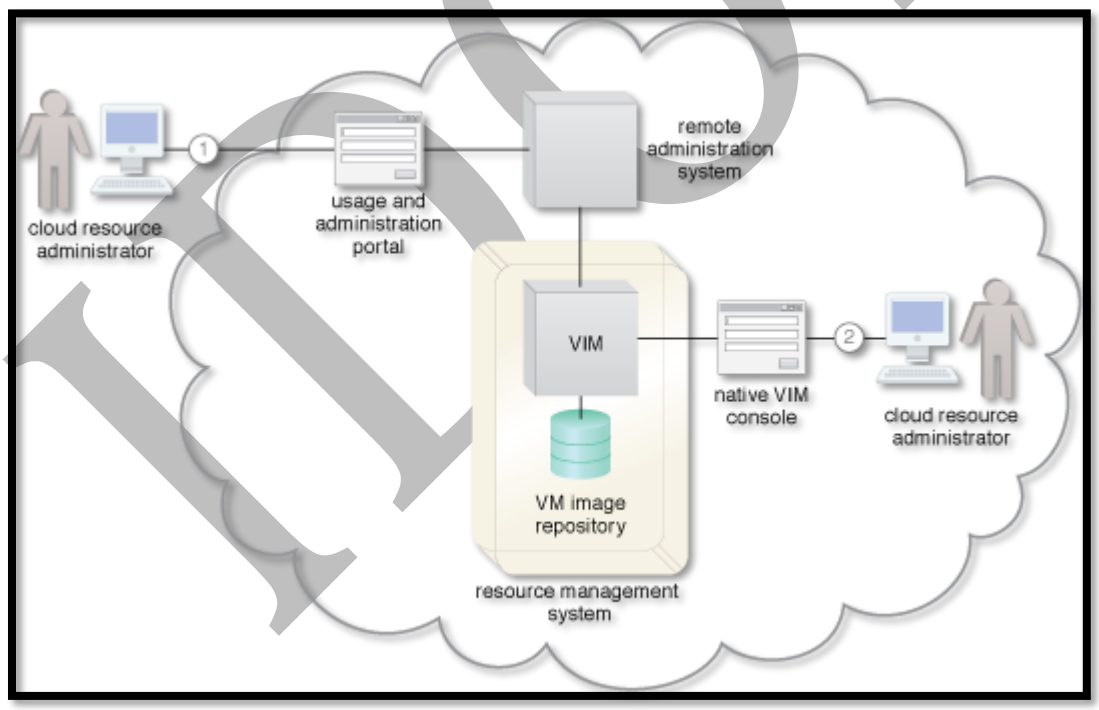


Fig 3.2.4

Figure 3.2.4 – The cloud consumer's cloud resource administrator accesses a usage and administration portal externally to administer a leased IT resource (1). The cloud provider's cloud resource administrator uses the native user-interface provided by the VIM to perform internal resource management tasks (2).

3.2.5 SLA Management System:

The SLA monitor mechanism is used to specifically observe the runtime performance of cloud services to ensure that they are fulfilling the contractual QoS requirements published in SLAs (Figure 1). The data collected by the SLA monitor is processed by an SLA management system to be aggregated into SLA reporting metrics. These systems can proactively repair or failover cloud services when exception conditions occur, such as when the SLA monitor reports a cloud service as "down."

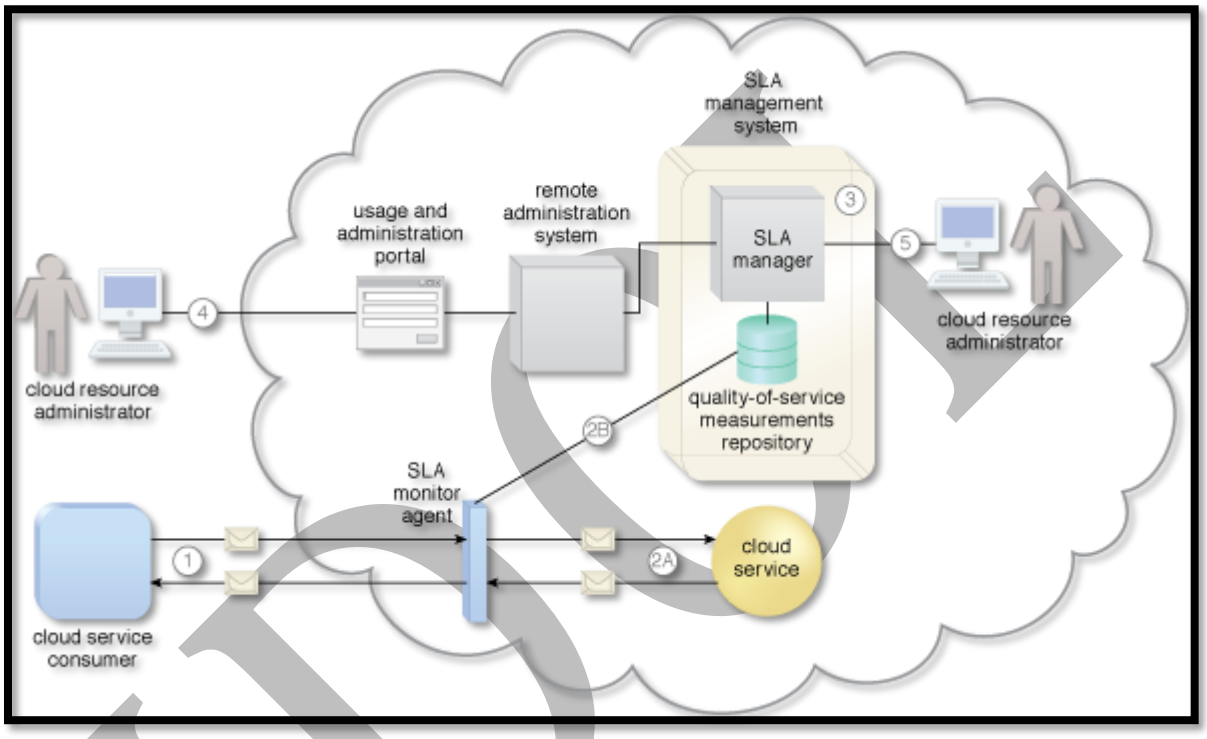


Fig 3.2.5

Figure 3.2.5 – The SLA monitor polls the cloud service by sending over polling request messages (MREQ1 to MREQN). The monitor receives polling response messages (M to M) that report that the service was "up" at each polling cycle (1a). The SLA monitor stores the "up" time—time period of all polling cycles 1 to N—in the log database (1b). The SLA monitor polls the cloud service that sends polling request messages (M to M). Polling response messages are not received (2a). The response messages continue to time out, so the SLA monitor stores the "down" time—time period of all polling cycles N+1 to N+M—in the log database (2b). The SLA monitor sends a polling request message (M) and receives the polling response message (M) (3a). The SLA monitor stores the "up" time in the log database (3b).

3.2.3.2 Billing Management System:

The billing management system mechanism is dedicated to the collection and processing of usage data as it pertains to cloud provider accounting and cloud consumer billing. Specifically, the billing management system relies on pay-per-use monitors to gather runtime usage data that is stored in a

repository that the system components then draw from for billing reporting and invoicing purposes (Figure 1).

The billing management system allows for the definition of different pricing policies as well as custom pricing models on a per-cloud consumer and/or per-IT resource basis. Pricing models can vary from the traditional pay-per-use models to flat-rate or pay-per-allocation models, or combinations thereof.

Billing arrangements can be based on pre-usage and post-usage payments. The latter type can include pre-defined limits or can be set up (with the mutual agreement of the cloud consumer) to allow for unlimited usage (and, consequently, no limit on subsequent billing). When limits are established, they are usually in the form of usage quotas. When quotas are exceeded, the billing management system can block further usage requests by cloud consumers.

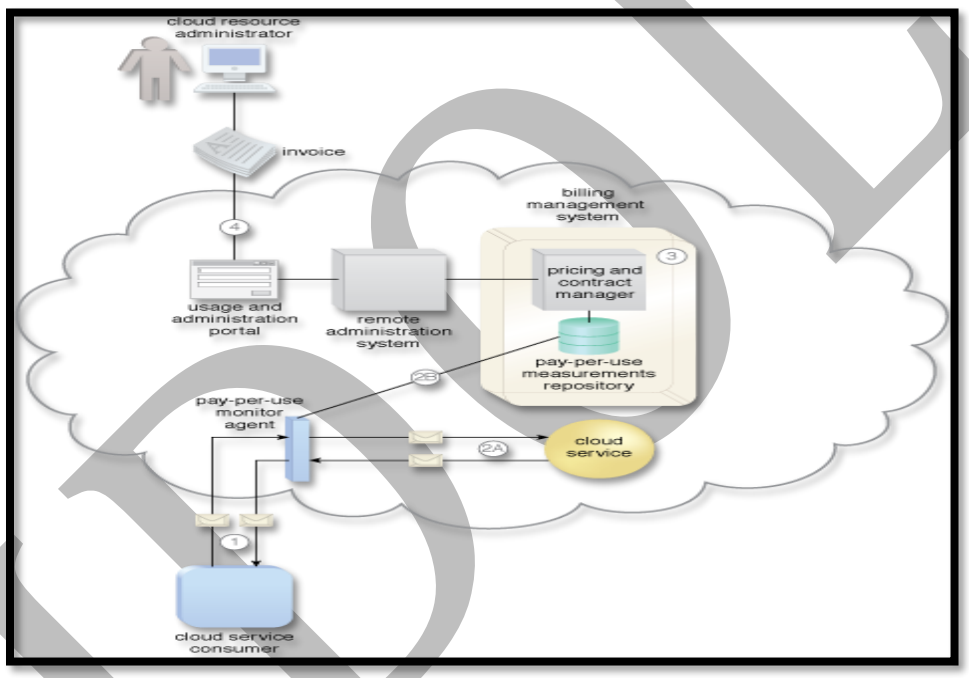


Fig 3.2.3.2

Figure 3.2.3.2 – A cloud service consumer exchanges messages with a cloud service (1). A pay-per-use monitor keeps track of the usage and collects data relevant to billing (2A), which is forwarded to a repository that is part of the billing management system (2B). The system periodically calculates the consolidated cloud service usage fees and generates an invoice for the cloud consumer (3). The invoice may be provided to the cloud consumer through the usage and administration portal (4).

3.2.7 Encryption:

- Data, by default, is coded in a readable format known as plaintext. When transmitted over a network, plaintext is vulnerable to unauthorized and potentially malicious access.
- The encryption mechanism is a digital coding system dedicated to preserving the confidentiality and integrity of data. It is used for encoding plaintext data into a protected and unreadable format.
- Encryption technology commonly relies on a standardized algorithm called a cipher to transform original plaintext data into encrypted data, referred to as ciphertext.
- When encryption is applied to plaintext data, the data is paired with a string of characters called an encryption key, a secret message that is established by and shared among authorized parties. The encryption key is used to decrypt the ciphertext back into its original plaintext format.
- Data encryption in the cloud is the process of transforming or encoding data before it's moved to cloud storage. Typically cloud service providers offer encryption services ranging from an encrypted connection to limited encryption of sensitive data and provide encryption keys to decrypt the data as needed.
- Encryption services like these prevent unauthorized free access to your system or file data without the decryption key, making it an effective data security method. Keeping information secure in the cloud should be your top priority. Just taking a few preventative measures around data encryption can tighten security for your most sensitive information. Follow these encryption tips to lock down your information in the cloud.

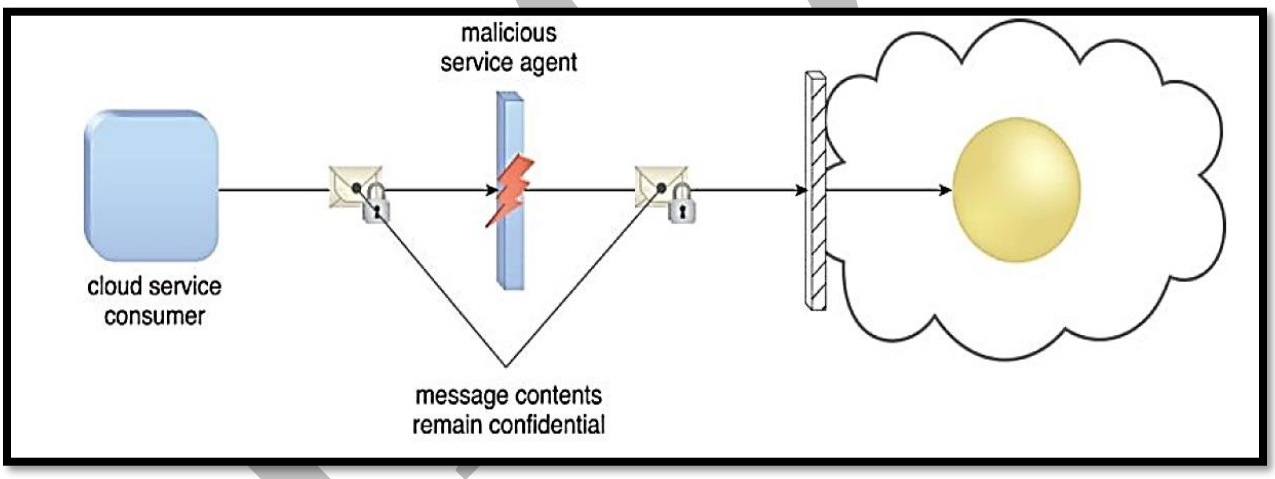


Fig 3.2.7

Figure 3.2.7 A malicious service agent is unable to retrieve data from an encrypted message. The retrieval attempt may furthermore be revealed to the cloud service consumer.

There are two common forms of encryption known as symmetric encryption and asymmetric encryption.

3.2.7.1 Symmetric Encryption:

- Symmetric encryption uses the same key for both encryption and decryption, both of which are performed by authorized parties that use the one shared key. Also known as secret key cryptography, messages that are encrypted with a specific key can be decrypted by only

that same key Note that symmetrical encryption does not have the characteristic of non-repudiation.

3.2.7.2 Asymmetric Encryption

- Asymmetric encryption relies on the use of two different keys, namely a private key and a public key. With asymmetric encryption (which is also referred to as public key cryptography), the private key is known only to its owner while the public key is commonly available. A document that was encrypted with a private key can only be correctly decrypted with the corresponding public key.
- Conversely, a document that was encrypted with a public key can be decrypted only using its private key counterpart. Asymmetric encryption is almost always computationally slower than symmetric encryption. Private Key encryption therefore offers integrity protection in addition to authenticity and non-repudiation.

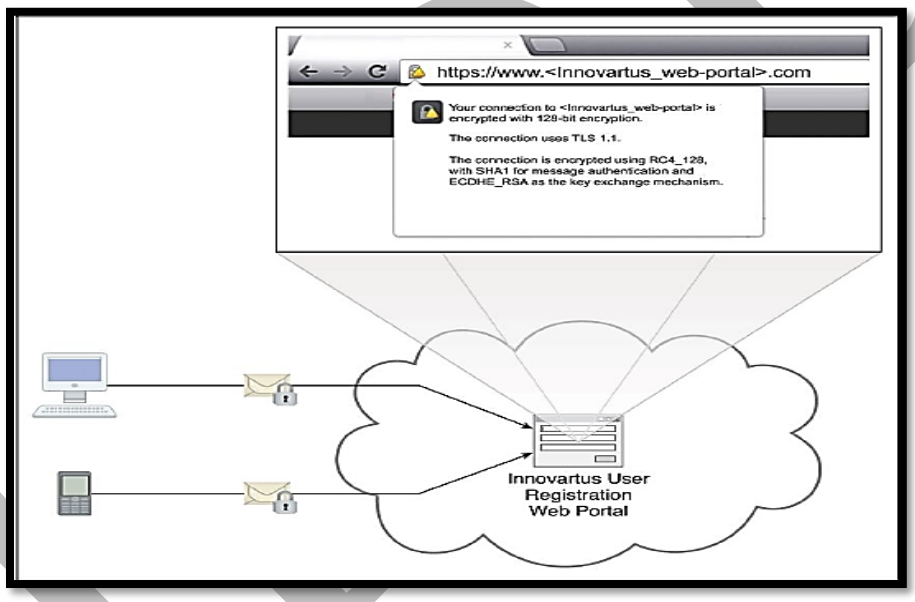


Fig 3.2.7.2

Figure 3.2.7.2 The encryption mechanism is added to the communication channel between outside users and Innovartus' User Registration Portal. This safeguards message confidentiality via the use of HTTPS.

3.2.8 Hashing

- Hashing the hashing mechanism is used when a one-way, non-reversible form of data protection is required. Once hashing has been applied to a message, it is locked and no key is provided for the message to be unlocked.
- A common application of this mechanism is the storage of passwords. Hashing technology can be used to derive a hashing code or message digest from a message, which is often of a fixed length and smaller than the original message.

- The message sender can then utilize the hashing mechanism to attach the message digest to the message. The recipient applies the same hash function to the message to verify that the produced message digest is identical to the one that accompanied the message.
- Any alteration to the original data results in an entirely different message digest and clearly indicates that tampering has occurred.

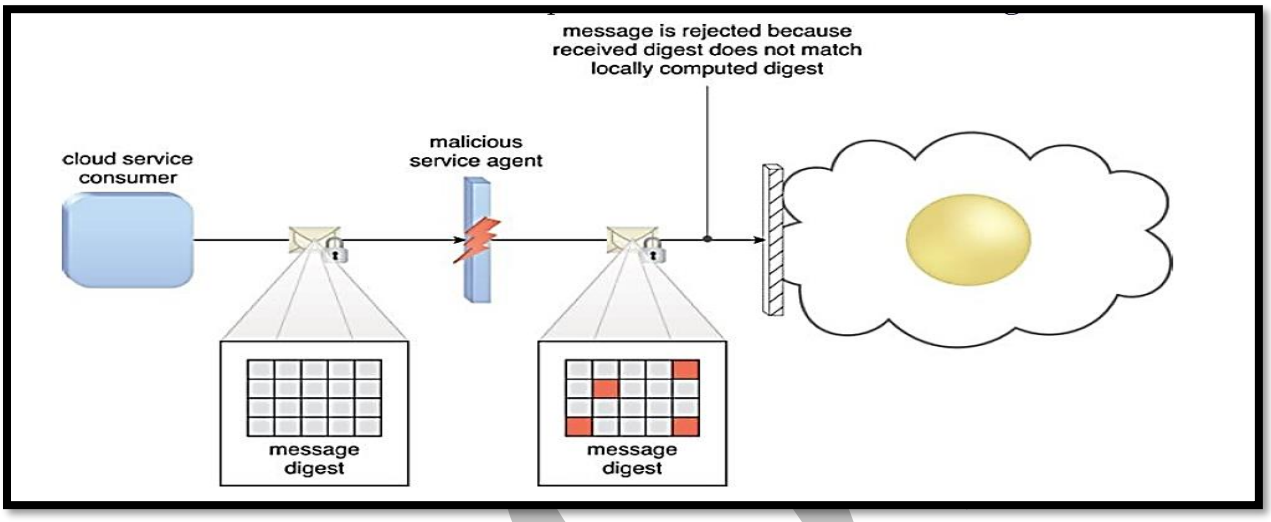


Fig 3.2.8.1

Figure 3.2.8. A hashing function is applied to protect the integrity of a message that is intercepted and altered by a malicious service agent, before it is forwarded. The firewall can be configured to determine that the message has been altered, thereby enabling it to reject the message before it can proceed to the cloud service.

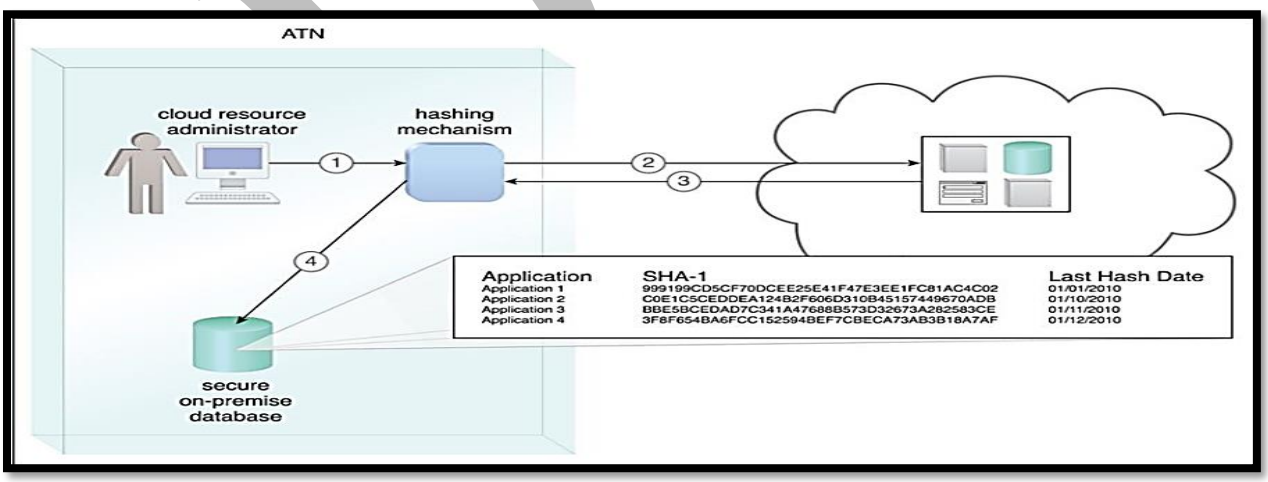


Fig 3.2.8.2

Figure 3.2.8.2 A hashing procedure is invoked when the PaaS environment is accessed (1). The applications that were ported to this environment are checked . (2). and their message digests are calculated. (3). The message digests are stored in a secure on-premise database. (4), and a notification are issued if any of their values are not identical to the ones in storage.

3.2.9 Digital Signature:

- The digital signature mechanism is a means of providing data authenticity and integrity through authentication and non-repudiation.
- A message is assigned a digital signature prior to transmission, which is then rendered invalid if the message experiences any subsequent, unauthorized modifications.
- A digital signature provides evidence that the message received is the same as the one created by its rightful sender.
- Both hashing and asymmetrical encryption are involved in the creation of a digital signature, which essentially exists as a message digest that was encrypted by a private key and appended to the original message. The recipient verifies the signature validity and uses the corresponding public key to decrypt the digital signature, which produces the message digest.

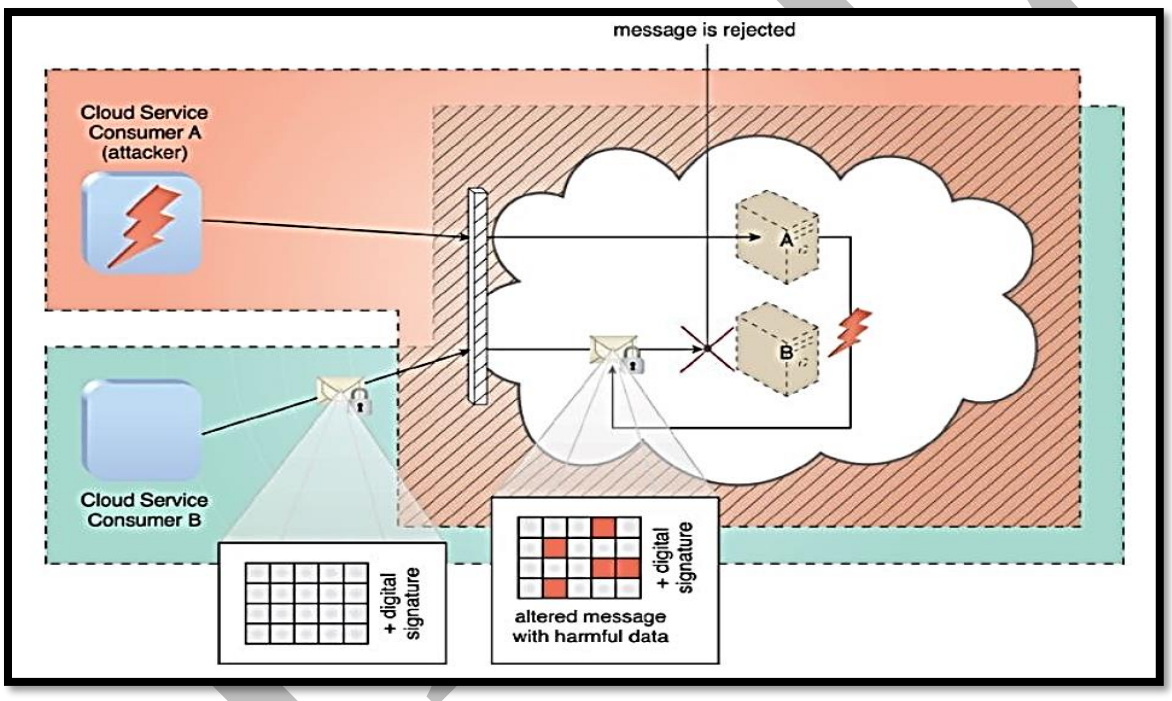


Fig 3.2.9.1

Figure 3.2.9.1 Cloud Service Consumer B sends a message that was digitally signed but was altered by trusted attacker Cloud Service Consumer A. Virtual Server B is configured to verify digital signatures before processing incoming messages even if they are within its trust boundary. The message is revealed as illegitimate due to its invalid digital signature, and is therefore rejected by Virtual Server B.

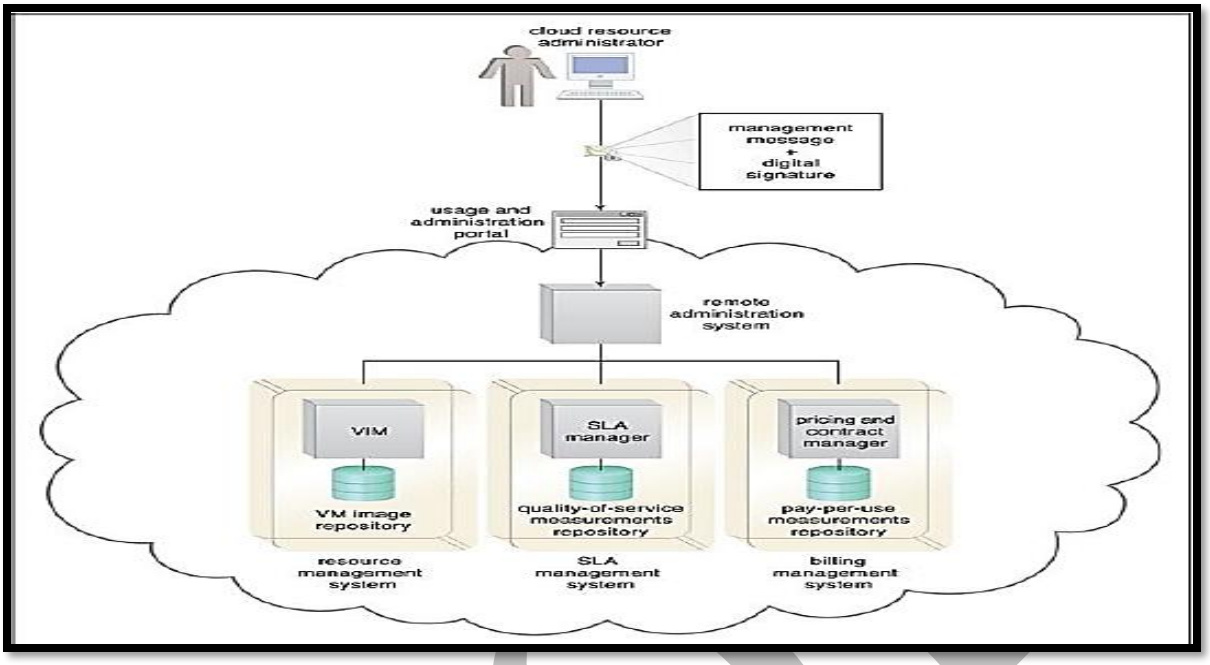


Fig 3.2.9.2

Figure 3.2.9.2 Whenever a cloud consumer performs a management action that is related to IT resources provisioned by DTGOV, the cloud service consumer program must include a digital signature in the message request to prove the legitimacy of its user.

3.2.10 Public Key Infrastructure (PKI):

- Public Key Infrastructure (PKI) A common approach for managing the issuance of asymmetric keys is based on the public key infrastructure (PKI) mechanism, which exists as a system of protocols, data formats, rules, and practices that enable large-scale systems to securely use public key cryptography.
- This system is used to associate public keys with their corresponding key owners (known as public key identification) while enabling the verification of key validity.
- PKIs rely on the use of digital certificates, which are digitally signed data structures that bind public keys to certificate owner identities, as well as to related information, such as validity periods. Digital certificates are usually digitally signed by a third-party certificate authority (CA), as illustrated in Figure 7.

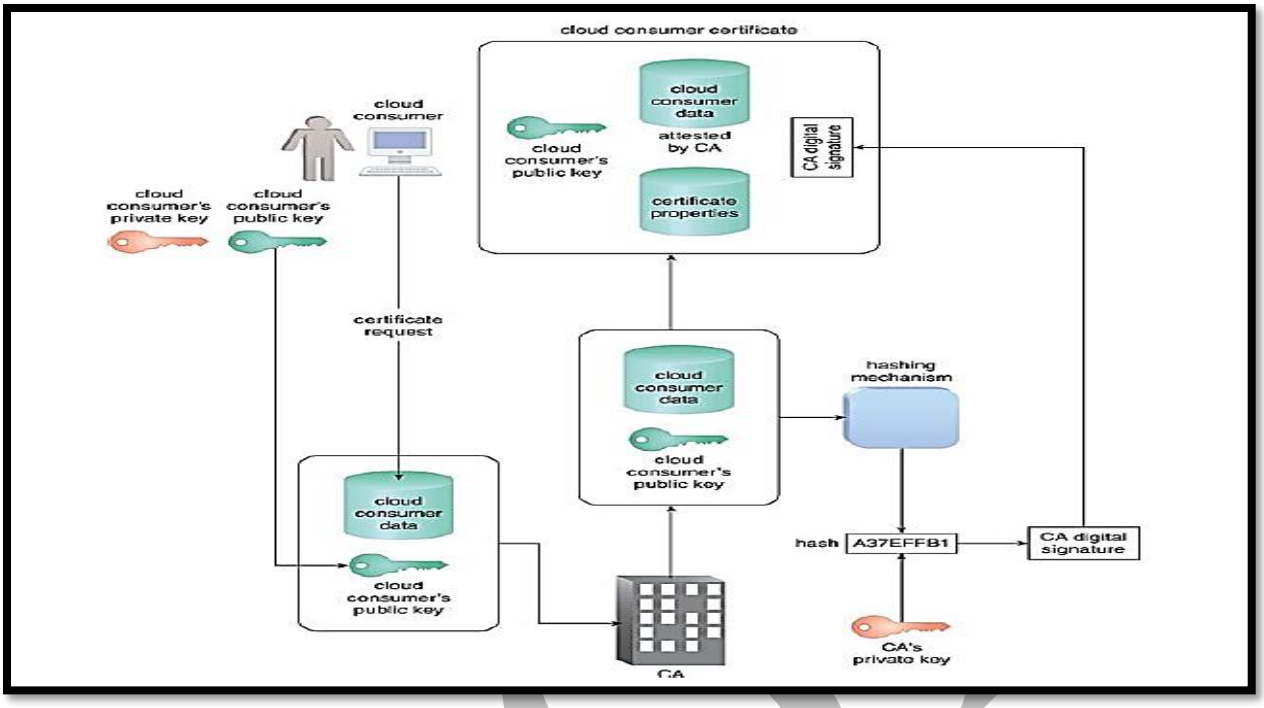


Fig 3.2.10.1

Figure 3.2.10.1 The common steps involved during the generation of certificates by a certificate authority.

- Public Key Infrastructure (PKI) Larger organizations, such as Microsoft, can act as their own CA and issue certificates to their clients and the public, since even individual users can generate certificates as long as they have the appropriate software tools.
- The PKI is a dependable method for implementing asymmetric encryption, managing cloud consumer and cloud provider identity information, and helping to defend against the malicious intermediary and insufficient authorization threats.
- The PKI mechanism is primarily used to counter the insufficient authorization threat.

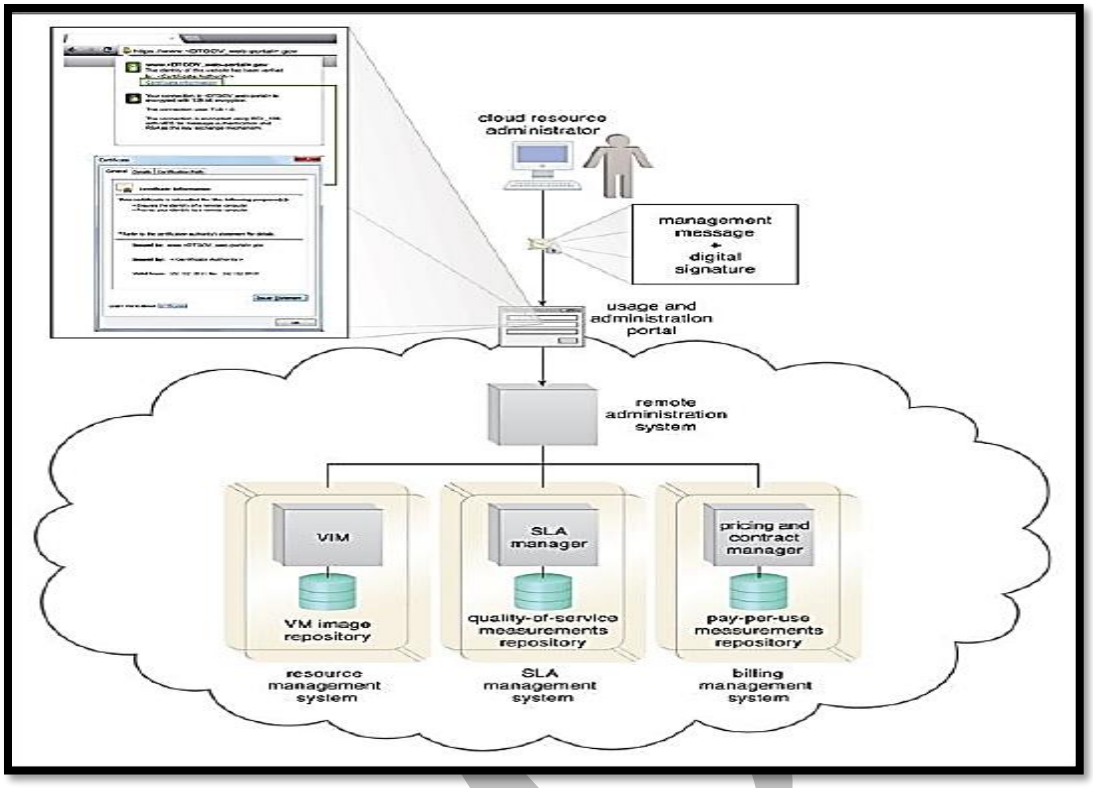


Fig 3.2.10.2

Figure 3.2.10.2 An external cloud resource administrator uses a digital certificate to access the Web-based management environment. DTGOV's digital certificate is used in the HTTPS connection and then signed by a trusted CA.

3.2.11 Identity and Access Management (IAM):

Identity and Access Management (IAM) The Identity and access management (IAM) mechanism encompasses the components and policies necessary to control and track user identities and access privileges for IT resources, environments, and systems. Specifically, IAM mechanisms exist as systems comprised of four main components:

1. **Authentication –** Username and password combinations remain the most common forms of user authentication credentials managed by the IAM system, which also can support digital signatures, digital certificates, biometric hardware (fingerprint readers), specialized software (such as voice analysis programs), and locking user accounts to registered IP or MAC addresses.
2. **Authorization –** The authorization component defines the correct granularity for access controls and oversees the relationships between identities, access control rights, and IT resource availability.
3. **User Management –** Related to the administrative capabilities of the system, the user management program is responsible for creating new user identities and access groups, resetting passwords, defining password policies, and managing privileges.

4. **Credential Management –** The credential management system establishes identities and access control rules for defined user accounts, which mitigates the threat of insufficient authorization. The IAM mechanism is primarily used to counter the insufficient authorization, denial of service, and overlapping trust boundaries threats.

3.2.12 Single Sign-On (SSO):

- Single Sign-On (SSO) Propagating the authentication and authorization information for a cloud service consumer across multiple cloud services can be a challenge, especially if numerous cloud services or cloud-based IT resources need to be invoked as part of the same overall runtime activity.
- The single sign-on (SSO) mechanism enables one cloud service consumer to be authenticated by a security broker, which establishes a security context that is persisted while the cloud service consumer accesses other cloud services or cloud-based IT resources.
- Otherwise, the cloud service consumer would need to re-authenticate itself with every subsequent request. The SSO mechanism essentially enables mutually independent cloud services and IT resources to generate and circulate runtime authentication and authorization credentials.

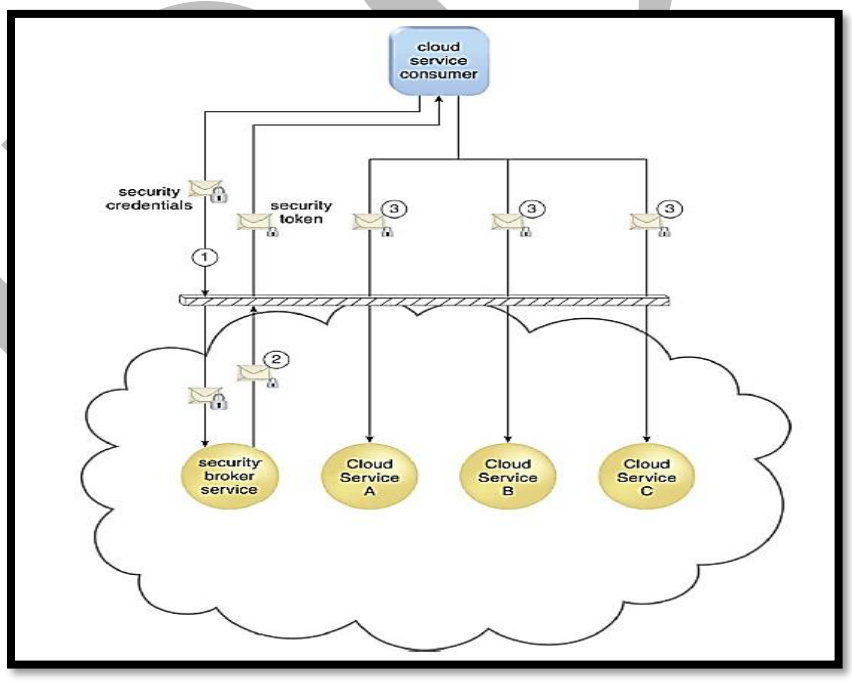


Fig 3.2.12.1

Figure 3.2.12.1 A cloud service consumer provides the security broker with login credentials (1). The security broker responds with an authentication token (message with small lock symbol) upon successful authentication, which contains cloud service consumer identity information (2) that is

used to automatically authenticate the cloud service consumer across Cloud Services A, B, and C (3).

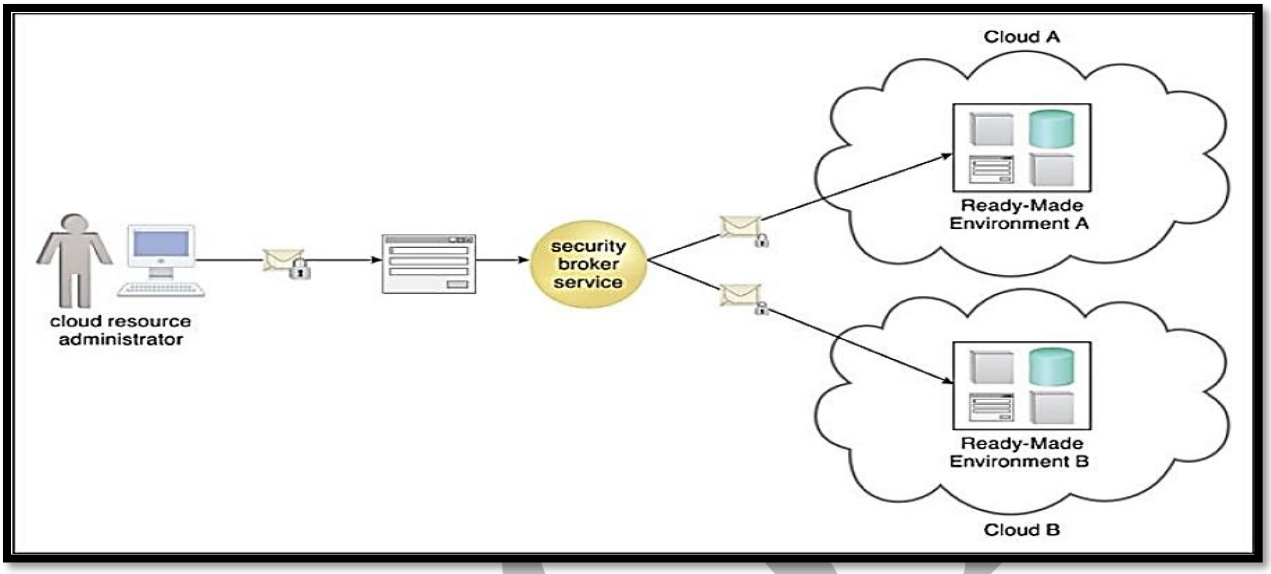


Fig 3.2.12.2

Figure 3.2.12.2. The credentials received by the security broker are propagated to ready-made environments across two different clouds. The security broker is responsible for selecting the appropriate security procedure with which to contact each cloud.

3.2.13 Cloud-Based Security Groups:

- Cloud-Based Security Groups Cloud resource segmentation is a process by which separate physical and virtual IT environments are created for different users and groups. For example, an organization's WAN can be partitioned according to individual network security requirements.
- One network can be established with a resilient firewall for external Internet access, while a second is deployed without a firewall because its users are internal and unable to access the Internet. Resource segmentation is used to enable virtualization by allocating a variety of physical IT resources to virtual machines.
- Cloud-Based Security Groups the cloud-based resource segmentation process creates cloud-based security group mechanisms that are determined through security policies. Networks are segmented into logical cloud-based security groups that form logical network perimeters multiple virtual servers running on the same physical server can become members of different logical cloud-based security groups (Figure 11).
- Virtual servers can further be separated into public-private groups, development-production groups, or any other designation configured by the cloud resource administrator.

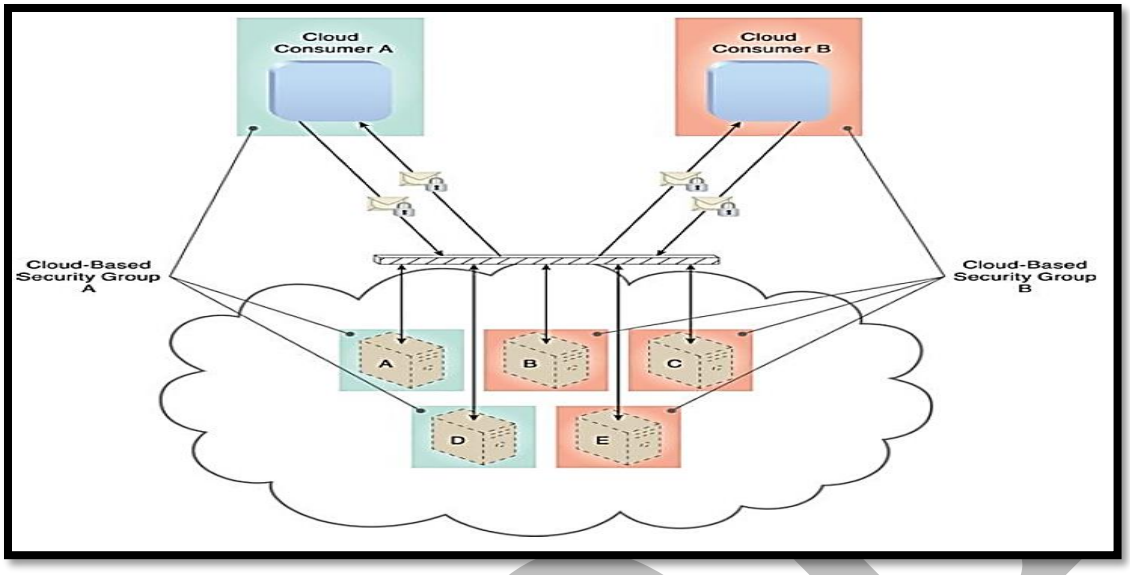


Fig 3.2.13.1

Figure 3.2.13.1 Cloud-Based Security Group A encompasses Virtual Servers A and D and is assigned to Cloud Consumer A. Cloud-Based Security Group B is comprised of Virtual Servers B, C, and E and is assigned to Cloud Consumer B. If Cloud Service Consumer A's credentials are compromised, the attacker would only be able to access and damage the virtual servers in Cloud-Based Security Group A, thereby protecting Virtual Servers B, C, and E.

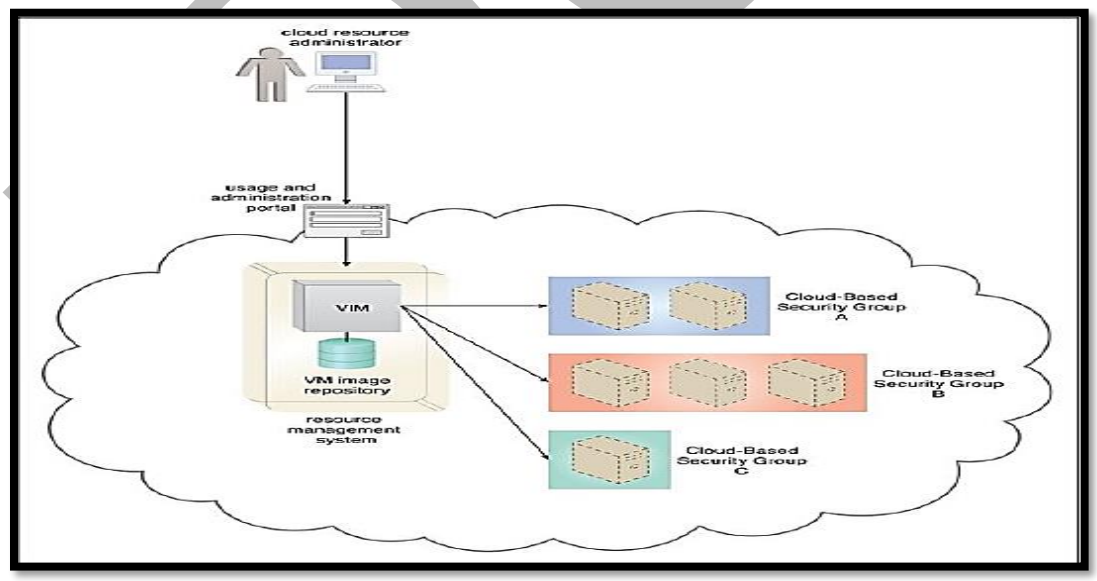


Fig 3.2.13.2

Figure 3.2.13.2 When an external cloud resource administrator accesses the Web portal to allocate a virtual server, the requested security credentials are assessed and mapped to an internal security policy that assigns a corresponding cloud-based security group to the new virtual server.

3.2.14 Hardened Virtual Server Images

- Hardened Virtual Server Images As previously discussed, a virtual server is created from a template configuration called a virtual server image (or virtual machine image). Hardening is the process of stripping unnecessary software from a system to limit potential vulnerabilities that can be exploited by attackers.
- Removing redundant programs, closing unnecessary server ports, and disabling unused services, internal root accounts, and guest access are all examples of hardening.
- A hardened virtual server image is a template for virtual service instance creation that has been subjected to a hardening process (Figure 13).
- This generally results in a virtual server template that is significantly more secure than the original standard image.

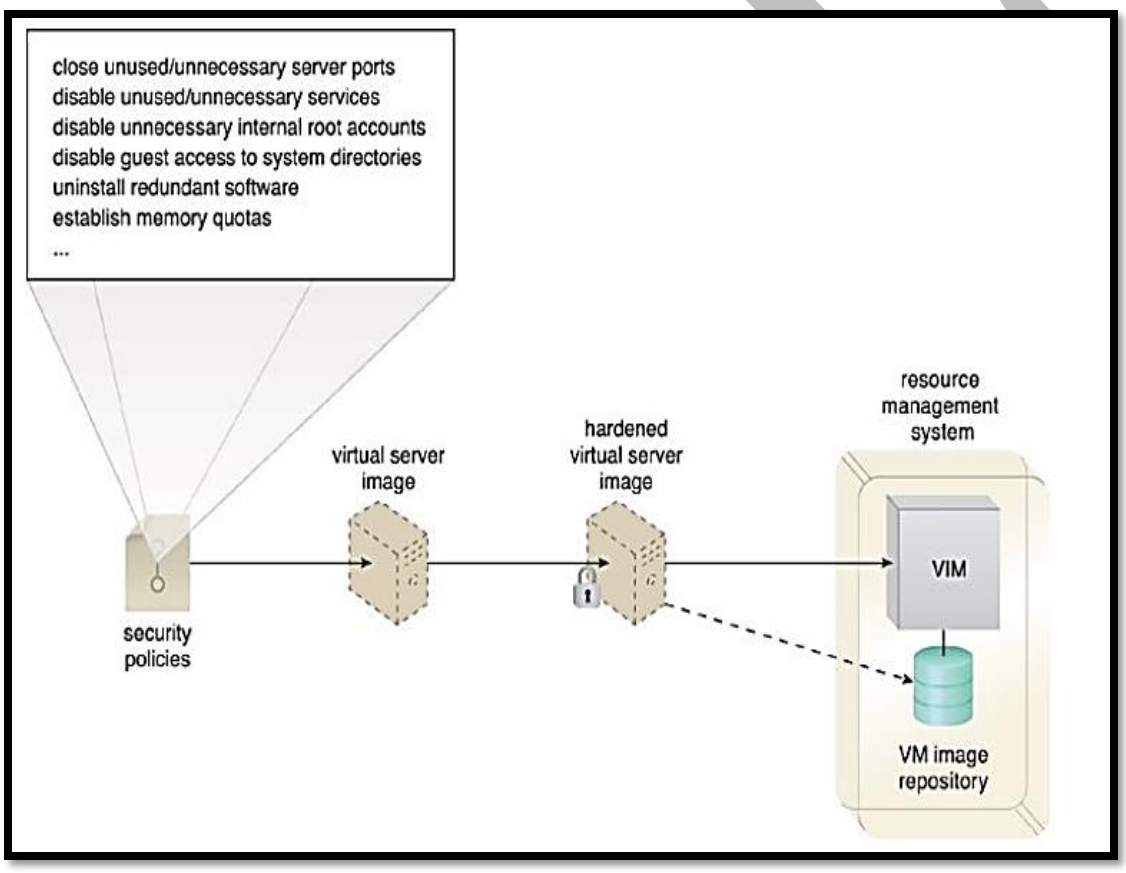


Fig 3.2.14.1

Figure 3.2.14.1 A cloud provider applies its security policies to harden its standard virtual server images. The hardened image template is saved in the VM images repository as part of a resource management system.

- Hardened virtual server images help counter the denial of service, insufficient authorization, and overlapping trust boundaries threats.

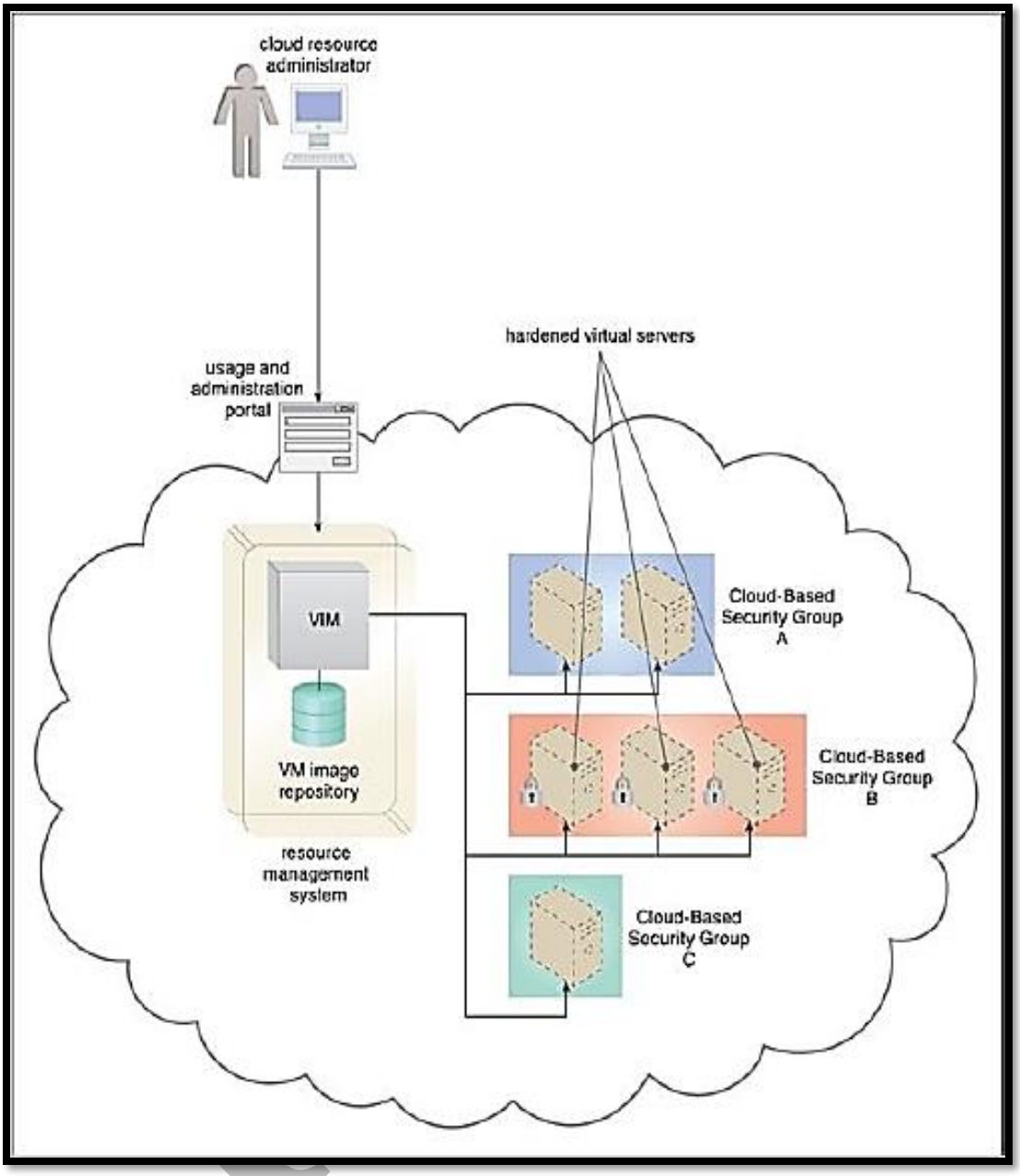


Fig 3.2.14.2

Figure 3.2.14.1 The cloud resource administrator chooses the hardened virtual server image option for the virtual servers provisioned for Cloud-Based Security Group B.

3.2.15 SAMPLE QUESTION EXERCISE

1. Explain Remote administration system. What are the two portals create by remote administration system?

2. What are the tasks that can commonly be performed by cloud consumers via a remote administration console? Explain.

3. What is Resource management? What are the tasks that are typically automated and implemented through the resource management system?

4. Explain SLA management system in detail.

5. Explain Billing management system in detail.

3.2. What is an encryption? Explain its types.

7. What is hashing? Explain with the help of diagram.

8. Explain digital signature in detail.

9. What is PKI (Public Key Infrastructure)? Explain in detail.

10. What is Identity and Access Management (IAM)? Explain its components.

11. Explain Single Sign-On (SSO) in details.

12. Explain Cloud-Based Security Groups in details.

13. How Hardened Virtual Server Images? Explain.

3.2.17 References:

1. https://patterns.arcitura.com/cloud-computing-patterns/mechanisms/remote_administration_system

2. Cloud Computing Concepts, Technology & Architecture By Thomas Erl, Zaigham Mahmood, and Ricardo Puttini Prentice Hall – 2013.

Unit 4
Chapter – 1
Fundamental Cloud Architectures

Unit Structure:

4.1.1 Workload Distribution Architecture
4.2.1 Resource Pooling Architecture
4.3.1 Dynamic Scalability Architecture
4.4.1 Elastic Resource Capacity Architecture
4.5.1 Service Load Balancing Architecture
4.6.1 Cloud Bursting Architecture
4.7.1 Elastic Disk Provisioning Architecture
4.8.1 Redundant Storage Architecture

Objective:

To learn how to use Cloud Services.

Introduction:

This chapter introduces and describes several of the more common foundational cloud architectural models, each explaining a common usage and characteristic of modern day cloud-based environments. Further the chapter also explores the involvement and importance of different combinations of cloud computing mechanisms in relation to these architectures.

4.1. Workload Distribution Architecture

- Resources on cloud can be horizontally scald using an addition or identical resource and a load balancer that is capable of providing run time distribution of workload among resources.
- This architecture of distribution has a dual advantage
  - **i.** Reduces overutilization of resources.
  - **ii.** Reduces underutilization of resources.

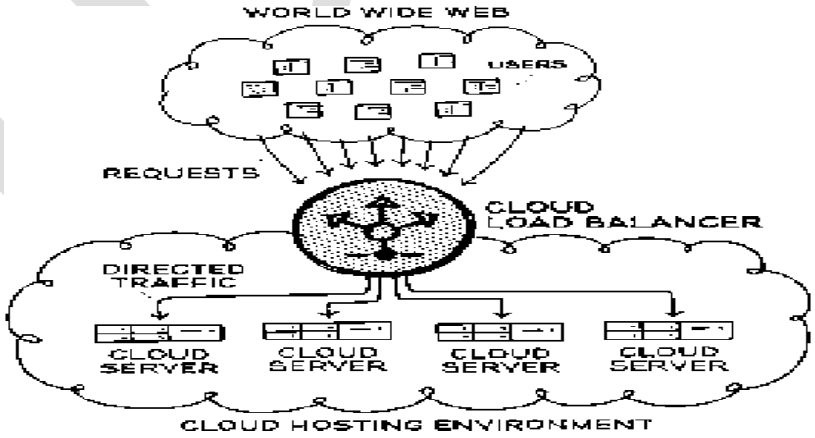


Figure: Workload Distribution Architecture

- Workload distribution is carried out in support of distributed virtual servers, storage devices and services.
- Load balancing system produces specialized variation that incorporates the aspect of load balancing like:
  - **i.** Load Balanced Service Instances Architecture
  - **ii.** Load Balanced Virtual Server Instances Architecture

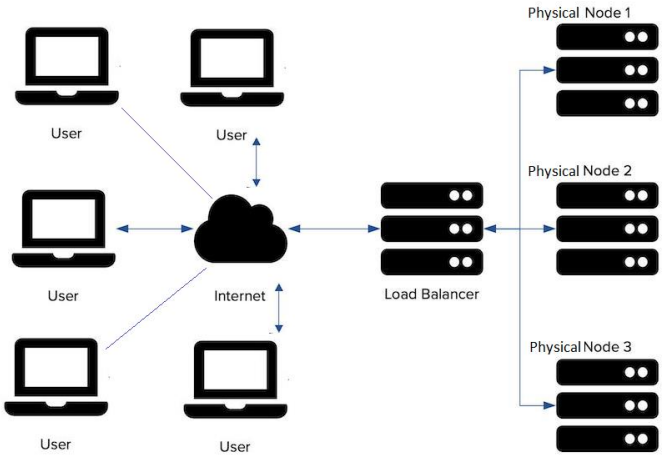
iii. Load Balanced Virtual Switches Architecture

- In addition to the above mentioned mechanism, the following mechanisms can also be part of this cloud architecture:
  - **i.** *Audit Monitor* – Resources that process the data can determine whether monitoring is necessary to fulfill legal and regulatory requirements.
  - **ii.** *Cloud Usage Monitor* – Various monitors can be involved to carry out runtime workload tracking and data processing.
  - **iii.** *Hypervisor* – Workloads between hypervisors and the virtual servers that they host may require distribution.
  - **iv.** *Logical Network Perimeter* – The logical network perimeter isolates cloud consumer network boundaries in relation to how and where workloads are distributed.
  - **v.** *Resource Cluster* – Clustered IT resources in active/active mode are commonly used to support workload balancing between different cluster nodes.
  - **vi.** *Resource Replication* – This mechanism can generate new instances of virtualized IT resources in response to runtime workload distribution demands.

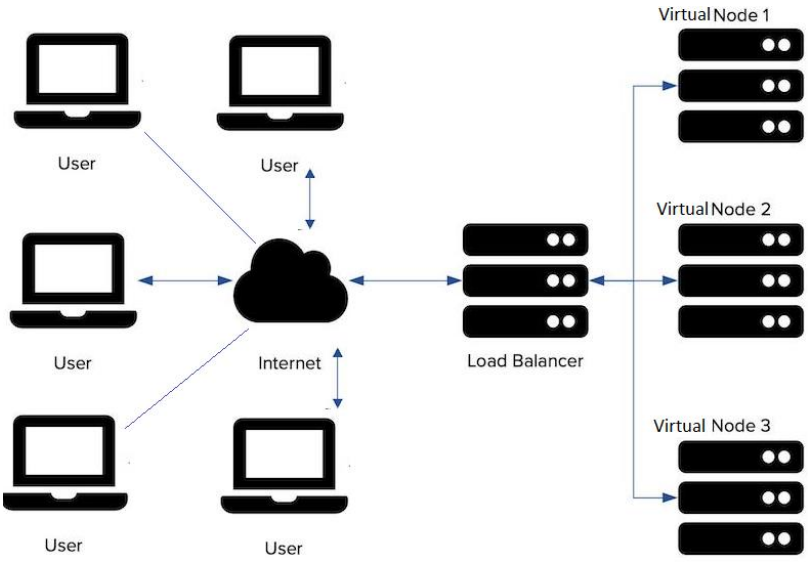
.

4.2. Resource Pooling Architecture

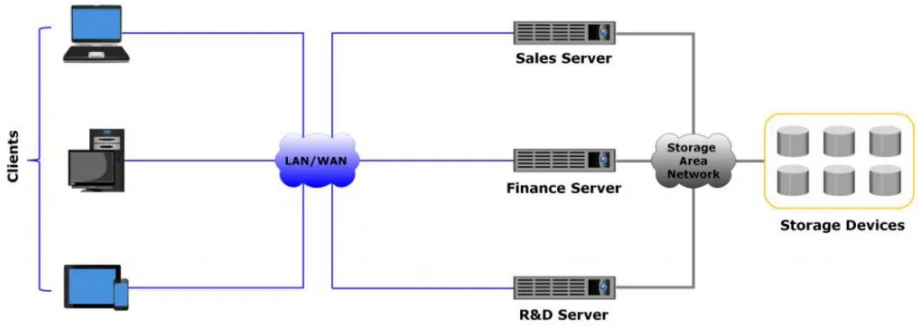
- This architecture is based on the use of one or more resource from a pool of resources, in which identical synchronized resources are grouped and maintained by a system. Examples of resource pools:
  1) **Physical server pools:** are group of physical servers networked to have installed operating systems and other necessary programs and/or applications and are ready for immediate use.



  2) **Virtual server pools:** are group of virtual servers networked to have installed operating systems and other necessary programs and/or applications and are ready for immediate use. They are usually configured using one of several available templates chosen by the cloud consumer during provisioning.
     - For example, a cloud consumer can set up a pool of mid-tier Windows servers with 4 GB of RAM or a pool of low-tier Ubuntu servers with 2 GB of RAM.

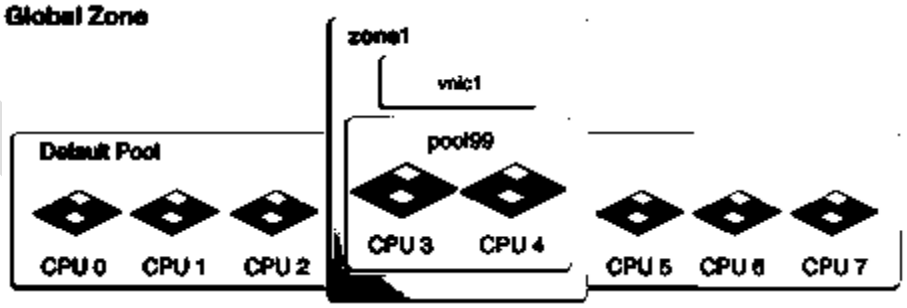


3) **Storage pools, or cloud storage device pools**: are group of file-based or block-based storage structures that contain empty and/or filled cloud storage devices.



4) **Network pools (or interconnect pools):** are group of different preconfigured network connectivity devices.
   - For example, a pool of virtual firewall devices or physical network switches can be created for redundant connectivity, load balancing, etc.

5) **CPU pools:** are group of processing units ready to be allocated to virtual servers, and are typically broken down into individual processing cores.



- Pools of physical RAM can be used in newly provisioned physical servers or to vertically scale physical servers.
- Dedicated pools can be created for each type of resource and individual pools can be grouped into a larger pool, in such case each individual pool becomes a sub-pool.
- Resource pools can become highly complex, with multiple pools created for specific cloud consumers or applications.
- A hierarchical structure can be established to form parent, sibling, and nested pools in order to facilitate the organization of diverse resource pooling requirements as shown in figure below.

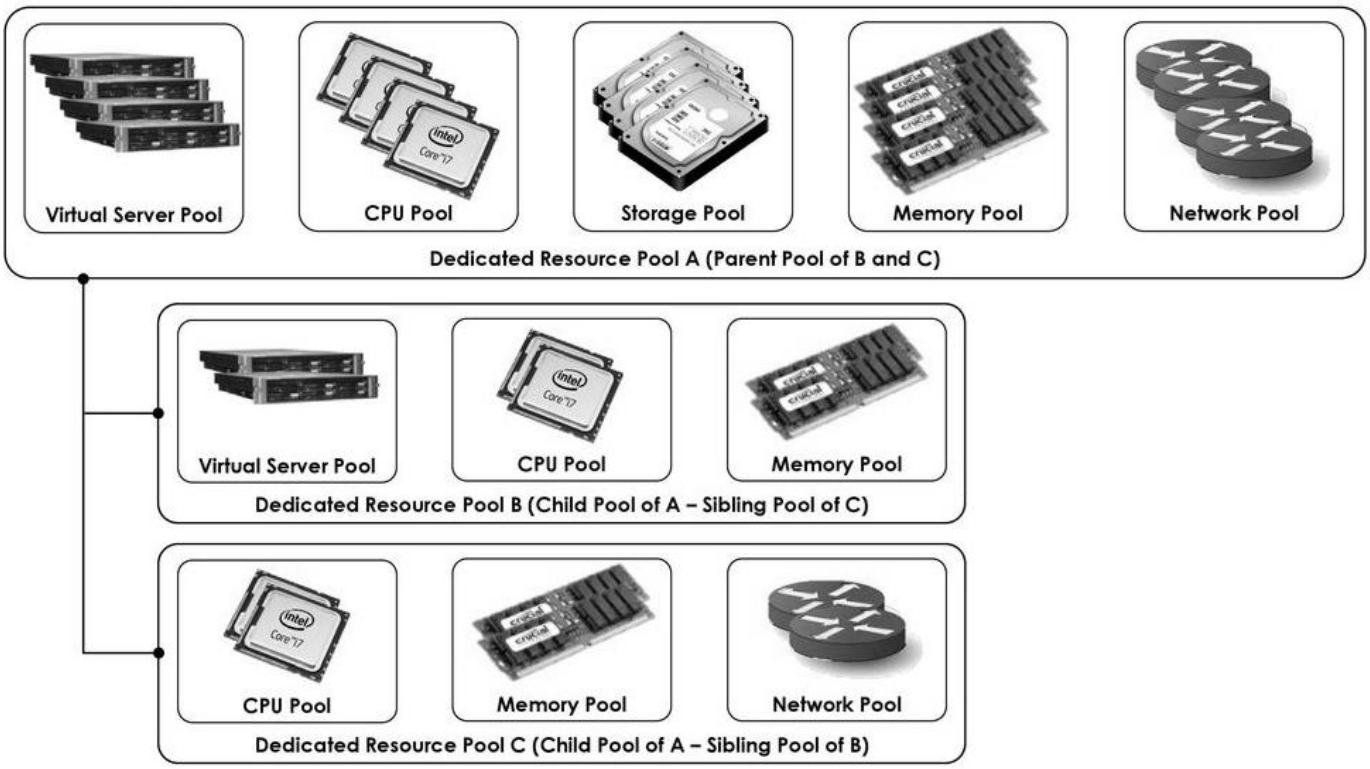


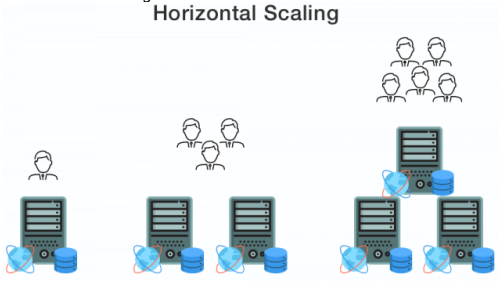
Figure: Different Pool Architecture

The *dynamic scalability architecture* is an architectural model based on a system of predefined scaling conditions that trigger the dynamic allocation of IT resources from resource pools. Dynamic allocation enables variable utilization as dictated by usage demand fluctuations, since unnecessary IT resources are efficiently reclaimed without requiring manual interaction.

The automated scaling listener is configured with workload thresholds that dictate when new IT resources need to be added to the workload processing. This mechanism can be provided with logic that determines how many additional IT resources can be dynamically provided, based on the terms of a given cloud consumer's provisioning contract.

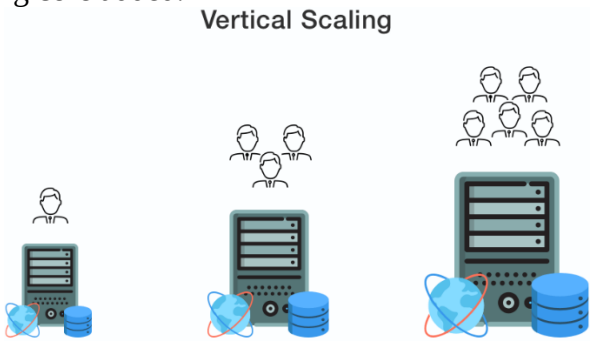
4.3. Dynamic Scalability Architecture

- This architecture is a model based on a system of predefined resource pool scaling conditions that trigger the dynamic allocation of cloud resources from pools.
- Dynamic allocation enables variable utilization as defined by usage demand fluctuations, resulting in effective resource utilization and unnecessary resources are efficiently reclaimed without requiring manual interaction.
- There are three types of dynamic scaling:
    1. *Dynamic Horizontal Scaling* – in this type the resource instances are scaled out and in to handle dynamic workloads during execution. The automatic scaling listener monitors requests and signals resource replication to initiate resource duplication, as per requirements and permissions set by administrator.



    2. *Dynamic Vertical Scaling* – in this type the resource instances are scaled up and down when there is a need to adjust the processing capacity of a single resource. For example, a

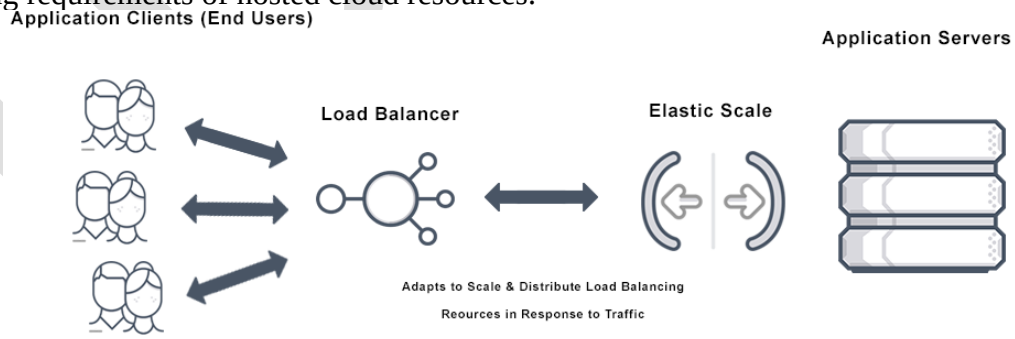
virtual server that is being overloaded can have its memory dynamically increased or it may have a processing core added.



3. *Dynamic Relocation* – in this type the resource is relocated to a host with more capacity. For example, a file may need to be moved from a tape-based SAN storage device with 4 GB per second I/O capacity to another diskbased SAN storage device with 8 GB per second I/O capacity.
    - ❖ The dynamic scalability architecture can be applied to a range of IT resources, including virtual servers and cloud storage devices. Along with the core automated scaling listener and resource replication mechanisms, the following mechanisms can also be used in this form of cloud architecture:
        - *Cloud Usage Monitor* – Specialized cloud usage monitors can track runtime usage in response to dynamic fluctuations caused by this architecture.
        - *Hypervisor* – The hypervisor is invoked by a dynamic scalability system to create or remove virtual server instances, or to be scaled itself.
        - *Pay-Per-Use Monitor* – The pay-per-use monitor is engaged to collect usage cost information in response to the scaling of IT resources.

4.4. Elastic Resource Capacity Architecture

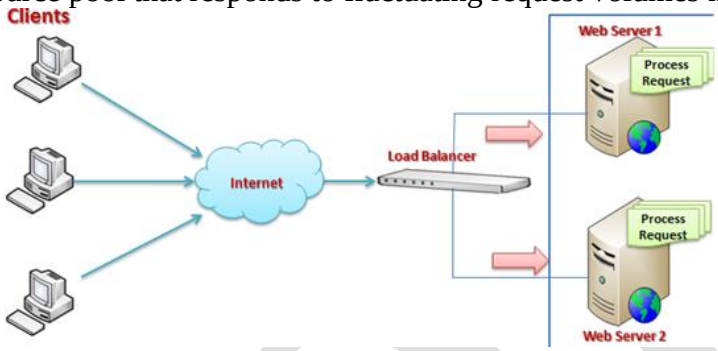
- This architecture is related to the dynamic provisioning of virtual servers, using a system that allocates and reclaims processors and memory in immediate response to the fluctuating processing requirements of hosted cloud resources.



- Resource pools are used by scaling technology that interacts with the hypervisor and/or VIM to retrieve and return CPU and RAM resources at runtime.
- The runtime processing of the virtual server is monitored so that additional processing power can be leveraged from the resource pool via dynamic allocation, before capacity thresholds are met.
- The virtual server and its hosted applications and resources are vertically scaled in response.
- This type of cloud architecture can be designed so that the intelligent automation engine script sends its scaling request via the VIM instead of to the hypervisor directly.
- Virtual servers that participate in elastic resource allocation systems may require rebooting in order for the dynamic resource allocation to take effect.

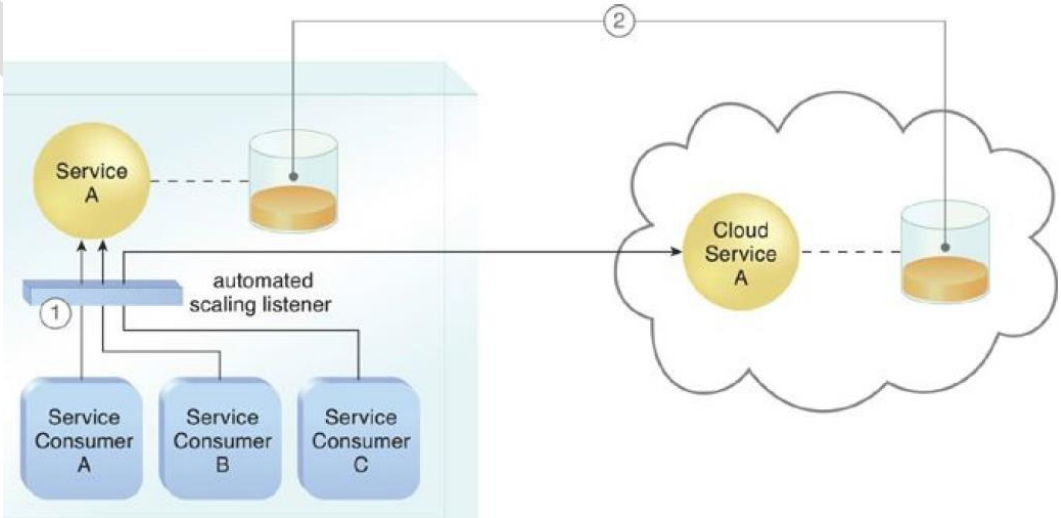
4.5. Service Load Balancing Architecture

- This architecture is a specialized variant of the workload distribution architecture that is geared specifically for scaling cloud service implementations.
- Redundant deployments of cloud services are created, with a load balancing system added to dynamically distribute workloads.
- The duplicate cloud service implementations are organized into a resource pool, while the load balancer is positioned as either an external or built-in component to allow the host servers to balance the workloads themselves.
- Depending on the anticipated workload and processing capacity of host server environments, multiple instances of each cloud service implementation can be generated as part of a resource pool that responds to fluctuating request volumes more efficiently.



4.6. Cloud Bursting Architecture

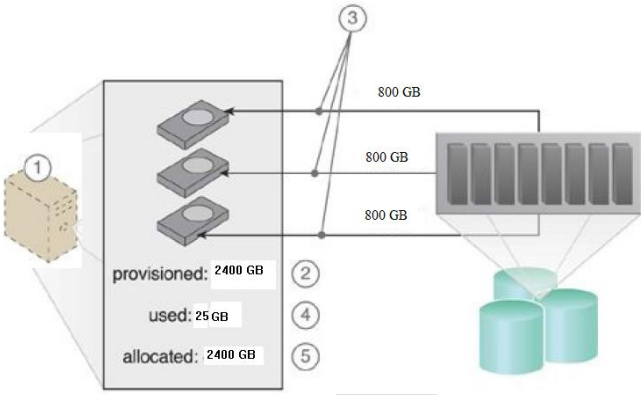
- This architecture establishes a form of dynamic scaling that scales or "bursts out" on-premise cloud resources into a cloud whenever predefined capacity thresholds have been reached.
- The corresponding cloud-based resources are redundantly pre-deployed but remain inactive until cloud bursting occurs. After they are no longer required, the resources are released and the architecture "bursts in" back to the on-premise environment.
- Cloud bursting is a flexible scaling architecture that provides cloud consumers with the option of using cloud-based IT resources only to meet higher usage demands.
- The foundation of this architectural model is based on the automated scaling listener and resource replication mechanisms.
- The automated scaling listener determines when to redirect requests to cloud based resources, and resource replication is used to maintain synchronicity between on-premise and cloud-based IT resources in relation to state information



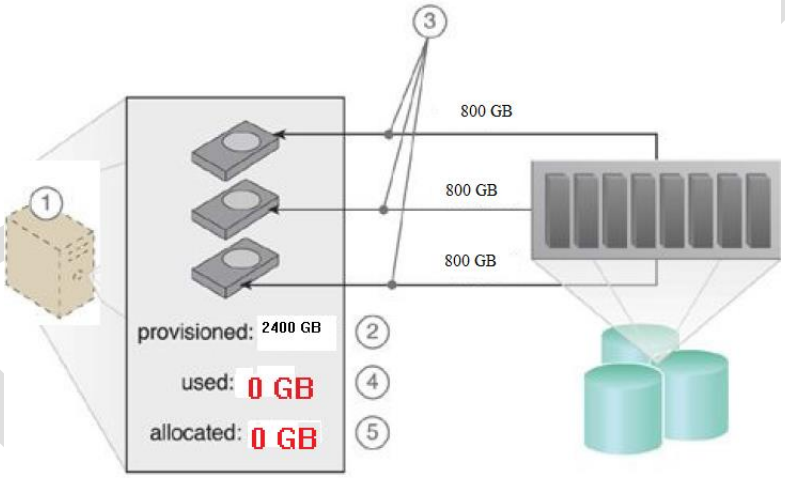
4.7. Elastic Disk Provisioning Architecture

- Cloud consumers are commonly charged for cloud-based storage space based on fixed-disk storage allocation, meaning the charges are predetermined by disk capacity and not aligned with actual data storage consumption.

- Cloud provisions a virtual server with the Windows Server 2019 OS and Three 800 GB hard drives. The cloud consumer is billed for using 2400 GB of storage space after installing the operating system, even though the operating system only requires 25 GB of storage space.



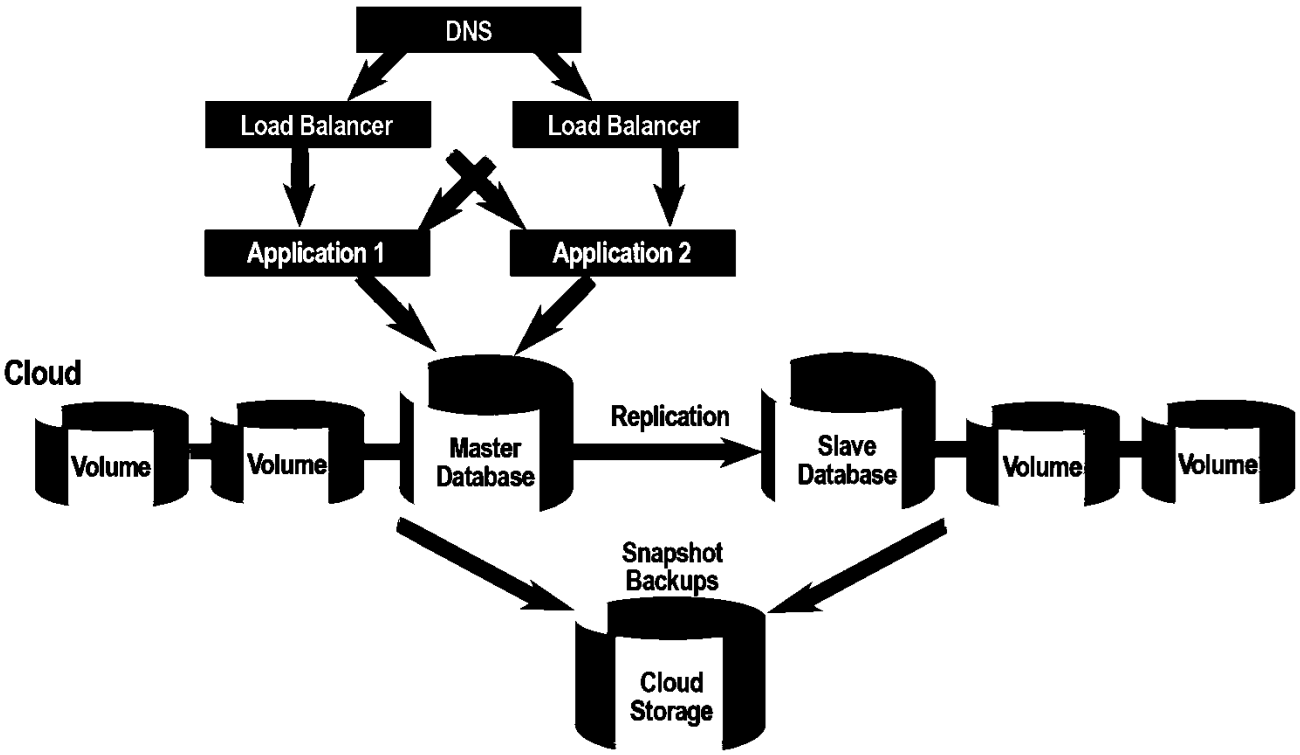
- The *elastic disk provisioning architecture* establishes a dynamic storage provisioning system that ensures that the cloud consumer is granularly billed for the exact amount of storage that it actually uses. This system uses thin provisioning technology for the dynamic allocation of storage space, and is further supported by runtime usage monitoring to collect accurate usage data for billing purposes



- Thin-provisioning software is installed on virtual servers that process dynamic storage allocation via the hypervisor, while the pay-per-use monitor tracks and reports granular billing-related disk usage data.

4.8. Redundant Storage Architecture

- Cloud storage devices are occasionally subject to failure and disruptions that are caused by network connectivity issues, controller or general hardware failure, or security breaches.
- A compromised cloud storage device's reliability can have a ripple effect and cause impact failure across all of the services, applications, and infrastructure components in the cloud that are reliant on its availability.
- The *redundant storage architecture* introduces a secondary duplicate cloud storage device as part of a failover system that synchronizes its data with the data in the primary cloud storage device.
- A storage service gateway diverts cloud consumer requests to the secondary device whenever the primary device fails.



- This cloud architecture primarily relies on a storage replication system that keeps the primary cloud storage device synchronized with its duplicate secondary cloud storage devices.
- Cloud providers may locate secondary cloud storage devices in a different geographical region than the primary cloud storage device, usually for economic reasons.
- The location of the secondary cloud storage devices can dictate the protocol and method used for synchronization, like some replication transport protocols have distance restrictions.

Chapter – 8
Advanced Cloud Architectures

Unit Structure:

8.1. Hypervisor Clustering Architecture
8.2. Load Balanced Virtual Server Instances Architecture
8.3. Non-Disruptive Service Relocation Architecture
8.4. Zero Downtime Architecture
8.5. Cloud Balancing Architecture
8.6. Resource Reservation Architecture
8.7. Dynamic Failure Detection and Recovery Architecture
8.8. Bare-Metal Provisioning Architecture
8.9. Rapid Provisioning Architecture
8.10. Storage Workload Management Architecture

Objective:

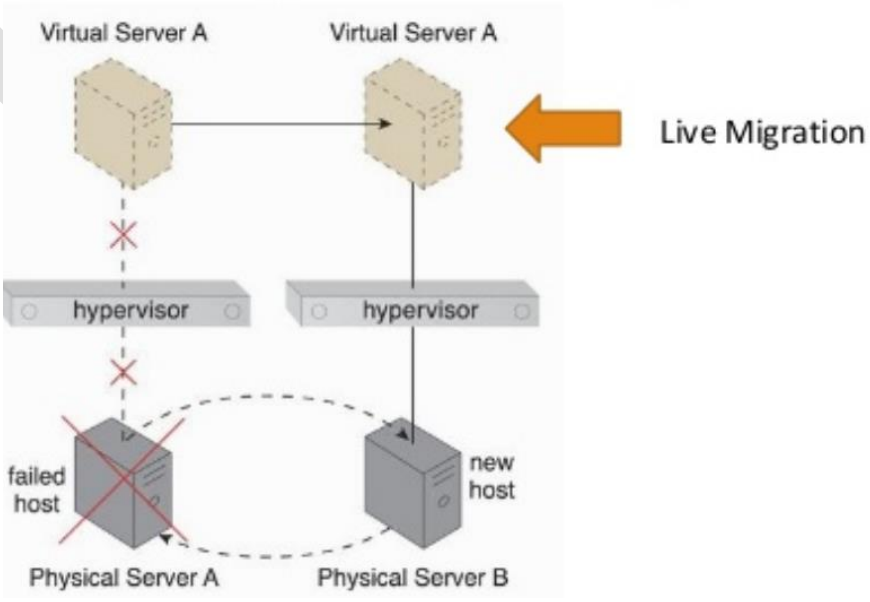
To learn how to use Advance Cloud Services.

Introduction:

This chapter introduces the cloud technology architectures distinct and sophisticated architectural layers, several of which can be built upon the more foundational environments established by the architectural models discussed in previous chapter.

8.1. Hypervisor Clustering Architecture

- Hypervisors are responsible for creating and hosting multiple virtual servers.
- Because of this dependency, any failure conditions that affect a hypervisor can cascaded effect on its virtual servers.
- The *hypervisor clustering architecture* establishes a high-availability cluster of hypervisors across multiple physical servers.
- If a given hypervisor or its underlying physical server becomes unavailable, the hosted virtual servers can be moved to another physical server or hypervisor to maintain runtime operations.

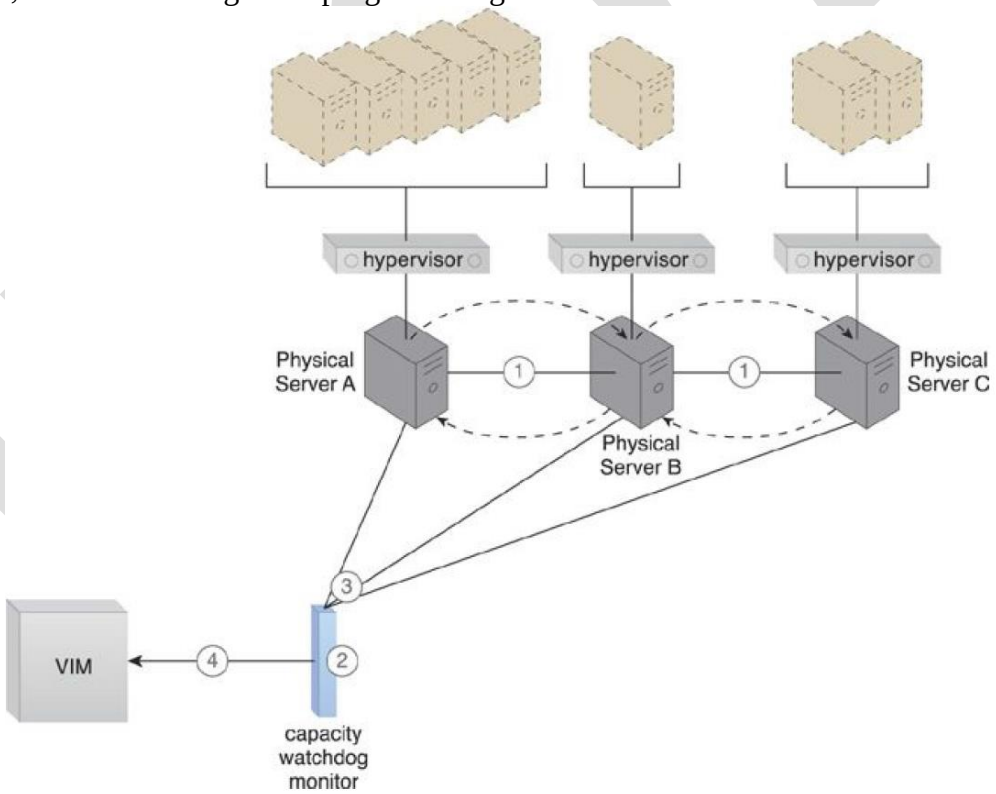


- The hypervisor cluster is controlled via a central VIM, which sends regular heartbeat messages to the hypervisors to confirm that they are up and running.

- Unacknowledged heartbeat messages cause the VIM to initiate the live VM migration program, in order to dynamically move the affected virtual servers to a new host.

8.2. Load Balanced Virtual Server Instances Architecture

- Sometime keeping cross-server workloads evenly balanced between physical servers whose operation and management are isolated can be the most challenging part on cloud.
- A physical server can easily end up hosting more virtual servers or receive larger workloads than its neighboring physical servers.
- Both physical server over and under-utilization can increase dramatically over time, leading to on-going performance challenges (for over-utilized servers) and constant waste (for the lost processing potential of under-utilized servers).
- The load balanced virtual server instances architecture establishes a capacity watchdog system that dynamically calculates virtual server instances and associated workloads, before distributing the processing across available physical server hosts.
- The capacity watchdog system is comprised of a capacity watchdog cloud usage monitor, the live VM migration program, and a capacity planner.
- The capacity watchdog monitor tracks physical and virtual server usage and reports any significant fluctuations to the capacity planner, which is responsible for dynamically calculating physical server computing capacities against virtual server capacity requirements.
- If the capacity planner decides to move a virtual server to another host to distribute the workload, the live VM migration program is signaled to move the virtual server.



8.3. Non-Disruptive Service Relocation Architecture

- A cloud service can become unavailable for a number of reasons, such as:
  - **1)** runtime usage demands that exceed its processing capacity
  - **2)** a maintenance update that mandates a temporary outage
  - **3)** permanent migration to a new physical server host
- Cloud service consumer requests are usually rejected if a cloud service becomes unavailable, which can potentially result in exception conditions.

- The *non-disruptive service relocation architecture* establishes a system by which a predefined event triggers the duplication or migration of a cloud service implementation at runtime, thereby avoiding any disruption.
- Instead of scaling cloud services in or out with redundant implementations, cloud service activity can be temporarily diverted to another hosting environment at runtime by adding a duplicate implementation onto a new host.
- Similarly, cloud service consumer requests can be temporarily redirected to a duplicate implementation when the original implementation needs to undergo a maintenance outage.
- The relocation of the cloud service implementation and any cloud service activity can also be permanent to accommodate cloud service migrations to new physical server hosts.
- A key aspect of the underlying architecture is that the new cloud service implementation is guaranteed to be successfully receiving and responding to cloud service consumer requests *before* the original cloud service implementation is deactivated or removed.
- A common approach is for live VM migration to move the entire virtual server instance that is hosting the cloud service.
- The automated scaling listener and/or load balancer mechanisms can be used to trigger a temporary redirection of cloud service consumer requests, in response to scaling and workload distribution requirements. Either mechanism can contact the VIM to initiate the live VM migration process.

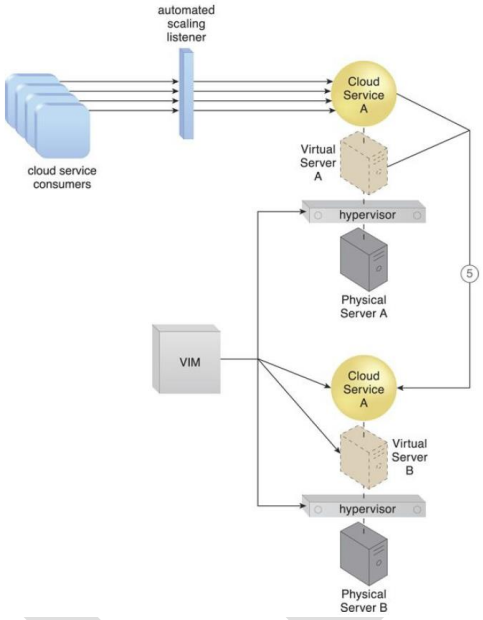


Figure : Before Failure

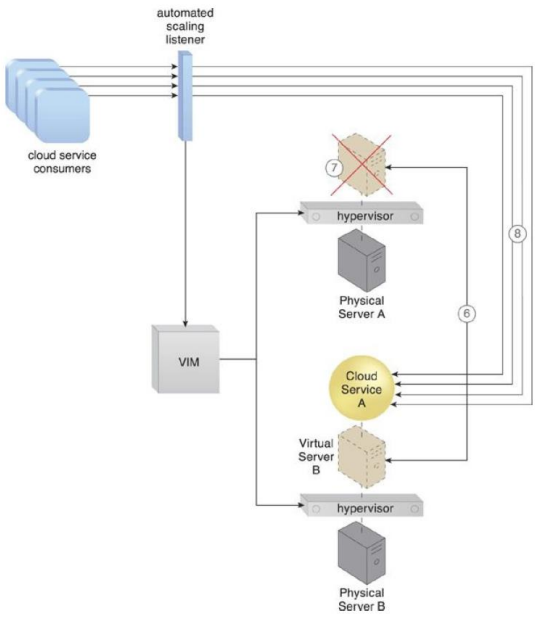


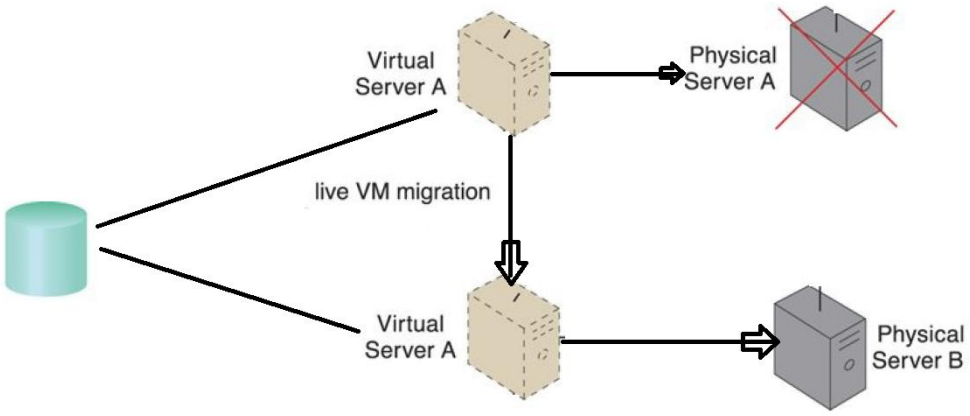
Figure : After Failure

- Virtual server migration can occur in one of the following two ways, depending on the location of the virtual server's disks and configuration:
    - A copy of the virtual server disks is created on the destination host, if the virtual server disks are stored on a local storage device or non-shared remote storage devices attached to the source host. After the copy has been created, both virtual server instances are synchronized and virtual server files are removed from the origin host.
    - Copying the virtual server disks is unnecessary if the virtual server's files are stored on a remote storage device that is shared between origin and destination hosts. Ownership of the virtual server is simply transferred from the origin to the destination physical server host, and the virtual server's state is automatically synchronized.

8.4. Zero Downtime Architecture

- A physical server naturally acts as a single point of failure for the virtual servers it hosts. As a result, when the physical server fails or is compromised, the availability of any (or all) hosted virtual servers can be affected. This makes the issuance of zero downtime guarantees by a cloud provider to cloud consumers challenging.

- The *zero downtime architecture* establishes a sophisticated failover system that allows virtual servers to be dynamically moved to different physical server hosts, in the event that their original physical server host fails



8.5. Cloud Balancing Architecture

This architecture establishes a specialized architectural model in which cloud resources can be load-balanced across multiple clouds. The cross-cloud balancing of cloud service consumer requests can help:

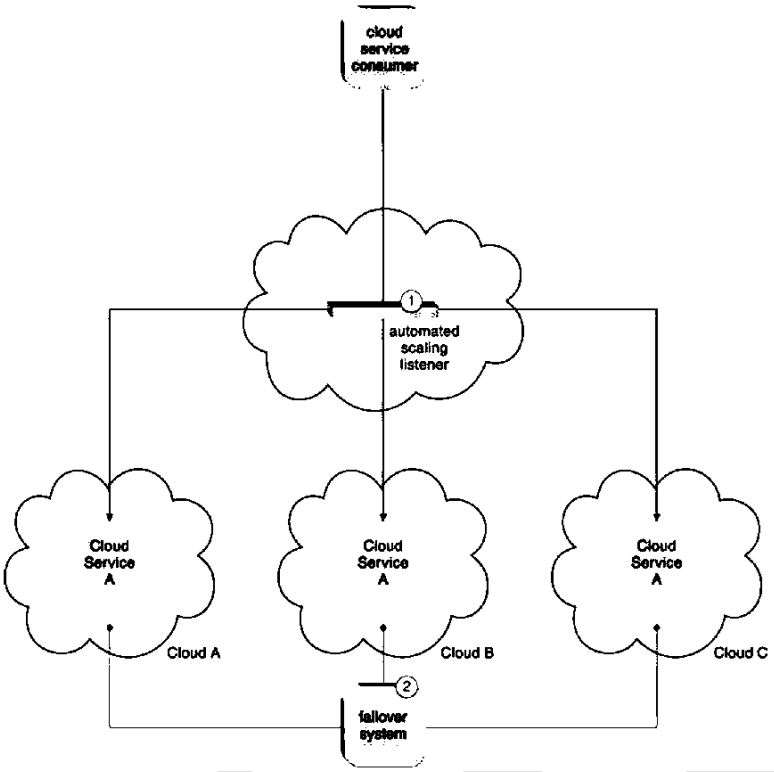
1) improve the performance and scalability of resources
2) increase the availability and reliability of resources
3) improve load-balancing and resource optimization

Its functionality is primarily based on the combination of the automated scaling listener and failover system mechanisms. Many more components and mechanisms can be part of a complete this architecture.

As a starting point, the two mechanisms are utilized as follows:

- The automated scaling listener redirects cloud service consumer requests to one of several redundant IT resource implementations, based on current scaling and performance requirements.
- The failover system ensures that redundant IT resources are capable of cross-cloud failover in the event of a failure within an IT resource or its underlying hosting environment. IT resource failures are announced so that the automated scaling listener can avoid inadvertently routing cloud service consumer requests to unavailable or unstable IT resources.

For a cloud balancing architecture to function effectively, the automated scaling listener needs to be aware of all redundant IT resource implementations within the scope of the cloud balanced architecture. Also if the manual synchronization of cross-cloud IT resource implementations is not possible, the resource replication mechanism may need to be incorporated to automate the synchronization.



8.6. Resource Reservation Architecture

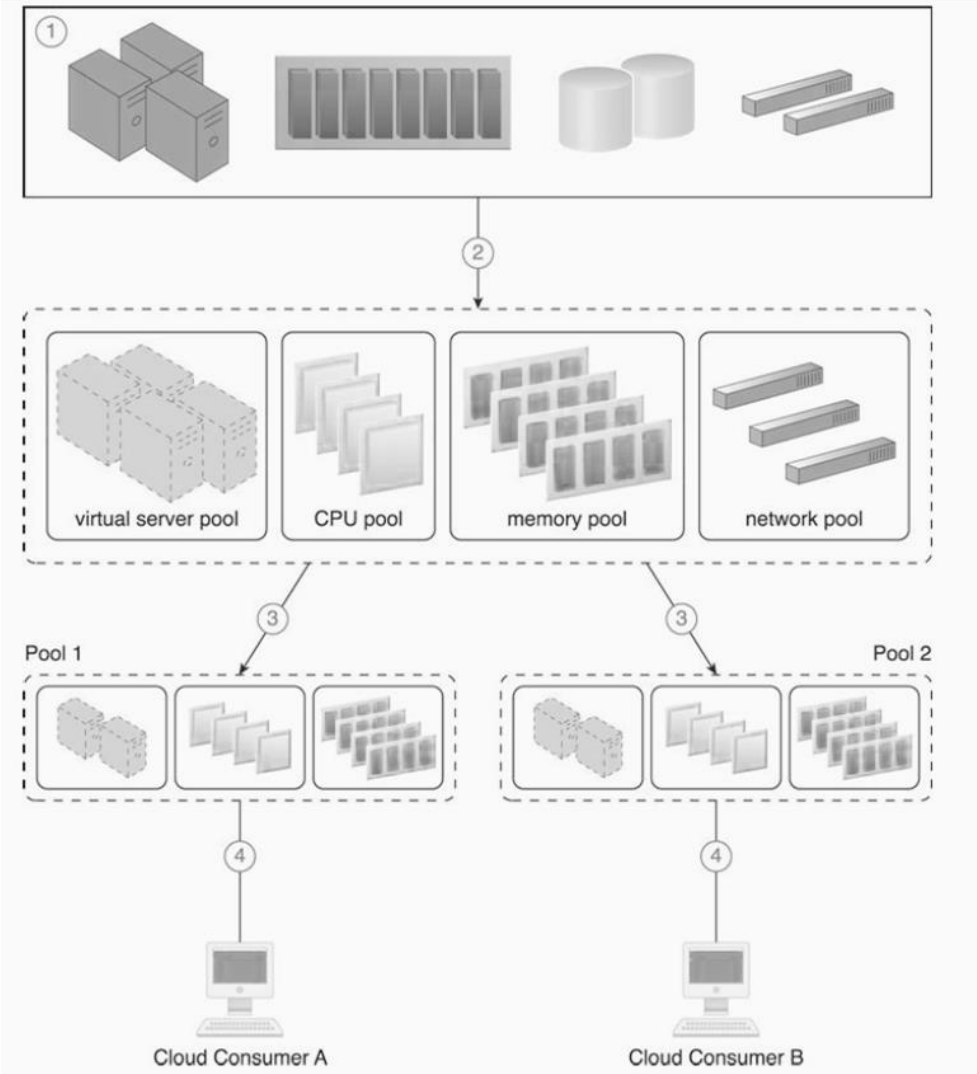
The *resource reservation architecture* establishes a system whereby one of the following is set aside exclusively for a given cloud consumer

- single resource
- part of an resource
- multiple resources

The creation of a resource reservation system can require involving the resource management system mechanism, which is used to define the usage thresholds for individual resources and resource pools. Reservations lock the amount of resources that each pool needs to keep, with the balance of the pool's resources still available for sharing and borrowing. The remote administration system mechanism is also used to enable front-end customization, so that cloud consumers have administration controls for the management of their reserved resource allocations.

The types of mechanisms that are commonly reserved within this architecture are cloud storage devices and virtual servers. Other mechanisms that may be part of the architecture can include:

- *Audit Monitor*
- *Cloud Usage Monitor*
- *Hypervisor*
- *Logical Network Perimeter*
- *Resource Replication*



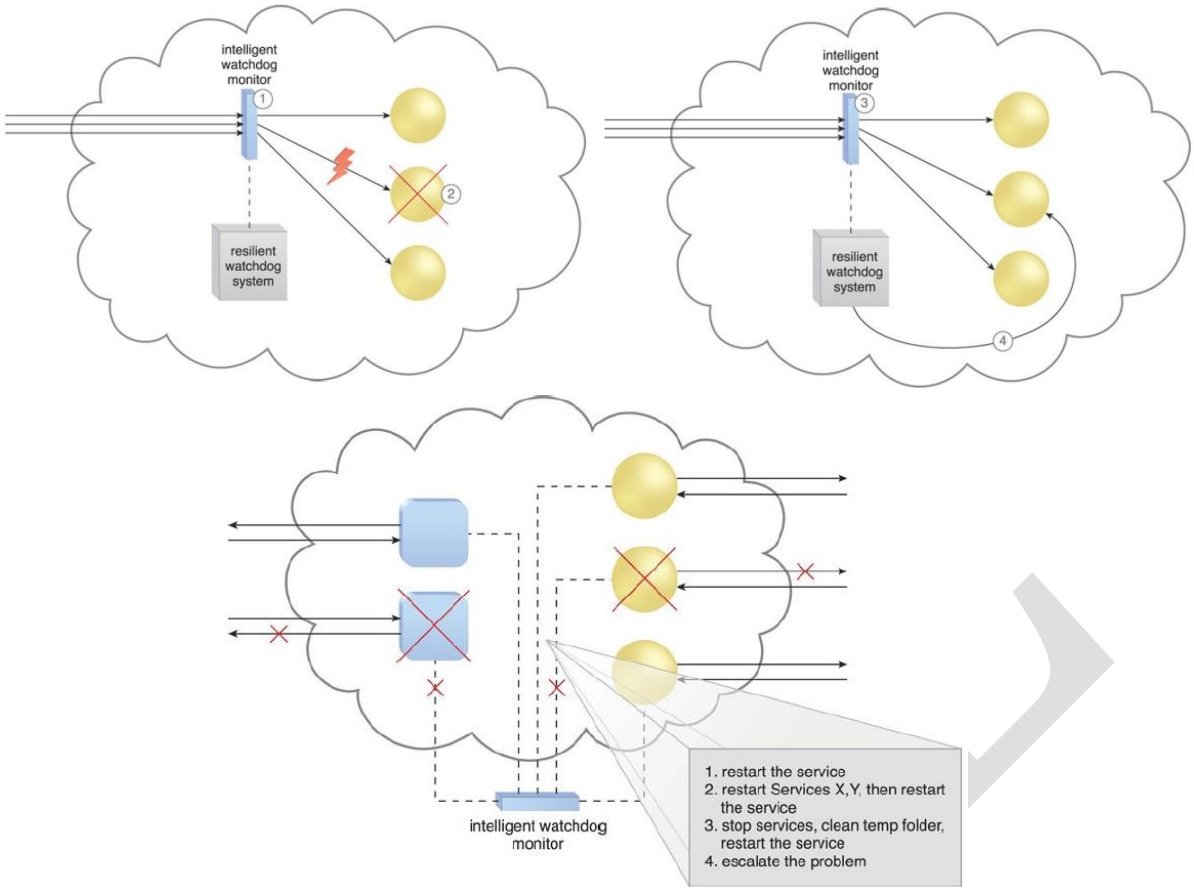
8.7. Dynamic Failure Detection and Recovery Architecture

This architecture establishes a resilient watchdog system to monitor and respond to a wide range of pre-defined failure scenarios. This system notifies and escalates the failure conditions that it cannot automatically resolve itself. It relies on a specialized cloud usage monitor called the intelligent watchdog monitor to actively track resources and take pre-defined actions in response to predefined events.

The resilient watchdog system performs the following five core functions:

1) watching
2) deciding upon an event
3) acting upon an event
4) reporting
5) escalating

Sequential recovery policies can be defined for each resource to determine the steps that the intelligent watchdog monitor needs to take when a failure condition occurs. For example, a recovery policy can state that one recovery attempt needs to be automatically carried out before issuing a notification

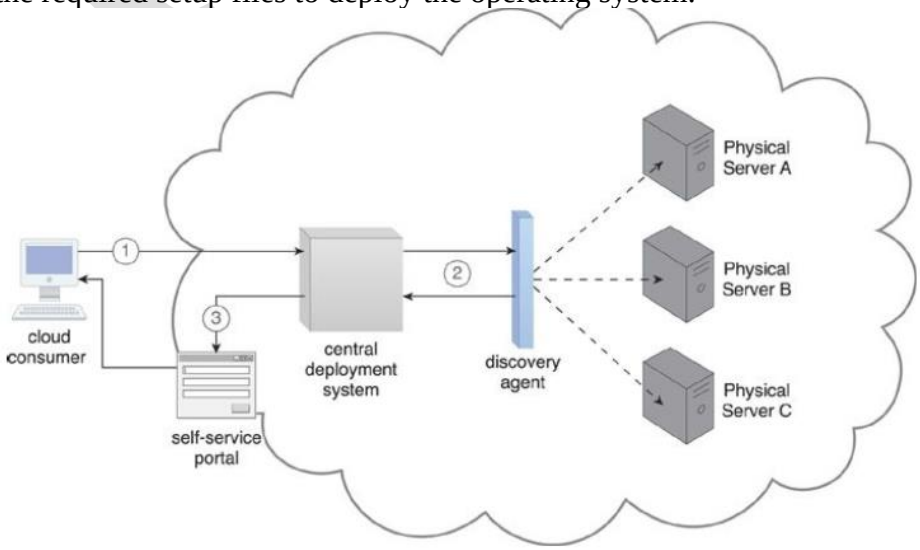


8.8. Bare-Metal Provisioning Architecture

This architecture establishes a system that utilizes this feature with specialized service agents, which are used to discover and effectively provision entire operating systems remotely.

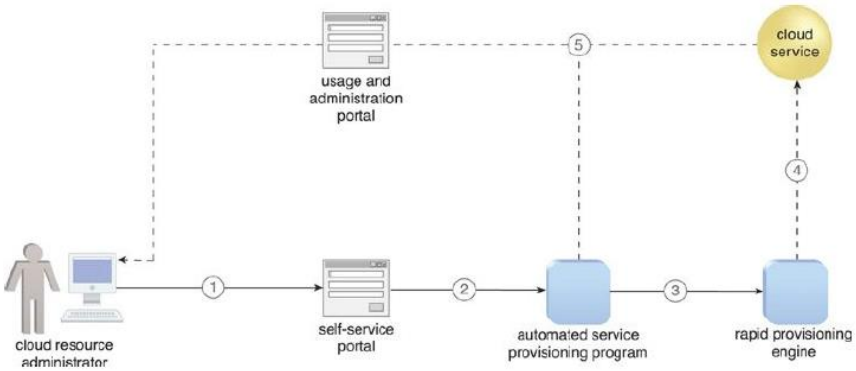
The remote management software that is integrated with the server's ROM becomes available upon server start-up. A Web-based or proprietary userinterface, like the portal provided by the remote administration system, is usually used to connect to the physical server's native remote management interface. IP addresses in IaaS platforms can be forwarded directly to cloud consumers so that they can perform bare-metal operating system installations independently.

The bare-metal provisioning system provides an auto-deployment feature that allows cloud consumers to connect to the deployment software and provision more than one server or operating system at the same time. The central deployment system connects to the servers via their management interfaces, and uses the same protocol to upload and operate as an agent in the physical server's RAM. The bare-metal server then becomes a raw client with a management agent installed, and the deployment software uploads the required setup files to deploy the operating system.



8.9. Rapid Provisioning Architecture

The *rapid provisioning architecture* establishes a system that automates the provisioning of a wide range of resources, either individually or as a collective. The underlying technology architecture for rapid resource provisioning can be sophisticated and complex, and relies on a system comprised of an automated provisioning program, rapid provisioning engine, and scripts and templates for on-demand provisioning.



(1) A cloud resource administrator requests a new cloud service through the self-service portal.
(2) The self-service portal passes the request to the automated service provisioning program installed on the virtual server.
(3) which passes the necessary tasks to be performed to the rapid provisioning engine.
(4) The rapid provisioning engine announces when the new cloud service is ready.
(5) The automated service provisioning program finalizes and publishes the cloud service on the usage and administration portal for cloud consumer access.

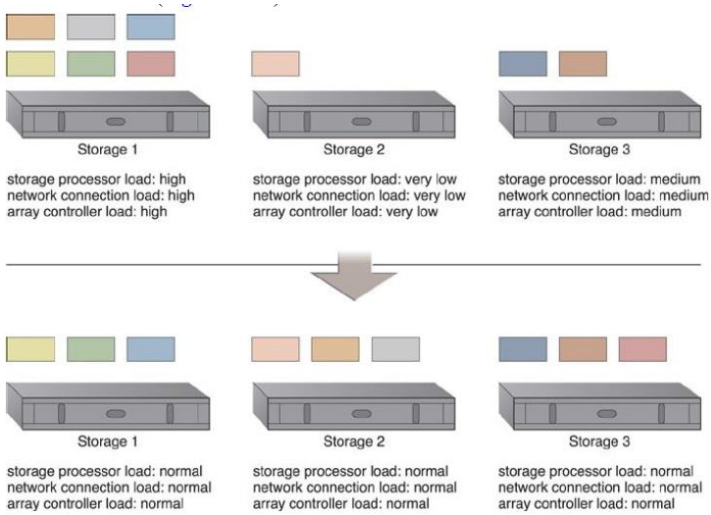
The step-by-step description describes the inner workings of a rapid provisioning engine:

1. A cloud consumer requests a new server through the self-service portal.
2. The sequence manager forwards the request to the deployment engine for the preparation of an operating system.
3. The deployment engine uses the virtual server templates for provisioning if the request is for a virtual server. Otherwise, the deployment engine sends the request to provision a physical server.
4. The pre-defined image for the requested type of operating system is used for the provisioning of the operating system, if available. Otherwise, the regular deployment process is executed to install the operating system.
5. The deployment engine informs the sequence manager when the operating system is ready.
6. The sequence manager updates and sends the logs to the sequence logger for storage.
7. The sequence manager requests that the deployment engine apply the operating system baseline to the provisioned operating system.
8. The deployment engine applies the requested operating system baseline.
9. The deployment engine informs the sequence manager that the operating system baseline has been applied.
10. The sequence manager updates and sends the logs of completed steps to the sequence logger for storage.
11. The sequence manager requests that the deployment engine install the applications.
12. The deployment engine deploys the applications on the provisioned server.
13. The deployment engine informs the sequence manager that the applications have been installed.
14. The sequence manager updates and sends the logs of completed steps to the sequence logger for storage.
15. The sequence manager requests that the deployment engine apply the application's configuration baseline.

16. The deployment engine applies the configuration baseline.
17. The deployment engine informs the sequence manager that the configuration baseline has been applied.
18. The sequence manager updates and sends the logs of completed steps to the sequence logger for storage.

8.10. Storage Workload Management Architecture

This architecture enables LUNs to be evenly distributed across available cloud storage devices, while a storage capacity system is established to ensure that runtime workloads are evenly distributed across the LUNs.



Combining cloud storage devices into a group allows LUN data to be distributed between available storage hosts equally. A storage management system is configured and an automated scaling listener is positioned to monitor and equalize runtime workloads among the grouped cloud storage devices.

Glossary:

- **Intelligent Automation Engine:** The intelligent automation engine automates administration tasks by executing scripts that contain workflow logic.
- **LUN:** A logical unit number (LUN) is a logical drive that represents a partition of a physical drive.
- **Storage Service Gateway:** The storage service gateway is a component that acts as the external interface to cloud storage services, and is capable of automatically redirecting cloud consumer requests whenever the location of the requested data has changed.
- **Storage Replication:** Storage replication is a variation of the resource replication mechanisms used to synchronously or asynchronously replicate data from a primary storage device to a secondary storage device. It can be used to replicate partial and entire LUNs.
- **Heartbeats:** Heartbeats are system-level messages exchanged between hypervisors, hypervisors and virtual servers, and hypervisors and VIMs.
- **Live VM migration:** Live VM migration is a system that is capable of relocating virtual servers or virtual server instances at runtime.
- **LUN migration:** LUN migration is a specialized storage program that is used to move LUNs from one storage device to another without interruption, while remaining transparent to cloud consumers.

References:

Cloud Computing Concepts, Technology & Architecture
- Thomas Erl, Zaigham Mahmood, and Ricardo Puttini – Prentice Hall - 2013
Chapter 11: Fundamental Cloud Architectures
Chapter 12: Advanced Cloud Architectures

Mastering Cloud Computing Foundations and Applications Programming
- Rajkumar Buyya, Christian Vecchiola, S. Thamarai Selvi - Elsevier – 2013

Distributed and Cloud Computing, From Parallel Processing to the Internet of Things
- Kai Hwang, Jack Dongarra, Geoffrey Fox – MK Publishers - 2012

Unit 4
Chapter – 1
Fundamental Cloud Architectures

Unit Structure:

4.1.1 Workload Distribution Architecture
4.2.1 Resource Pooling Architecture
4.3.1 Dynamic Scalability Architecture
4.4.1 Elastic Resource Capacity Architecture
4.5.1 Service Load Balancing Architecture
4.6.1 Cloud Bursting Architecture
4.7.1 Elastic Disk Provisioning Architecture
4.8.1 Redundant Storage Architecture

Objective:

To learn how to use Cloud Services.

Introduction:

This chapter introduces and describes several of the more common foundational cloud architectural models, each explaining a common usage and characteristic of modern day cloud-based environments. Further the chapter also explores the involvement and importance of different combinations of cloud computing mechanisms in relation to these architectures.

4.1. Workload Distribution Architecture

- Resources on cloud can be horizontally scald using an addition or identical resource and a load balancer that is capable of providing run time distribution of workload among resources.
- This architecture of distribution has a dual advantage
  - **i.** Reduces overutilization of resources.
  - **ii.** Reduces underutilization of resources.

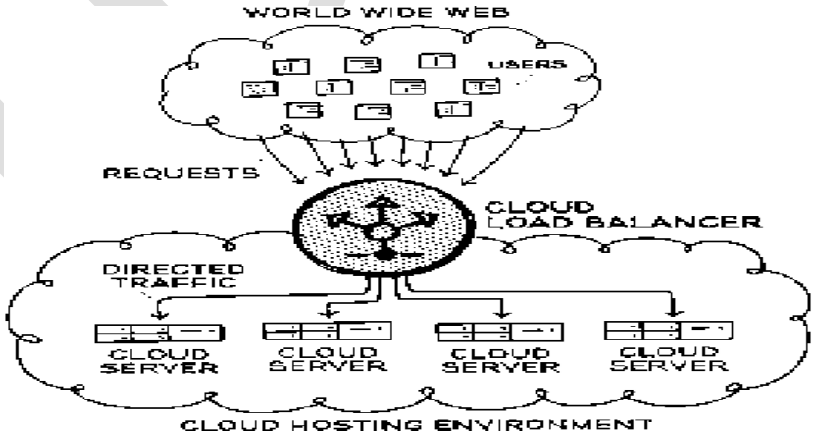


Figure: Workload Distribution Architecture

- Workload distribution is carried out in support of distributed virtual servers, storage devices and services.
- Load balancing system produces specialized variation that incorporates the aspect of load balancing like:
  - **i.** Load Balanced Service Instances Architecture

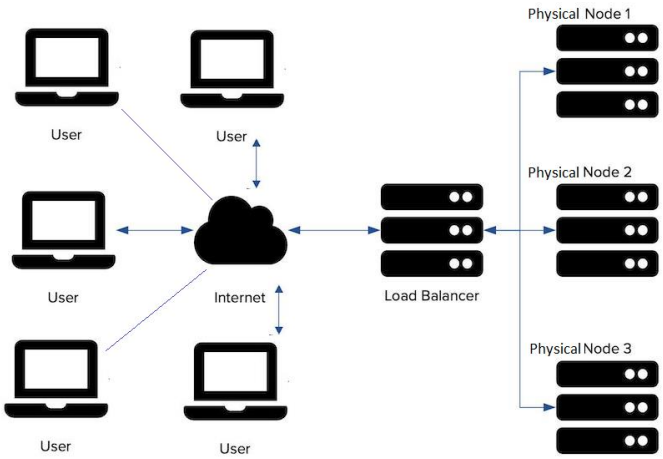
ii. Load Balanced Virtual Server Instances Architecture
iii. Load Balanced Virtual Switches Architecture

- In addition to the above mentioned mechanism, the following mechanisms can also be part of this cloud architecture:
  i. *Audit Monitor* – Resources that process the data can determine whether monitoring is necessary to fulfill legal and regulatory requirements.
  ii. *Cloud Usage Monitor* – Various monitors can be involved to carry out runtime workload tracking and data processing.
  iii. *Hypervisor* – Workloads between hypervisors and the virtual servers that they host may require distribution.
  iv. *Logical Network Perimeter* – The logical network perimeter isolates cloud consumer network boundaries in relation to how and where workloads are distributed.
  v. *Resource Cluster* – Clustered IT resources in active/active mode are commonly used to support workload balancing between different cluster nodes.
  vi. *Resource Replication* – This mechanism can generate new instances of virtualized IT resources in response to runtime workload distribution demands.

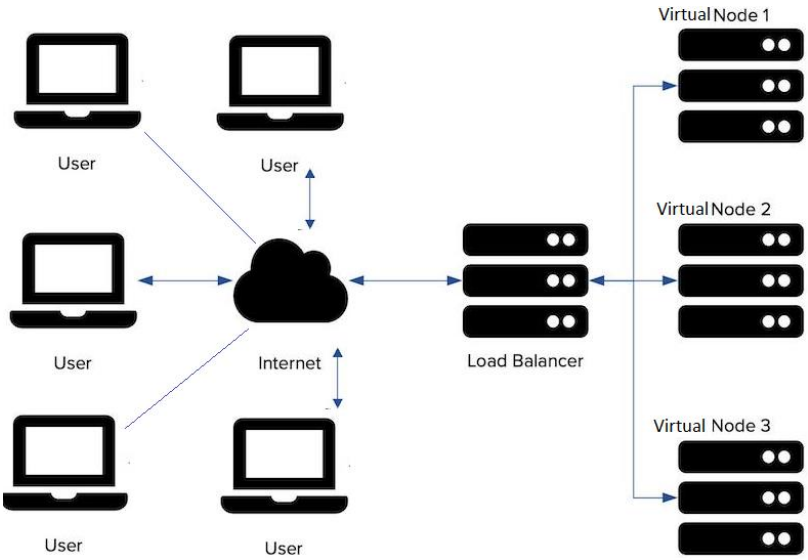
.

4.2. Resource Pooling Architecture

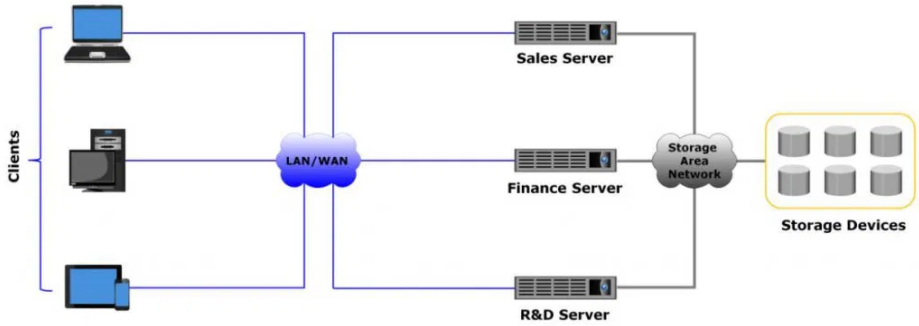
- This architecture is based on the use of one or more resource from a pool of resources, in which identical synchronized resources are grouped and maintained by a system. Examples of resource pools:
  1) **Physical server pools:** are group of physical servers networked to have installed operating systems and other necessary programs and/or applications and are ready for immediate use.



  2) **Virtual server pools:** are group of virtual servers networked to have installed operating systems and other necessary programs and/or applications and are ready for immediate use. They are usually configured using one of several available templates chosen by the cloud consumer during provisioning.
     - For example, a cloud consumer can set up a pool of mid-tier Windows servers with 4 GB of RAM or a pool of low-tier Ubuntu servers with 2 GB of RAM.

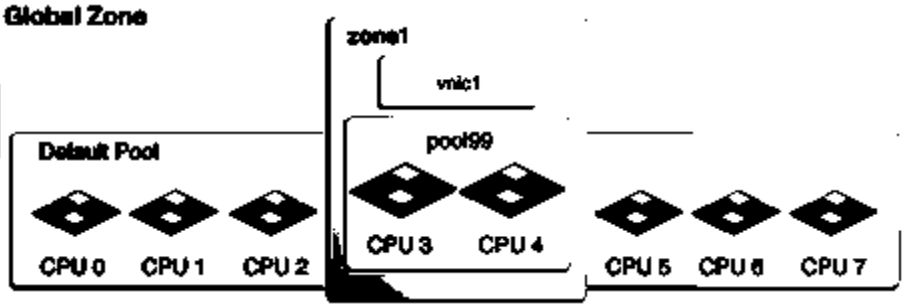


3) **Storage pools, or cloud storage device pools**: are group of file-based or block-based storage structures that contain empty and/or filled cloud storage devices.



4) **Network pools (or interconnect pools):** are group of different preconfigured network connectivity devices.
   - For example, a pool of virtual firewall devices or physical network switches can be created for redundant connectivity, load balancing, etc.

5) **CPU pools:** are group of processing units ready to be allocated to virtual servers, and are typically broken down into individual processing cores.



- Pools of physical RAM can be used in newly provisioned physical servers or to vertically scale physical servers.
- Dedicated pools can be created for each type of resource and individual pools can be grouped into a larger pool, in such case each individual pool becomes a sub-pool.
- Resource pools can become highly complex, with multiple pools created for specific cloud consumers or applications.
- A hierarchical structure can be established to form parent, sibling, and nested pools in order to facilitate the organization of diverse resource pooling requirements as shown in figure below.

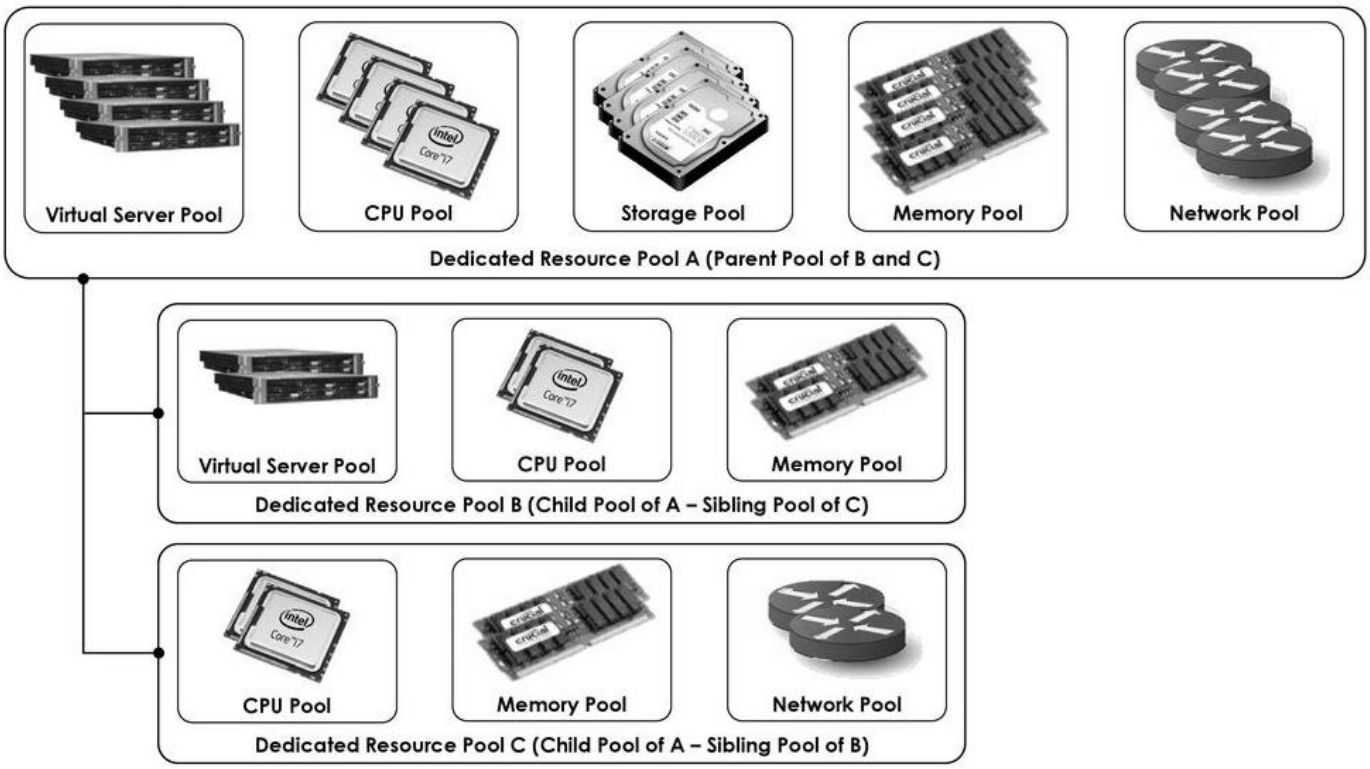


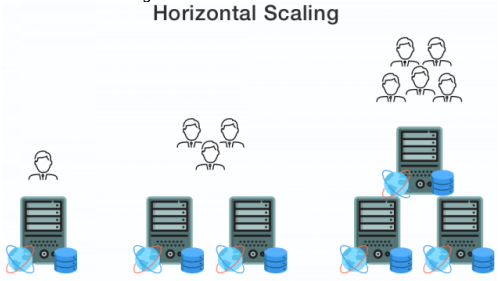
Figure: Different Pool Architecture

The *dynamic scalability architecture* is an architectural model based on a system of predefined scaling conditions that trigger the dynamic allocation of IT resources from resource pools. Dynamic allocation enables variable utilization as dictated by usage demand fluctuations, since unnecessary IT resources are efficiently reclaimed without requiring manual interaction.

The automated scaling listener is configured with workload thresholds that dictate when new IT resources need to be added to the workload processing. This mechanism can be provided with logic that determines how many additional IT resources can be dynamically provided, based on the terms of a given cloud consumer's provisioning contract.

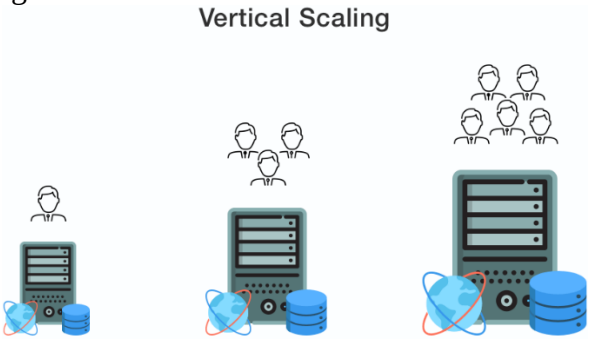
4.3. Dynamic Scalability Architecture

- This architecture is a model based on a system of predefined resource pool scaling conditions that trigger the dynamic allocation of cloud resources from pools.
- Dynamic allocation enables variable utilization as defined by usage demand fluctuations, resulting in effective resource utilization and unnecessary resources are efficiently reclaimed without requiring manual interaction.
- There are three types of dynamic scaling:
    1. *Dynamic Horizontal Scaling* – in this type the resource instances are scaled out and in to handle dynamic workloads during execution. The automatic scaling listener monitors requests and signals resource replication to initiate resource duplication, as per requirements and permissions set by administrator.



    2. *Dynamic Vertical Scaling* – in this type the resource instances are scaled up and down when there is a need to adjust the processing capacity of a single resource. For example, a

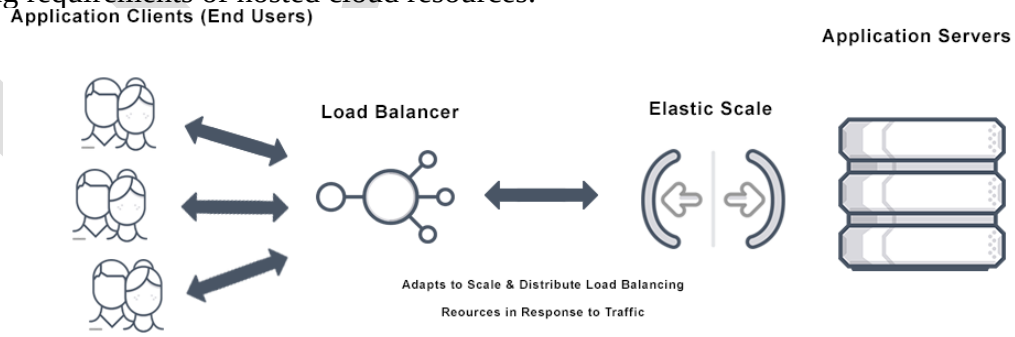
virtual server that is being overloaded can have its memory dynamically increased or it may have a processing core added.



3. *Dynamic Relocation* – in this type the resource is relocated to a host with more capacity. For example, a file may need to be moved from a tape-based SAN storage device with 4 GB per second I/O capacity to another diskbased SAN storage device with 8 GB per second I/O capacity.
    - ❖ The dynamic scalability architecture can be applied to a range of IT resources, including virtual servers and cloud storage devices. Along with the core automated scaling listener and resource replication mechanisms, the following mechanisms can also be used in this form of cloud architecture:
        - *Cloud Usage Monitor* – Specialized cloud usage monitors can track runtime usage in response to dynamic fluctuations caused by this architecture.
        - *Hypervisor* – The hypervisor is invoked by a dynamic scalability system to create or remove virtual server instances, or to be scaled itself.
        - *Pay-Per-Use Monitor* – The pay-per-use monitor is engaged to collect usage cost information in response to the scaling of IT resources.

4.4. Elastic Resource Capacity Architecture

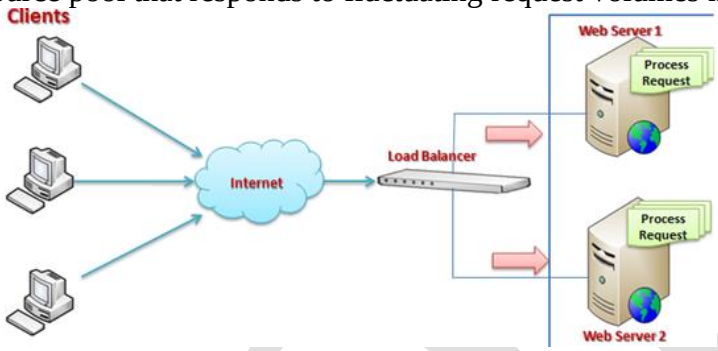
- This architecture is related to the dynamic provisioning of virtual servers, using a system that allocates and reclaims processors and memory in immediate response to the fluctuating processing requirements of hosted cloud resources.



- Resource pools are used by scaling technology that interacts with the hypervisor and/or VIM to retrieve and return CPU and RAM resources at runtime.
- The runtime processing of the virtual server is monitored so that additional processing power can be leveraged from the resource pool via dynamic allocation, before capacity thresholds are met.
- The virtual server and its hosted applications and resources are vertically scaled in response.
- This type of cloud architecture can be designed so that the intelligent automation engine script sends its scaling request via the VIM instead of to the hypervisor directly.
- Virtual servers that participate in elastic resource allocation systems may require rebooting in order for the dynamic resource allocation to take effect.

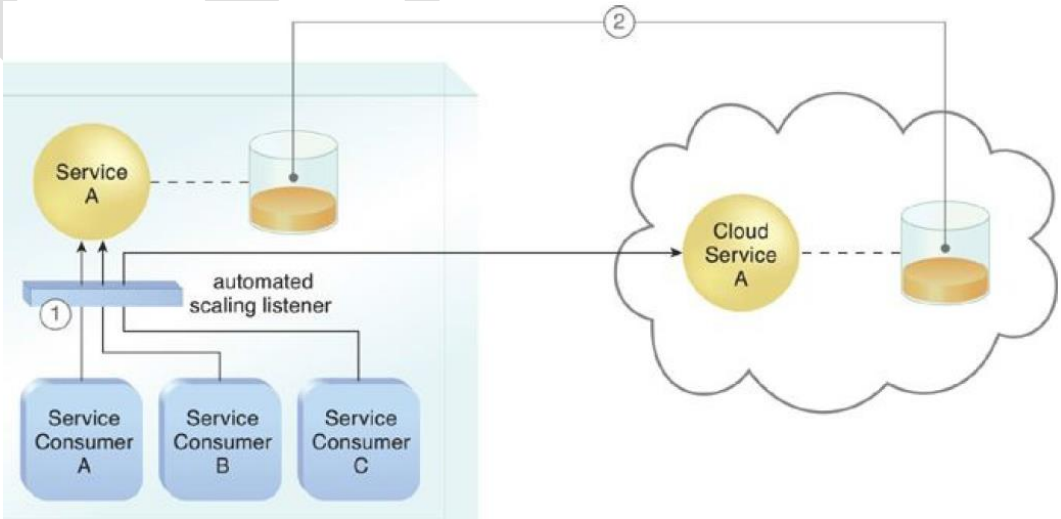
4.5. Service Load Balancing Architecture

- This architecture is a specialized variant of the workload distribution architecture that is geared specifically for scaling cloud service implementations.
- Redundant deployments of cloud services are created, with a load balancing system added to dynamically distribute workloads.
- The duplicate cloud service implementations are organized into a resource pool, while the load balancer is positioned as either an external or built-in component to allow the host servers to balance the workloads themselves.
- Depending on the anticipated workload and processing capacity of host server environments, multiple instances of each cloud service implementation can be generated as part of a resource pool that responds to fluctuating request volumes more efficiently.



4.6. Cloud Bursting Architecture

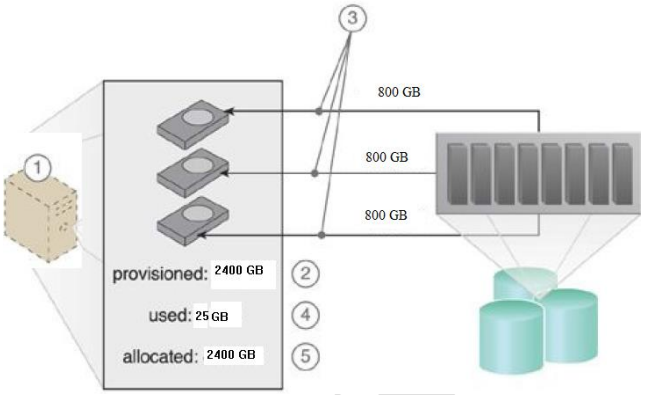
- This architecture establishes a form of dynamic scaling that scales or "bursts out" on-premise cloud resources into a cloud whenever predefined capacity thresholds have been reached.
- The corresponding cloud-based resources are redundantly pre-deployed but remain inactive until cloud bursting occurs. After they are no longer required, the resources are released and the architecture "bursts in" back to the on-premise environment.
- Cloud bursting is a flexible scaling architecture that provides cloud consumers with the option of using cloud-based IT resources only to meet higher usage demands.
- The foundation of this architectural model is based on the automated scaling listener and resource replication mechanisms.
- The automated scaling listener determines when to redirect requests to cloud based resources, and resource replication is used to maintain synchronicity between on-premise and cloud-based IT resources in relation to state information



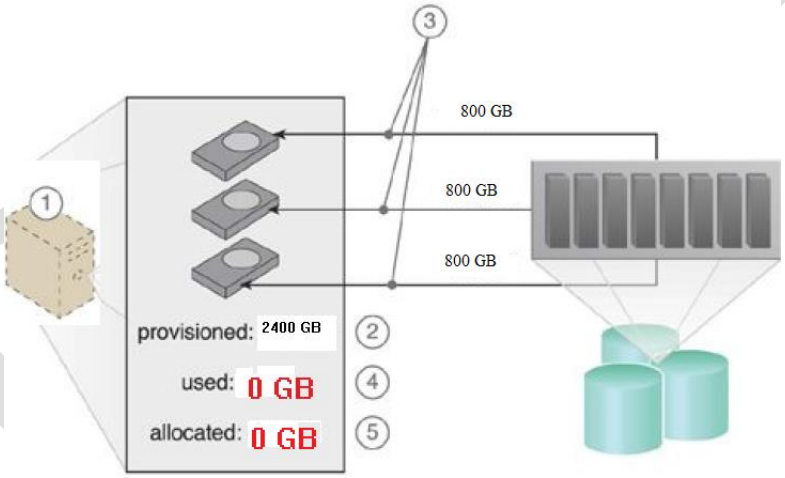
4.7. Elastic Disk Provisioning Architecture

- Cloud consumers are commonly charged for cloud-based storage space based on fixed-disk storage allocation, meaning the charges are predetermined by disk capacity and not aligned with actual data storage consumption.

- Cloud provisions a virtual server with the Windows Server 2019 OS and Three 800 GB hard drives. The cloud consumer is billed for using 2400 GB of storage space after installing the operating system, even though the operating system only requires 25 GB of storage space.



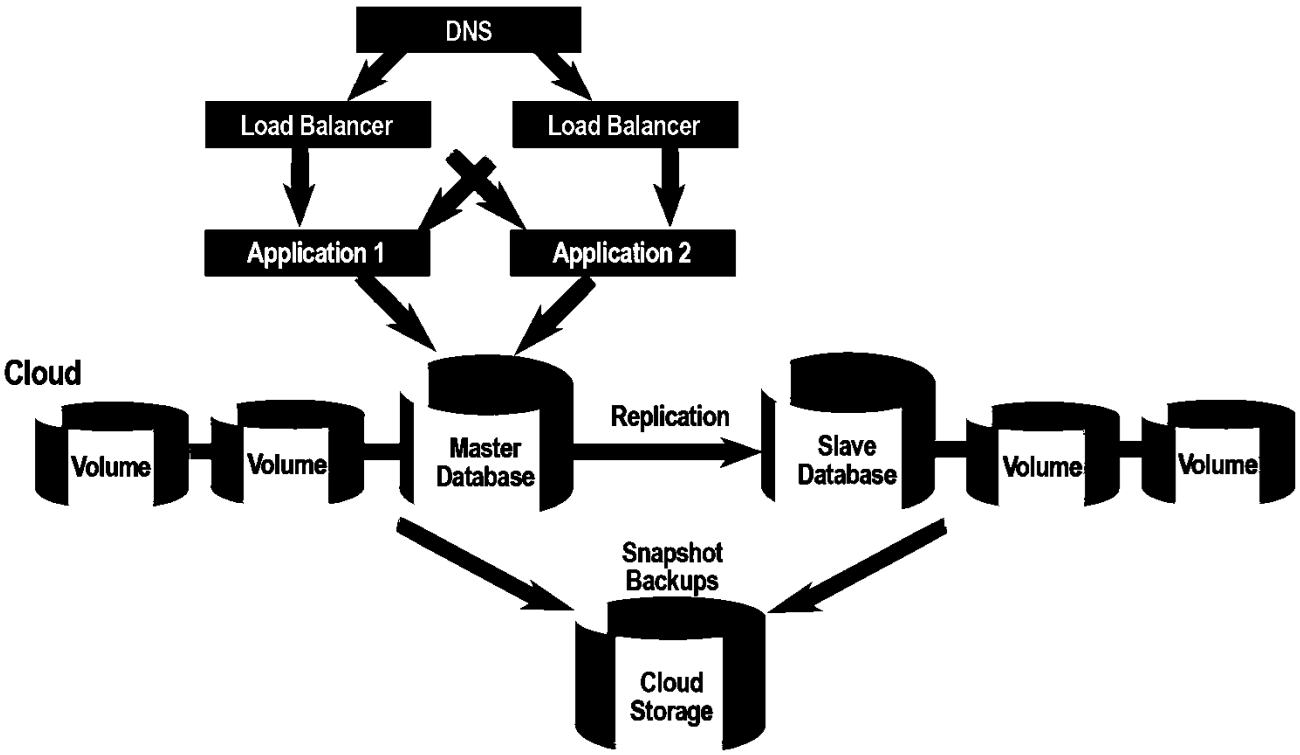
- The *elastic disk provisioning architecture* establishes a dynamic storage provisioning system that ensures that the cloud consumer is granularly billed for the exact amount of storage that it actually uses. This system uses thin provisioning technology for the dynamic allocation of storage space, and is further supported by runtime usage monitoring to collect accurate usage data for billing purposes



- Thin-provisioning software is installed on virtual servers that process dynamic storage allocation via the hypervisor, while the pay-per-use monitor tracks and reports granular billing-related disk usage data.

4.8. Redundant Storage Architecture

- Cloud storage devices are occasionally subject to failure and disruptions that are caused by network connectivity issues, controller or general hardware failure, or security breaches.
- A compromised cloud storage device's reliability can have a ripple effect and cause impact failure across all of the services, applications, and infrastructure components in the cloud that are reliant on its availability.
- The *redundant storage architecture* introduces a secondary duplicate cloud storage device as part of a failover system that synchronizes its data with the data in the primary cloud storage device.
- A storage service gateway diverts cloud consumer requests to the secondary device whenever the primary device fails.



- This cloud architecture primarily relies on a storage replication system that keeps the primary cloud storage device synchronized with its duplicate secondary cloud storage devices.
- Cloud providers may locate secondary cloud storage devices in a different geographical region than the primary cloud storage device, usually for economic reasons.
- The location of the secondary cloud storage devices can dictate the protocol and method used for synchronization, like some replication transport protocols have distance restrictions.

Chapter – 2
Advanced Cloud Architectures

Unit Structure:

4.2.1. Hypervisor Clustering Architecture
4.2.2. Load Balanced Virtual Server Instances Architecture
4.2.3. Non-Disruptive Service Relocation Architecture
4.2.4. Zero Downtime Architecture
4.2.5. Cloud Balancing Architecture
4.2.6. Resource Reservation Architecture
4.2.7. Dynamic Failure Detection and Recovery Architecture
4.2.8. Bare-Metal Provisioning Architecture
4.2.9. Rapid Provisioning Architecture
4.2.10. Storage Workload Management Architecture

Objective:

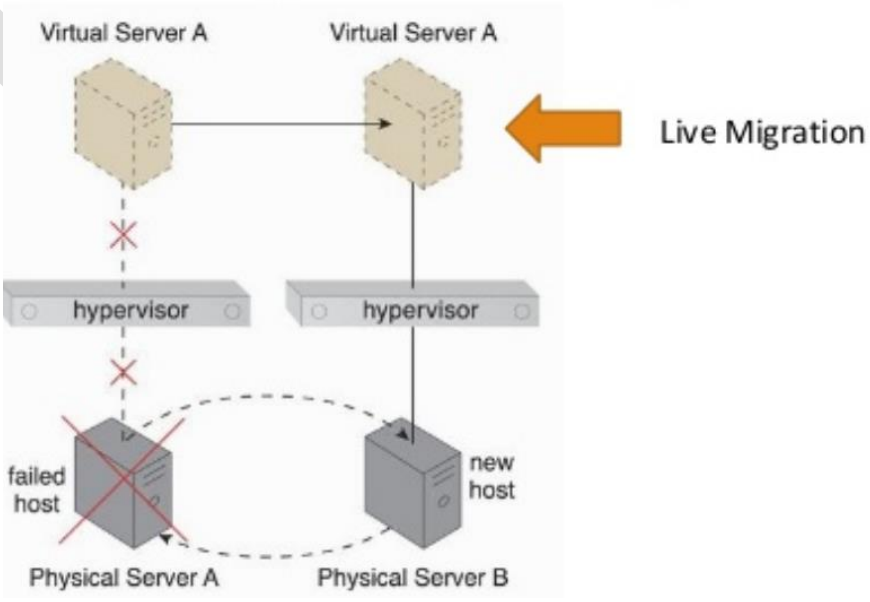
To learn how to use Advance Cloud Services.

Introduction:

This chapter introduces the cloud technology architectures distinct and sophisticated architectural layers, several of which can be built upon the more foundational environments established by the architectural models discussed in previous chapter.

4.2.1 Hypervisor Clustering Architecture

- Hypervisors are responsible for creating and hosting multiple virtual servers.
- Because of this dependency, any failure conditions that affect a hypervisor can cascaded effect on its virtual servers.
- The *hypervisor clustering architecture* establishes a high-availability cluster of hypervisors across multiple physical servers.
- If a given hypervisor or its underlying physical server becomes unavailable, the hosted virtual servers can be moved to another physical server or hypervisor to maintain runtime operations.

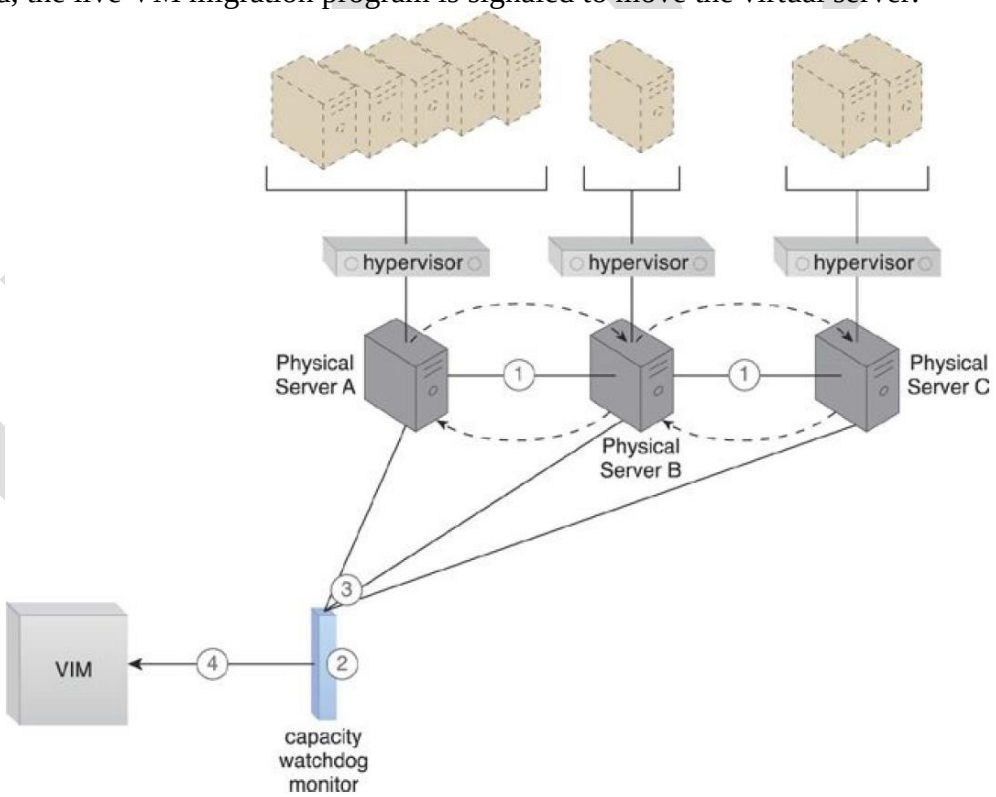


- The hypervisor cluster is controlled via a central VIM, which sends regular heartbeat messages to the hypervisors to confirm that they are up and running.

- Unacknowledged heartbeat messages cause the VIM to initiate the live VM migration program, in order to dynamically move the affected virtual servers to a new host.

4.2.2 Load Balanced Virtual Server Instances Architecture

- Sometime keeping cross-server workloads evenly balanced between physical servers whose operation and management are isolated can be the most challenging part on cloud.
- A physical server can easily end up hosting more virtual servers or receive larger workloads than its neighboring physical servers.
- Both physical server over and under-utilization can increase dramatically over time, leading to on-going performance challenges (for over-utilized servers) and constant waste (for the lost processing potential of under-utilized servers).
- The load balanced virtual server instances architecture establishes a capacity watchdog system that dynamically calculates virtual server instances and associated workloads, before distributing the processing across available physical server hosts.
- The capacity watchdog system is comprised of a capacity watchdog cloud usage monitor, the live VM migration program, and a capacity planner.
- The capacity watchdog monitor tracks physical and virtual server usage and reports any significant fluctuations to the capacity planner, which is responsible for dynamically calculating physical server computing capacities against virtual server capacity requirements.
- If the capacity planner decides to move a virtual server to another host to distribute the workload, the live VM migration program is signaled to move the virtual server.



4.2.3 Non-Disruptive Service Relocation Architecture

- A cloud service can become unavailable for a number of reasons, such as:
  - **1)** runtime usage demands that exceed its processing capacity
  - **2)** a maintenance update that mandates a temporary outage
  - **3)** permanent migration to a new physical server host
- Cloud service consumer requests are usually rejected if a cloud service becomes unavailable, which can potentially result in exception conditions.

- The *non-disruptive service relocation architecture* establishes a system by which a predefined event triggers the duplication or migration of a cloud service implementation at runtime, thereby avoiding any disruption.
- Instead of scaling cloud services in or out with redundant implementations, cloud service activity can be temporarily diverted to another hosting environment at runtime by adding a duplicate implementation onto a new host.
- Similarly, cloud service consumer requests can be temporarily redirected to a duplicate implementation when the original implementation needs to undergo a maintenance outage.
- The relocation of the cloud service implementation and any cloud service activity can also be permanent to accommodate cloud service migrations to new physical server hosts.
- A key aspect of the underlying architecture is that the new cloud service implementation is guaranteed to be successfully receiving and responding to cloud service consumer requests *before* the original cloud service implementation is deactivated or removed.
- A common approach is for live VM migration to move the entire virtual server instance that is hosting the cloud service.
- The automated scaling listener and/or load balancer mechanisms can be used to trigger a temporary redirection of cloud service consumer requests, in response to scaling and workload distribution requirements. Either mechanism can contact the VIM to initiate the live VM migration process.

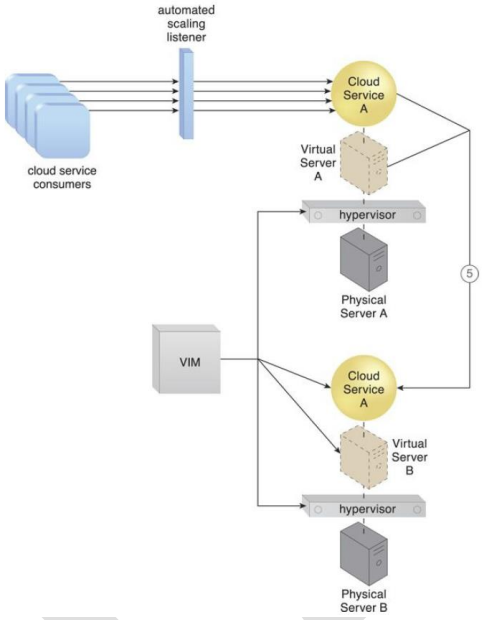


Figure : Before Failure

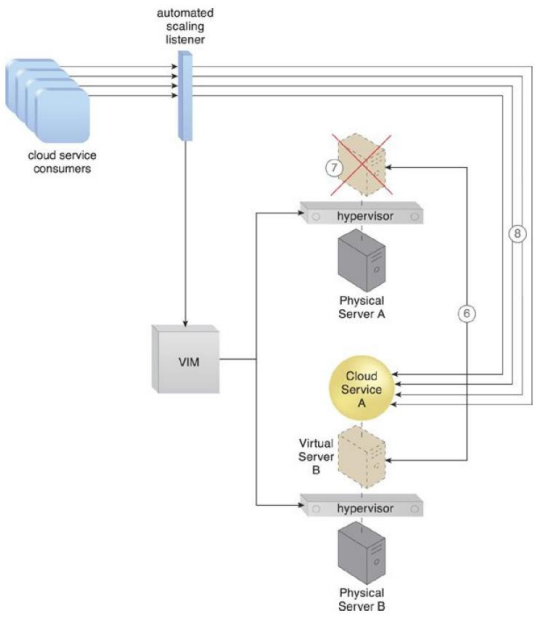


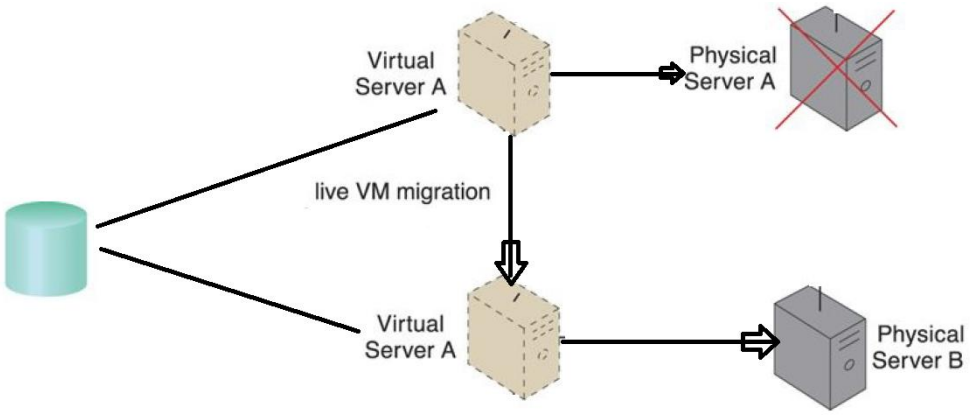
Figure : After Failure

- Virtual server migration can occur in one of the following two ways, depending on the location of the virtual server's disks and configuration:
    - A copy of the virtual server disks is created on the destination host, if the virtual server disks are stored on a local storage device or non-shared remote storage devices attached to the source host. After the copy has been created, both virtual server instances are synchronized and virtual server files are removed from the origin host.
    - Copying the virtual server disks is unnecessary if the virtual server's files are stored on a remote storage device that is shared between origin and destination hosts. Ownership of the virtual server is simply transferred from the origin to the destination physical server host, and the virtual server's state is automatically synchronized.

4.2.4 Zero Downtime Architecture

- A physical server naturally acts as a single point of failure for the virtual servers it hosts. As a result, when the physical server fails or is compromised, the availability of any (or all) hosted virtual servers can be affected. This makes the issuance of zero downtime guarantees by a cloud provider to cloud consumers challenging.

- The *zero downtime architecture* establishes a sophisticated failover system that allows virtual servers to be dynamically moved to different physical server hosts, in the event that their original physical server host fails



4.2.5 Cloud Balancing Architecture

This architecture establishes a specialized architectural model in which cloud resources can be load-balanced across multiple clouds. The cross-cloud balancing of cloud service consumer requests can help:

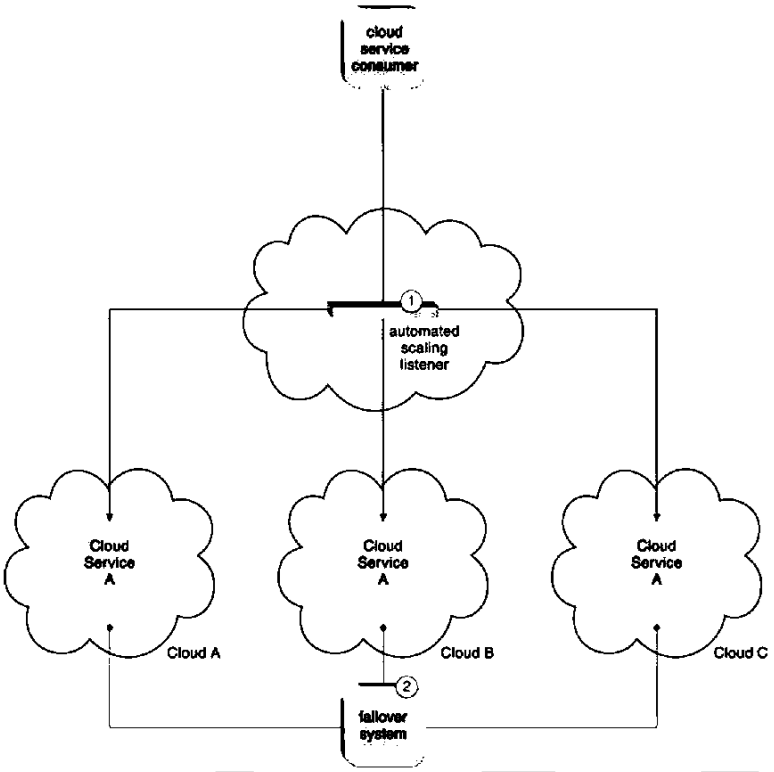
1) improve the performance and scalability of resources
2) increase the availability and reliability of resources
3) improve load-balancing and resource optimization

Its functionality is primarily based on the combination of the automated scaling listener and failover system mechanisms. Many more components and mechanisms can be part of a complete this architecture.

As a starting point, the two mechanisms are utilized as follows:

- The automated scaling listener redirects cloud service consumer requests to one of several redundant IT resource implementations, based on current scaling and performance requirements.
- The failover system ensures that redundant IT resources are capable of cross-cloud failover in the event of a failure within an IT resource or its underlying hosting environment. IT resource failures are announced so that the automated scaling listener can avoid inadvertently routing cloud service consumer requests to unavailable or unstable IT resources.

For a cloud balancing architecture to function effectively, the automated scaling listener needs to be aware of all redundant IT resource implementations within the scope of the cloud balanced architecture. Also if the manual synchronization of cross-cloud IT resource implementations is not possible, the resource replication mechanism may need to be incorporated to automate the synchronization.



4.2.6 Resource Reservation Architecture

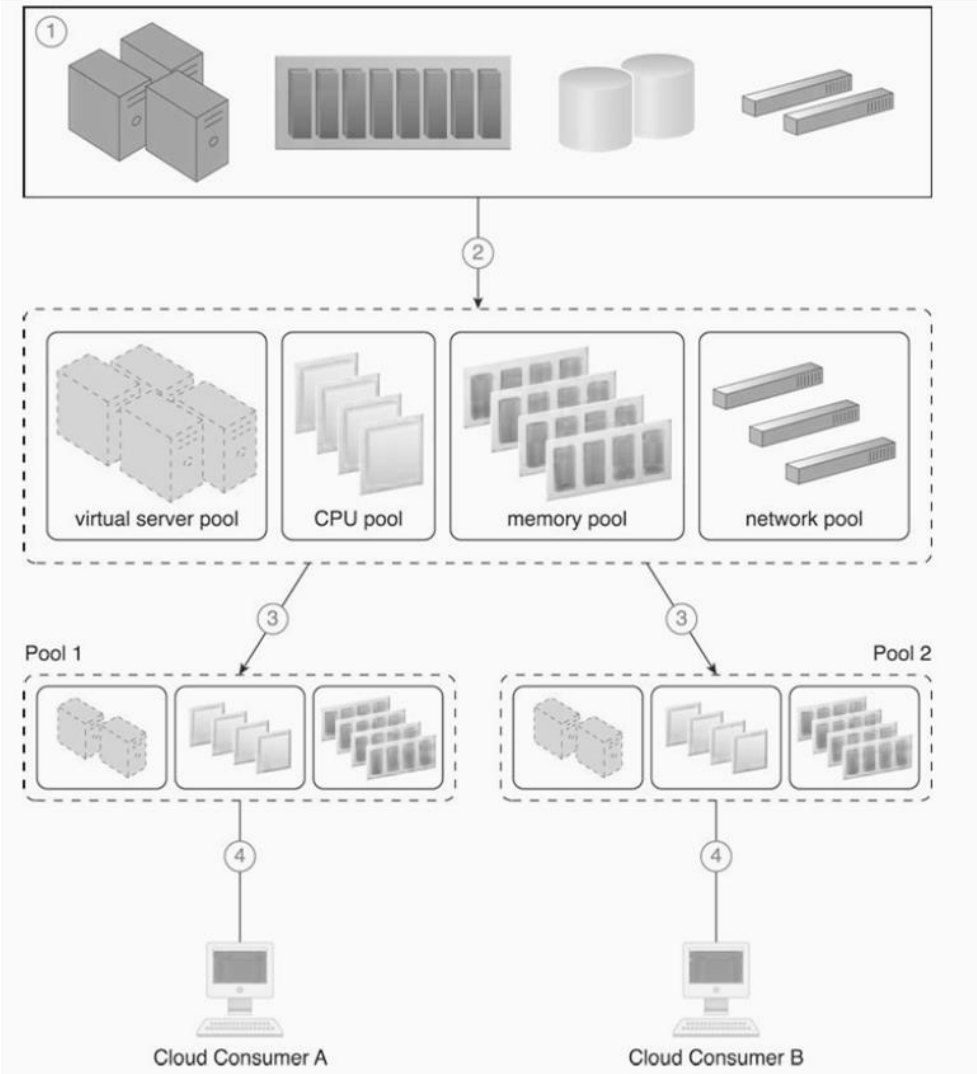
The *resource reservation architecture* establishes a system whereby one of the following is set aside exclusively for a given cloud consumer

- single resource
- part of an resource
- multiple resources

The creation of a resource reservation system can require involving the resource management system mechanism, which is used to define the usage thresholds for individual resources and resource pools. Reservations lock the amount of resources that each pool needs to keep, with the balance of the pool's resources still available for sharing and borrowing. The remote administration system mechanism is also used to enable front-end customization, so that cloud consumers have administration controls for the management of their reserved resource allocations.

The types of mechanisms that are commonly reserved within this architecture are cloud storage devices and virtual servers. Other mechanisms that may be part of the architecture can include:

- *Audit Monitor*
- *Cloud Usage Monitor*
- *Hypervisor*
- *Logical Network Perimeter*
- *Resource Replication*



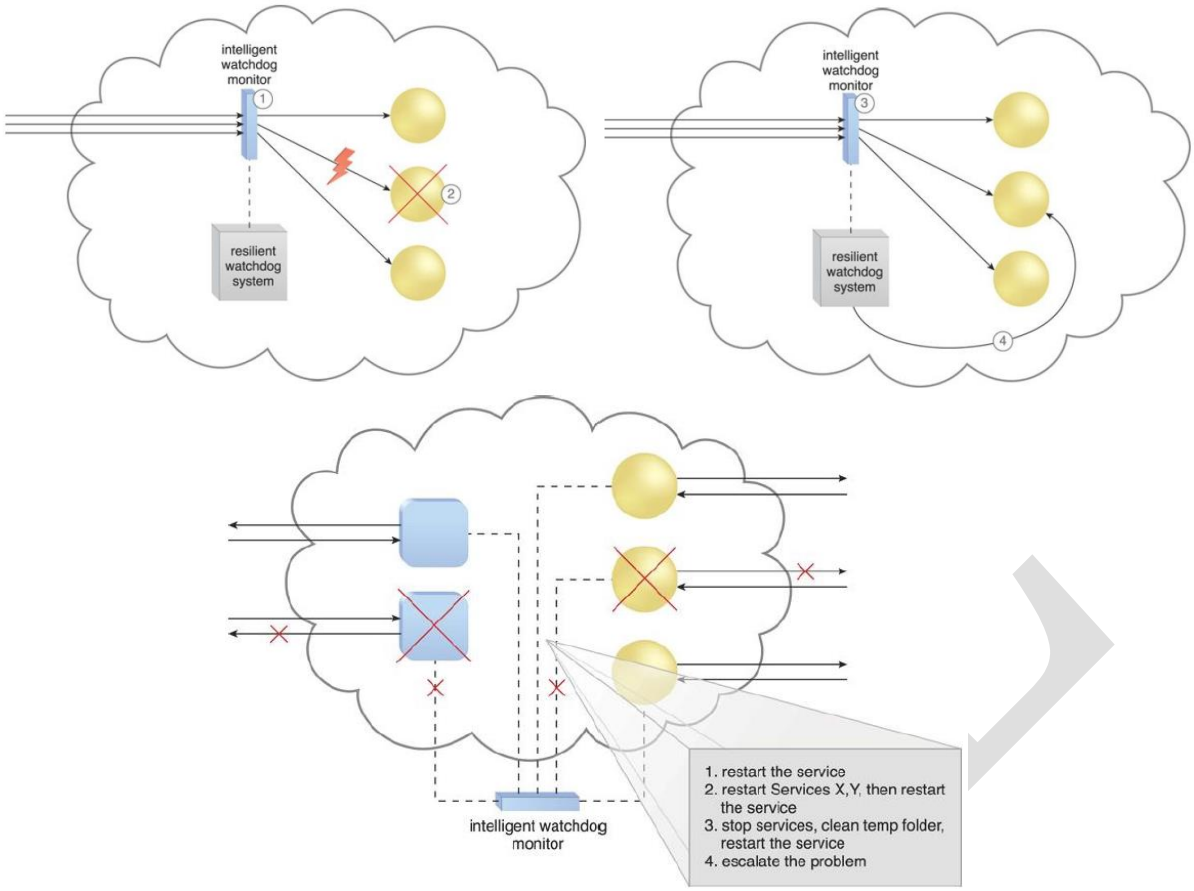
4.2.7 Dynamic Failure Detection and Recovery Architecture

This architecture establishes a resilient watchdog system to monitor and respond to a wide range of pre-defined failure scenarios. This system notifies and escalates the failure conditions that it cannot automatically resolve itself. It relies on a specialized cloud usage monitor called the intelligent watchdog monitor to actively track resources and take pre-defined actions in response to predefined events.

The resilient watchdog system performs the following five core functions:

1) watching
2) deciding upon an event
3) acting upon an event
4) reporting
5) escalating

Sequential recovery policies can be defined for each resource to determine the steps that the intelligent watchdog monitor needs to take when a failure condition occurs. For example, a recovery policy can state that one recovery attempt needs to be automatically carried out before issuing a notification

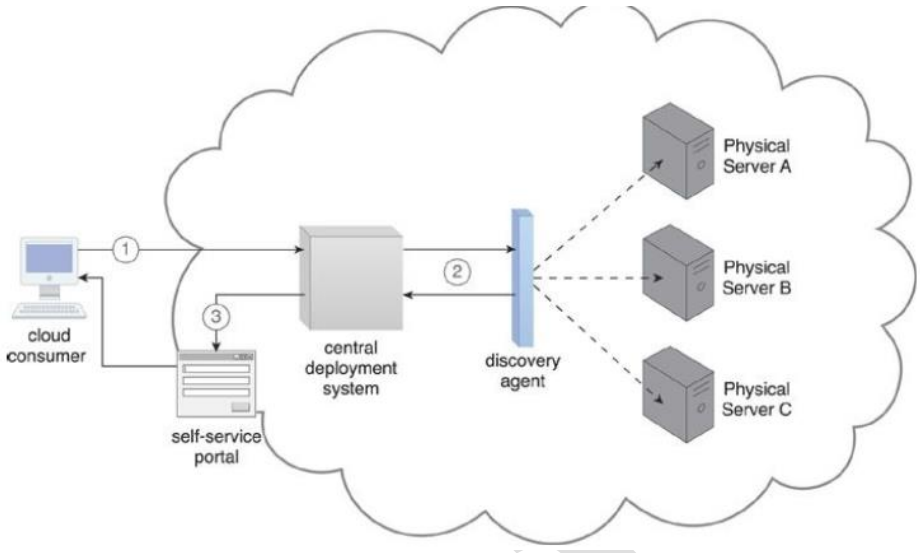


4.2.8 Bare-Metal Provisioning Architecture

This architecture establishes a system that utilizes this feature with specialized service agents, which are used to discover and effectively provision entire operating systems remotely.

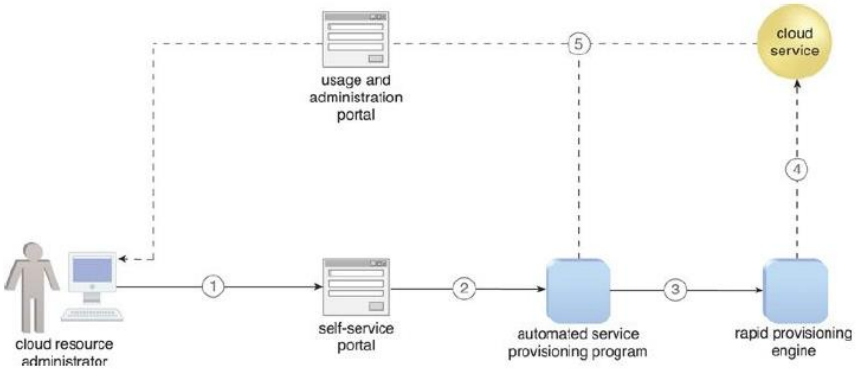
The remote management software that is integrated with the server's ROM becomes available upon server start-up. A Web-based or proprietary userinterface, like the portal provided by the remote administration system, is usually used to connect to the physical server's native remote management interface. IP addresses in IaaS platforms can be forwarded directly to cloud consumers so that they can perform bare-metal operating system installations independently.

The bare-metal provisioning system provides an auto-deployment feature that allows cloud consumers to connect to the deployment software and provision more than one server or operating system at the same time. The central deployment system connects to the servers via their management interfaces, and uses the same protocol to upload and operate as an agent in the physical server's RAM. The bare-metal server then becomes a raw client with a management agent installed, and the deployment software uploads the required setup files to deploy the operating system.



4.2.9 Rapid Provisioning Architecture

The *rapid provisioning architecture* establishes a system that automates the provisioning of a wide range of resources, either individually or as a collective. The underlying technology architecture for rapid resource provisioning can be sophisticated and complex, and relies on a system comprised of an automated provisioning program, rapid provisioning engine, and scripts and templates for on-demand provisioning.



(1) A cloud resource administrator requests a new cloud service through the self-service portal.
(2) The self-service portal passes the request to the automated service provisioning program installed on the virtual server.
(3) which passes the necessary tasks to be performed to the rapid provisioning engine.
(4) The rapid provisioning engine announces when the new cloud service is ready.
(5) The automated service provisioning program finalizes and publishes the cloud service on the usage and administration portal for cloud consumer access.

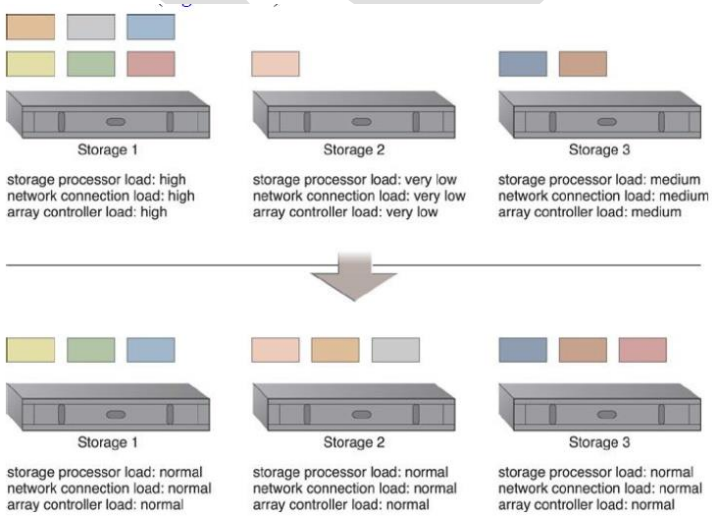
The step-by-step description describes the inner workings of a rapid provisioning engine:

1. A cloud consumer requests a new server through the self-service portal.
2. The sequence manager forwards the request to the deployment engine for the preparation of an operating system.
3. The deployment engine uses the virtual server templates for provisioning if the request is for a virtual server. Otherwise, the deployment engine sends the request to provision a physical server.
4. The pre-defined image for the requested type of operating system is used for the provisioning of the operating system, if available. Otherwise, the regular deployment process is executed to install the operating system.
5. The deployment engine informs the sequence manager when the operating system is ready.

6. The sequence manager updates and sends the logs to the sequence logger for storage.
7. The sequence manager requests that the deployment engine apply the operating system baseline to the provisioned operating system.
8. The deployment engine applies the requested operating system baseline.
9. The deployment engine informs the sequence manager that the operating system baseline has been applied.
10. The sequence manager updates and sends the logs of completed steps to the sequence logger for storage.
11. The sequence manager requests that the deployment engine install the applications.
12. The deployment engine deploys the applications on the provisioned server.
13. The deployment engine informs the sequence manager that the applications have been installed.
14. The sequence manager updates and sends the logs of completed steps to the sequence logger for storage.
15. The sequence manager requests that the deployment engine apply the application's configuration baseline.
16. The deployment engine applies the configuration baseline.
17. The deployment engine informs the sequence manager that the configuration baseline has been applied.
18. The sequence manager updates and sends the logs of completed steps to the sequence logger for storage.

4.2.10 Storage Workload Management Architecture

This architecture enables LUNs to be evenly distributed across available cloud storage devices, while a storage capacity system is established to ensure that runtime workloads are evenly distributed across the LUNs.



Combining cloud storage devices into a group allows LUN data to be distributed between available storage hosts equally. A storage management system is configured and an automated scaling listener is positioned to monitor and equalize runtime workloads among the grouped cloud storage devices.

Glossary:

- **Intelligent Automation Engine:** The intelligent automation engine automates administration tasks by executing scripts that contain workflow logic.
- **LUN:** A logical unit number (LUN) is a logical drive that represents a partition of a physical drive.
- **Storage Service Gateway:** The storage service gateway is a component that acts as the external interface to cloud storage services, and is capable of automatically redirecting cloud consumer requests whenever the location of the requested data has changed.
- **Storage Replication:** Storage replication is a variation of the resource replication mechanisms used to synchronously or asynchronously replicate data from a primary storage device to a secondary storage device. It can be used to replicate partial and entire LUNs.
- **Heartbeats:** Heartbeats are system-level messages exchanged between hypervisors, hypervisors and virtual servers, and hypervisors and VIMs.
- **Live VM migration:** Live VM migration is a system that is capable of relocating virtual servers or virtual server instances at runtime.
- **LUN migration:** LUN migration is a specialized storage program that is used to move LUNs from one storage device to another without interruption, while remaining transparent to cloud consumers.

References:

Cloud Computing Concepts, Technology & Architecture
- Thomas Erl, Zaigham Mahmood, and Ricardo Puttini – Prentice Hall - 2013
Chapter 11: Fundamental Cloud Architectures
Chapter 12: Advanced Cloud Architectures

Mastering Cloud Computing Foundations and Applications Programming
- Rajkumar Buyya, Christian Vecchiola, S. Thamarai Selvi - Elsevier – 2013

Distributed and Cloud Computing, From Parallel Processing to the Internet of Things
- Kai Hwang, Jack Dongarra, Geoffrey Fox – MK Publishers - 2012

Unit 5: Chapter 1
Cloud Delivery Model Considerations

Unit Structure

5.0 OBJECTIVES

- Describe cloud delivery models for IaaS
- Describe cloud delivery models for PaaS
- Describe cloud delivery models for SaaS
- Describe different ways in which cloud delivery models are administered and utilized by cloud consumers
- Working with IaaS Environments
- Working with PaaS Environments
- Working with SaaS Environments

5.1 INTRODUCTION

A cloud delivery model represents a specific combination of IT resources offered by a cloud provider. This terminology is typically associated with cloud computing and frequently used to describe a type of remote environment and the level of control.

5.2 Cloud Delivery Models: The Cloud Provider Perspective

This section explores the architecture and administration of IaaS, PaaS, and SaaS cloud delivery models from the point of view of the cloud provider (Figure 5.1). The integration and management of these cloud-based environments as part of greater environments and how they can relate to different technologies and cloud mechanism combinations are examined.

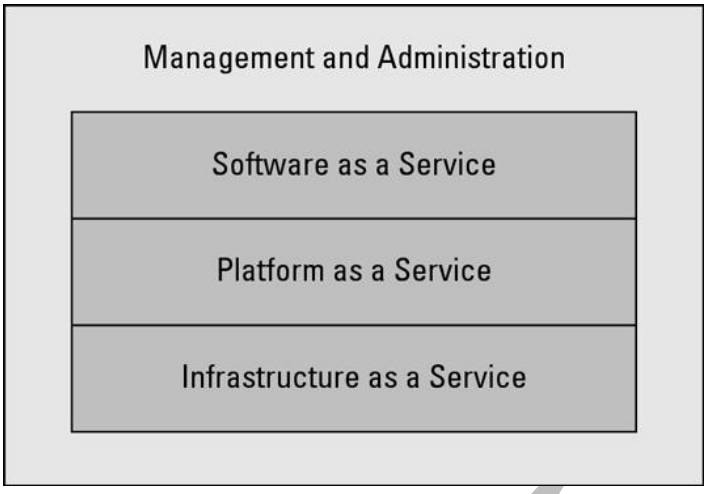


Figure 5.1

5.2.1 Building IaaS Environments

The virtual server and cloud storage device mechanisms represent the two most fundamental IT resources that are delivered as part of a standard rapid provisioning architecture within IaaS environments. They are offered in various standardized configurations that are defined by the following properties:

- Operating System
- Primary Memory Capacity
- Processing Capacity
- Virtualized Storage Capacity

Memory and virtualized storage capacity is usually allocated with increments of 1 GB to simplify the provisioning of underlying physical IT resources. When limiting cloud consumer access to virtualized environments, IaaS offerings are preemptively assembled by cloud providers via virtual server images that capture the pre-defined configurations. Some cloud providers may offer cloud consumers direct administrative access to physical IT resources, in which case the bare-metal provisioning architecture may come into play.

Snapshots can be taken of a virtual server to record its current state, memory, and configuration of a virtualized IaaS environment for backup and replication purposes, in support of horizontal and vertical scaling requirements. For example, a virtual server can use its snapshot to become reinitialized in another hosting environment after its capacity has been increased to allow for vertical scaling. The snapshot can alternatively be used to duplicate a virtual server. The management of custom virtual server images is a vital feature that is provided via the remote administration system mechanism. Most cloud providers also support importing and exporting options for custom-built virtual server images in both proprietary and standard formats.

Data Centers

Cloud providers can offer IaaS-based IT resources from multiple geographically diverse data centers, which provides the following primary benefits:

- Multiple data centers can be linked together for increased resiliency. Each data center is placed in a different location to lower the chances of a single failure forcing all of the data centers to go offline simultaneously.
- Connected through high-speed communications networks with low latency, data centers can perform load balancing, IT resource backup and replication, and increase storage capacity, while improving availability and reliability. Having multiple data centers spread over a greater area further reduces network latency.
- Data centers that are deployed in different countries make access to IT resources more convenient for cloud consumers that are constricted by legal and regulatory requirements.

When an IaaS environment is used to provide cloud consumers with virtualized network environments, each cloud consumer is segregated into a tenant environment that isolates IT resources from the rest of the cloud through the Internet. VLANs and network access control software collaboratively realize the corresponding logical network perimeters.

Scalability and Reliability

Within IaaS environments, cloud providers can automatically provision virtual servers via the dynamic vertical scaling type of the dynamic scalability architecture. This can be performed through the VIM, as long as the host physical servers have sufficient capacity. The VIM can scale virtual servers out using resource replication as part of a resource pool architecture, if a given physical server has insufficient capacity to support vertical scaling. The load balancer mechanism, as part of a workload distribution architecture, can be used to distribute the workload among IT resources in a pool to complete the horizontal scaling process.

Manual scalability requires the cloud consumer to interact with a usage and administration program to explicitly request IT resource scaling. In contrast, automatic scalability requires the automated scaling listener to monitor the workload and reactively scale the resource capacity. This mechanism typically acts as a monitoring agent that tracks IT resource usage in order to notify the resource management system when capacity has been exceeded.

Replicated IT resources can be arranged in high-availability configuration that forms a failover system for implementation via standard VIM features. Alternatively, a high-availability/high-performance resource cluster can be created at the physical or virtual server level, or both simultaneously. The multipath resource access architecture is commonly employed to enhance reliability via the use of redundant access paths, and some cloud providers further offer the provisioning of dedicated IT resources via the resource reservation architecture.

Monitoring

Cloud usage monitors in an IaaS environment can be implemented using the VIM or specialized monitoring tools that directly comprise and/or interface with the virtualization platform. Several common capabilities of the IaaS platform involve monitoring:

- ***Virtual Server Lifecycles*** - Recording and tracking uptime periods and the allocation of IT resources, for pay-per-use monitors and time-based billing purposes.
- ***Data Storage*** - Tracking and assigning the allocation of storage capacity to cloud storage devices on virtual servers, for pay-per-use monitors that record storage usage

for billing purposes.

- ***Network Traffic*** - For pay-per-use monitors that measure inbound and outbound network usage and SLA monitors that track QoS metrics, such as response times and network losses.
- ***Failure Conditions*** - For SLA monitors that track IT resource and QoS metrics to provide warning in times of failure.
- ***Event Triggers*** - For audit monitors that appraise and evaluate the regulatory compliance of select IT resources.

Monitoring architectures within IaaS environments typically involve service agents that communicate directly with backend management systems.

Security

Cloud security mechanisms that are relevant for securing IaaS environments include:

- encryption, hashing, digital signature, and PKI mechanisms for overall protection of data transmission
- IAM and SSO mechanisms for accessing services and interfaces in security systems that rely on user identification, authentication, and authorization capabilities
- cloud-based security groups for isolating virtual environments through hypervisors and network segments via network management software
- hardened virtual server images for internal and externally available virtual server environments
- various cloud usage monitors to track provisioned virtual IT resources to detect abnormal usage patterns.

5.2.2 Equipping PaaS Environments

PaaS environments typically need to be outfitted with a selection of application development and deployment platforms in order to accommodate different programming models, languages, and frameworks. A separate ready-made environment is usually created for each programming stack that contains the necessary software to run applications specifically developed for the platform.

Each platform is accompanied by a matching SDK and IDE, which can be custom-built or enabled by IDE plugins supplied by the cloud provider. IDE toolkits can simulate the cloud runtime locally within the PaaS environment and usually include executable application servers. The security restrictions that are inherent to the runtime are also simulated in the development environment, including checks for unauthorized attempts to access system IT resources.

Cloud providers often offer a resource management system mechanism that is customized for the PaaS platform so that cloud consumers can create and control customized virtual server images with ready-made environments. This mechanism also provides features specific to the PaaS platform, such as managing deployed applications and configuring multitenancy. Cloud providers further rely on a variation of the rapid provisioning architecture known as platform provisioning, which is designed specifically to provision ready-made environments.

Scalability and Reliability

The scalability requirements of cloud services and applications that are deployed within PaaS environments are generally addressed via dynamic scalability and workload distribution architectures that rely on the use of native automated scaling listeners and load balancers. The resource pooling architecture is further utilized to provision IT resources from resource pools made available to multiple cloud consumers.

Cloud providers can evaluate network traffic and server-side connection usage against the instance's workload, when determining how to scale an overloaded application as per parameters and cost limitations provided by the cloud consumer. Alternatively, cloud consumers can configure the application designs to customize the incorporation of available mechanisms themselves.

The reliability of ready-made environments and hosted cloud services and applications can be supported with standard failover system mechanisms (Figure 142), as well as the non-disruptive service relocation architecture, so as to shield cloud consumers from failover conditions. The resource reservation architecture may also be in place to offer exclusive access to PaaS-based IT resources. As

with other IT resources, ready-made environments can also span multiple data centers and geographical regions to further increase availability and resiliency

Monitoring

Specialized cloud usage monitors in PaaS environments are used to monitor the following:

- ***Ready-Made Environment Instances*** - The applications of these instances are recorded by pay-per-use monitors for the calculation of time-based usage fees.
- ***Data Persistence*** - This statistic is provided by pay-per-use monitors that record the number of objects, individual occupied storage sizes, and database transactions per billing period.
- ***Network Usage*** - Inbound and outbound network usage is tracked for pay- per-use monitors and SLA monitors that track network-related QoS metrics.
- ***Failure Conditions*** - SLA monitors that track the QoS metrics of IT resources need to capture failure statistics.
- ***Event Triggers*** - This metric is primarily used by audit monitors that need to respond to certain types of events.

Security

The PaaS environment, by default, does not usually introduce the need for new cloud security mechanisms beyond those that are already provisioned for IaaS environments.

5.2.3 Optimizing SaaS Environments

In SaaS implementations, cloud service architectures are generally based on multitenant environments that enable and regulate concurrent cloud consumer access (Figure 5.2).

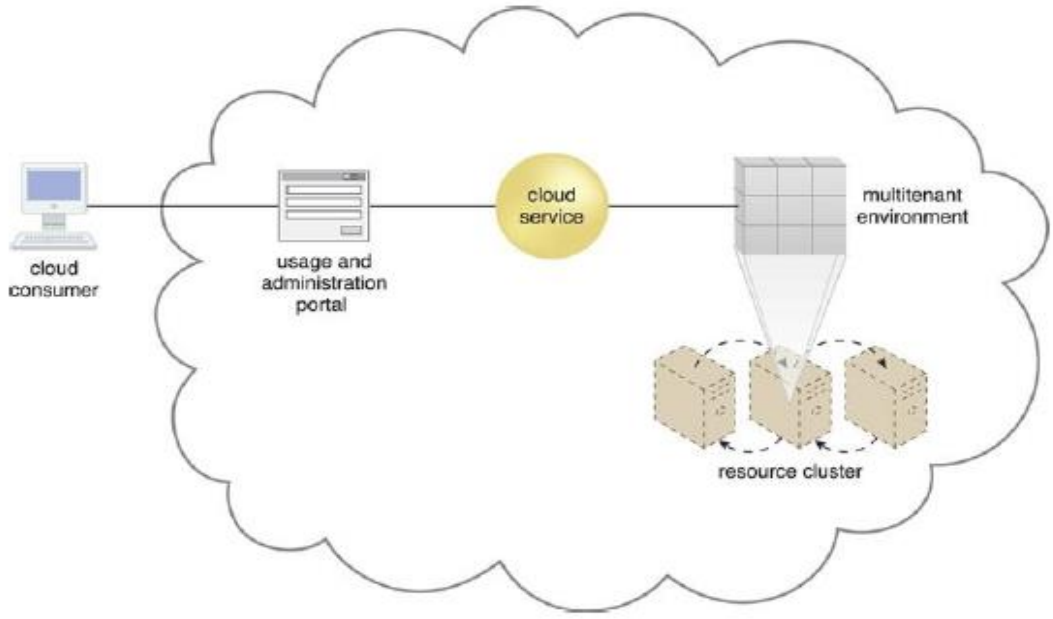


Figure 5.2 The SaaS-based cloud service is hosted by a multitenant environment deployed in a high-performance virtual server cluster. A usage and administration portal is used by the cloud consumer to access and configure the cloud service.

SaaS IT resource segregation does not typically occur at the infrastructure level in SaaS environments, as it does in IaaS and PaaS environments.

SaaS implementations rely heavily on the features provided by the native dynamic scalability and workload distribution architectures, as well as nondisruptive service relocation to ensure that failover conditions do not impact the availability of SaaS-based cloud services.

However, it is vital to acknowledge that, unlike the relatively vanilla designs of IaaS and PaaS products, each SaaS deployment will bring with it unique architectural, functional, and runtime requirements. These requirements are specific to the nature of the business logic the SaaS-based cloud service is programmed with, as well as the distinct usage patterns it is subjected to by its cloud service consumers.

For example, consider the diversity in functionality and usage of the following recognized online SaaS offerings:

- Collaborative authoring and information-sharing (Wikipedia, Blogger)
- Collaborative management (Zimbra, Google Apps)
- Conferencing services for instant messaging, audio/video communications (Skype, Google Talk)
- Enterprise management systems (ERP, CRM, CM)
- File-sharing and content distribution (YouTube, Dropbox)
- Industry-specific software (engineering, bioinformatics)
- Messaging systems (e-mail, voicemail)
- Mobile application marketplaces (Android Play Store, Apple App Store)

- Office productivity software suites (Microsoft Office, Adobe Creative Cloud)
- Search engines (Google, Yahoo)
- Social networking media (Twitter, LinkedIn)

Now consider that many of the previously listed cloud services are offered in one or more of the following implementation mediums:

- Mobile application
- REST service
- Web service

Each of these SaaS implementation mediums provide Web-based APIs for interfacing by cloud consumers. Examples of online SaaS-based cloud services with Web-based APIs include:

- Electronic payment services (PayPal)
- Mapping and routing services (Google Maps)
- Publishing tools (WordPress)

Mobile-enabled SaaS implementations are commonly supported by the multidevice broker mechanism, unless the cloud service is intended exclusively for access by specific mobile devices.

The potentially diverse nature of SaaS functionality, the variation in implementation technology, and the tendency to offer a SaaS-based cloud service redundantly with multiple different implementation mediums makes the design of SaaS environments highly specialized. Though not essential to a SaaS implementation, specialized processing requirements can prompt the need to incorporate architectural models, such as:

- ***Service Load Balancing*** - for workload distribution across redundant SaaS-based cloud service implementations
- ***Dynamic Failure Detection and Recovery*** - to establish a system that can automatically resolve some failure conditions without disruption in service to the SaaS implementation.
- ***Storage Maintenance Window*** - to allow for planned maintenance outages that do not impact SaaS implementation availability
- ***Elastic Resource Capacity/Elastic Network Capacity*** - to establish inherent elasticity within the SaaS-based cloud service architecture that enables it to automatically accommodate a range of runtime scalability requirements
- ***Cloud Balancing*** - to instill broad resiliency within the SaaS implementation, which can be especially important for cloud services subjected to extreme concurrent usage volumes

Specialized cloud usage monitors can be used in SaaS environments to track the following types of metrics:

- ***Tenant Subscription Period*** - This metric is used by pay-per-use monitors to record and track application usage for time-based billing. This type of monitoring usually incorporates application licensing and regular assessments of leasing periods that extend beyond the hourly periods of IaaS and PaaS environments.

- ***Application Usage*** - This metric, based on user or security groups, is used with pay-per-use monitors to record and track application usage for billing purposes.
- ***Tenant Application Functional Module*** - This metric is used by pay-per- use monitors for function-based billing. Cloud services can have different functionality tiers according to whether the cloud consumer is free-tier or a paid subscriber.

5.3 Cloud Delivery Models: The Cloud Consumer Perspective

This section raises various considerations concerning the different ways in which cloud delivery models are administered and utilized by cloud consumers.

5.3.1 Working with IaaS Environments

Virtual servers are accessed at the operating system level through the use of remote terminal applications. Accordingly, the type of client software used directly depends on the type of operating system that is running at the virtual server, of which two common options are:

- ***Remote Desktop (or Remote Desktop Connection) Client*** - for Windows- based environments and presents a Windows GUI desktop
- ***SSH Client*** - for Mac and other Linux-based environments to allow for secure channel connections to text-based shell accounts running on the server OS

Figure 5.3 illustrates a typical usage scenario for virtual servers that are being offered as IaaS services after having been created with management interfaces

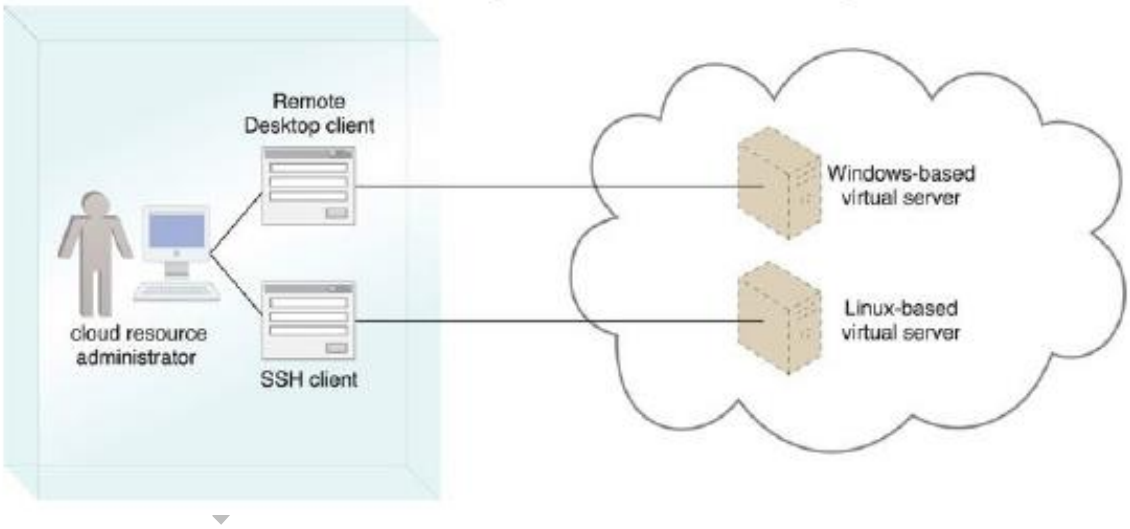


Figure 5.3 A cloud resource administration uses the Windows-based Remote Desktop client to administrator a Windows-based virtual server and the SSH client for the Linux-based virtual server.

A cloud storage device can be attached directly to the virtual servers and accessed through the virtual servers' functional interface for management by the operating system. Alternatively, a cloud storage device can be attached to an IT resource that is being hosted outside of the cloud, such as an on-premise device over a WAN or VPN. In these cases, the following formats for the manipulation and transmission of cloud storage data are commonly used:

• ***Networked File System*** - System-based storage access, whose rendering of files is similar to how folders are organized in operating systems (NFS, CIFS)

- ***Storage Area Network Devices*** - Block-based storage access collates and formats geographically diverse data into cohesive files for optimal network transmission (iSCSI, Fibre Channel)
- ***Web-Based Resources*** - Object-based storage access by which an interface that is not integrated into the operating system logically represents files, which can be accessed through a Web-based interface (Amazon S3)

IT Resource Provisioning Considerations

Cloud consumers have a high degree of control over how and to what extent IT resources are provisioned as part of their IaaS environments.

For example:

- Controlling scalability features (automated scaling, load balancing)
- Controlling the lifecycle of virtual IT resources (shutting down, restarting, powering up of virtual devices)
- Controlling the virtual network environment and network access rules (firewalls, logical network perimeters)
- Establishing and displaying service provisioning agreements (account conditions, usage terms)
- Managing the attachment of cloud storage devices
- Managing the pre-allocation of cloud-based IT resources (resource reservation)
- Managing credentials and passwords for cloud resource administrators
- Managing credentials for cloud-based security groups that access virtualized IT resources through an IAM
- Managing security-related configurations
- Managing customized virtual server image storage (importing, exporting, backup)
- Selecting high-availability options (failover, IT resource clustering)
- Selecting and monitoring SLA metrics
- Selecting basic software configurations (operating system, pre-installed software for new virtual servers) selecting IaaS resource instances from a number of available hardware-related configurations and options (processing capabilities, RAM, storage)
- Selecting the geographical regions in which cloud-based IT resources should be hosted
- Tracking and managing costs

The management interface for these types of provisioning tasks is usually a usage and administration portal, but may also be offered via the use of command line interface (CLI) tools that can simplify the execution of many scripted administrative actions.

Even though standardizing the presentation of administrative features and controls is typically preferred, using different tools and user-interfaces can sometimes be justified. For example, a script can be made to turn virtual servers on and off nightly through a CLI, while adding or removing storage capacity can be more easily carried out using a portal.

5.3.2 Working with PaaS Environments

A typical PaaS IDE can offer a wide range of tools and programming resources, such as software libraries, class libraries, frameworks, APIs, and various runtime capabilities that emulate the intended cloud-based deployment environment. These features allow developers to create, test, and run application code within the cloud or locally (on-premise) while using the IDE to emulate the cloud deployment environment. Compiled or completed applications are then bundled and uploaded to the cloud, and deployed via the ready-made environments. This deployment process can also be controlled through the IDE.

PaaS also allows for applications to use cloud storage devices as independent data storing systems for holding development-specific data (for example in a repository that is available outside of the cloud environment). Both SQL and NoSQL database structures are generally supported.

IT Resource Provisioning Considerations

PaaS environments provide less administrative control than IaaS environments, but still offer a significant range of management features.

For example:

- Establishing and displaying service provisioning agreements, such as account conditions and usage terms
- Selecting software platform and development frameworks for ready-made environments
- Selecting instance types, which are most commonly frontend or backend instances
- Selecting cloud storage devices for use in ready-made environments
- Controlling the lifecycle of PaaS-developed applications (deployment, starting, shutdown, restarting, and release)
- Controlling the versioning of deployed applications and modules
- Configuring availability and reliability-related mechanisms
- Managing credentials for developers and cloud resource administrators using IAM
- Managing general security settings, such as accessible network ports
- Selecting and monitoring PaaS-related SLA metrics
- Managing and monitoring usage and IT resource costs
- Controlling scalability features such as usage quotas, active instance thresholds, and the configuration and deployment of the automated scaling listener and load balancer mechanisms

5.3.3 Working with SaaS Services

Because SaaS-based cloud services are almost always accompanied by refined and generic APIs, they are usually designed to be incorporated as part of larger distributed solutions. A common example of this is Google Maps, which offers a comprehensive API that enables mapping information and images to be incorporated into Web sites and Web-based applications.

Many SaaS offerings are provided free of charge, although these cloud services often come with data collecting sub-programs that harvest usage data for the benefit of the cloud provider. When using any SaaS product that is sponsored by third parties, there is a reasonable chance that it is performing a form of background information gathering. Reading the cloud provider's agreement will usually help shed light on any secondary activity that the cloud service is designed to perform.

Cloud consumers using SaaS products supplied by cloud providers are relieved of the responsibilities of implementing and administering their underlying hosting environments. Customization options are usually available to cloud consumers; however, these options are generally limited to the runtime usage control of the cloud service instances that are generated specifically by and for the cloud consumer.

For example:

- Managing security-related configurations
- Managing select availability and reliability options
- Managing usage costs
- Managing user accounts, profiles, and access authorization
- Selecting and monitoring SLAs
- Setting manual and automated scalability options and limitations.

Unit 5: Chapter 2
Cost Metrics and Pricing Models

Unit Structure

5.2..0 OBJECTIVES

This chapter provides metrics, formulas, and practices to assist cloud consumers in performing accurate financial analysis of cloud adoption plans.

5.2..1 INTRODUCTION

Reducing operating costs and optimizing IT environments are pivotal to understanding and being able to compare the cost models behind provisioning on-premise and cloud-based environments. The pricing structures used by public clouds are typically based on utility-centric pay-per-usage models, enabling organizations to avoid up-front infrastructure investments. These models need to be assessed against the financial implications of on-premise infrastructure investments and associated total cost-of-ownership commitments.

5.2..2 Business Cost Metrics

This section begins by describing the common types of metrics used to evaluate the estimated costs and business value of leasing cloud-based IT resources when compared to the purchase of on-premise IT resources.

Up-Front and On-Going Costs

Up-front costs are associated with the initial investments that organizations need to make in order to fund the IT resources they intend to use. This includes both the costs associated with obtaining the IT resources, as well as expenses required to deploy and administer them.

- Up-front costs for the purchase and deployment of on-premise IT resources tend to be high. Examples of up-front costs for on-premise environments can include hardware, software, and the labor required for deployment.
- Up-front costs for the leasing of cloud-based IT resources tend to be low. Examples of up-front costs for cloud-based environments can include the labor costs required to assess and set up a cloud environment.

On-going costs represent the expenses required by an organization to run and maintain IT resources it uses.

- On-going costs for the operation of on-premise IT resources can vary. Examples include licensing fees, electricity, insurance, and labor.
- On-going costs for the operation of cloud-based IT resources can also vary, but often exceed the on-going costs of on-premise IT resources (especially over a longer period of time). Examples include virtual hardware leasing fees, bandwidth usage fees, licensing fees, and labor.

Additional Costs

To supplement and extend a financial analysis beyond the calculation and comparison of standard up-front and on-going business cost metrics, several other more specialized business cost metrics can be taken into account.
For example:

- ***Cost of Capital*** - The ***cost of capital*** is a value that represents the cost incurred by raising required funds. For example, it will generally be more expensive to raise an initial investment of $150,000 than it will be to raise this amount over a period of three years. The relevancy of this cost depends on how the organization goes about gathering the funds it requires. If the cost of capital for an initial investment is high, then it further helps justify the leasing of cloud-based IT resources.
- ***Sunk Costs*** - An organization will often have existing IT resources that are already paid for and operational. The prior investment that has been made in these on-premise IT resources is referred to as ***sunk costs.*** When comparing up-front costs together with significant sunk costs, it can be more difficult to justify the leasing of cloud-based IT resources as an alternative.
- ***Integration Costs*** - Integration testing is a form of testing required to measure the effort required to make IT resources compatible and interoperable within a foreign environment, such as a new cloud platform. Depending on the cloud deployment model and cloud delivery model being considered by an organization, there may be the need to further allocate funds to carry out integration testing and additional labor related to enable interoperability between cloud service consumers and cloud services. These expenses are referred to as ***integration costs.*** High integration costs can make the option of leasing cloud-based IT resources less appealing.
- ***Locked-in Costs*** - As explained in the ***Risks and Challenges*** section in Chapter 3. cloud environments can impose portability limitations. When performing a metrics analysis over a longer period of time, it may be necessary to take into consideration the possibility of having

to move from one cloud provider to another. Due to the fact that cloud service consumers can become dependent on proprietary characteristics of a cloud environment, there are ***locked-in costs*** associated with this type of move. Locked-in costs can further decrease the long-term business value of leasing cloud-based IT resources.

5.2..3 Cloud Usage Cost Metrics

The following sections describe a set of usage cost metrics for calculating costs associated with cloud-based IT resource usage measurements:

- ***Network Usage*** - inbound and outbound network traffic, as well as intracloud network traffic
- ***Server Usage*** - virtual server allocation (and resource reservation)
- ***Cloud Storage Device*** - storage capacity allocation
- ***Cloud Service*** - subscription duration, number of nominated users, number of transactions (of cloud services and cloud-based applications)

For each usage cost metric a description, measurement unit, and measurement frequency is provided, along with the cloud delivery model most applicable to the metric. Each metric is further supplemented with a brief example.

5.2..3.1 Network Usage

Defined as the amount of data that is transferred over a network connection, network usage is typically calculated using separately measured ***inbound network usage traffic*** and ***outbound network usage traffic*** metrics in relation to cloud services or other IT resources.

Inbound Network Usage Metric

- ***Description*** - inbound network traffic
- ***Measurement*** - £, inbound network traffic in bytes
- ***Frequency*** - continuous and cumulative over a predefined period
- ***Cloud Delivery Model*** - IaaS, PaaS, SaaS
- ***Example*** - up to 1 GB free, $0.001/GB up to 5.2. TB a month

Outbound Network Usage Metric

- ***Description*** - outbound network traffic
- ***Measurement*** - £, outbound network traffic in bytes
- ***Frequency*** - continuous and cumulative over a predefined period
- ***Cloud Delivery Model*** - IaaS, PaaS, SaaS
- ***Example*** - up to 1 GB free a month, $0.01/GB between 1 GB to 5.2. TB per month

Network usage metrics can be applied to WAN traffic between IT resources of one cloud that are located in different geographical regions in order to calculate costs for synchronization, data replication, and related forms of processing. Conversely, LAN usage and other network traffic among IT resources that reside at the same data center are typically not tracked.

Intra-Cloud WAN Usage Metric

- ***Description*** - network traffic between geographically diverse IT resources of the same cloud
- ***Measurement*** - £, intra-cloud WAN traffic in bytes
- ***Frequency*** - continuous and cumulative over a predefined period

- ***Cloud Delivery Model*** - IaaS, PaaS, SaaS
- ***Example*** - up to 500 MB free daily and $0.01/GB thereafter, $0.005/GB after 1 TB per month

Many cloud providers do not charge for inbound traffic in order to encourage cloud consumers to migrate data to the cloud. Some also do not charge for WAN traffic within the same cloud.

Network-related cost metrics are determined by the following properties:

- ***Static IP Address Usage*** - IP address allocation time (if a static IP is required)
- ***Network Load-Balancing*** - the amount of load-balanced network traffic (in bytes)
- ***Virtual Firewall*** - the amount of firewall-processed network traffic (as per allocation time)

5.2..3.2 Server Usage

The allocation of virtual servers is measured using common pay-per-use metrics in IaaS and PaaS environments that are quantified by the number of virtual servers and ready-made environments. This form of server usage measurement is divided into ***on-demand virtual machine instance allocation*** and ***reserved virtual machine instance allocation*** metrics.

The former metric measures pay-per-usage fees on a short-term basis, while the latter metric calculates up-front reservation fees for using virtual servers over extended periods. The up-front reservation fee is usually used in conjunction with the discounted pay-per-usage fees.

On-Demand Virtual Machine Instance Allocation Metric

- ***Description*** - uptime of a virtual server instance
- ***Measurement*** - E, virtual server start date to stop date
- ***Frequency*** - continuous and cumulative over a predefined period
- ***Cloud Delivery Model*** - IaaS, PaaS
- ***Example*** - $0.5.2./hour small instance, $0.20/hour medium instance, $0.50/hour large instance

Reserved Virtual Machine Instance Allocation Metric

- ***Description*** - up-front cost for reserving a virtual server instance
- ***Measurement*** - E, virtual server reservation start date to expiry date
- ***Frequency*** - daily, monthly, yearly
- ***Cloud Delivery Model*** - IaaS, PaaS
- ***Example*** - $55.5.2./small instance, $55.50/medium instance, $245.50/large instance

3-Cloud Storage Device Usage

Cloud storage is generally charged by the amount of space allocated within a predefined period, as measured by the ***on-demand storage allocation*** metric. Similar to IaaS-based cost metrics, on-demand storage allocation fees are usually based on short time increments (such as on an hourly basis). Another common cost metric for cloud storage is ***I/O data transferred,*** which measures the amount of transferred input and output data.

On-Demand Storage Space Allocation Metric

- ***Description*** - duration and size of on-demand storage space allocation in bytes
- ***Measurement*** - E, date of storage release / reallocation to date of storage allocation (resets upon change in storage size)
- ***Frequency*** – continuous
- ***Cloud Delivery Model*** - IaaS, PaaS, SaaS
- ***Example*** - $0.01/GB per hour (typically expressed as GB/month)

I/O Data Transferred Metric

- ***Description*** - amount of transferred I/O data
- ***Measurement*** - E, I/O data in bytes
- ***Frequency*** - continuous
- ***Cloud Delivery Model*** - IaaS, PaaS
- ***Example*** - $0.5.2./TB

5.2..3.4 Cloud Service Usage

Cloud service usage in SaaS environments is typically measured using the following three metrics:

Application Subscription Duration Metric

- ***Description*** - duration of cloud service usage subscription
- ***Measurement*** - E, subscription start date to expiry date
- ***Frequency*** - daily, monthly, yearly
- ***Cloud Delivery Model*** - SaaS
- ***Example*** - $65.50 per month

Number of Nominated Users Metric

- ***Description*** - number of registered users with legitimate access
- ***Measurement*** - number of users
- ***Frequency*** - monthly, yearly
- ***Cloud Delivery Model*** - SaaS
- ***Example*** - $0.50/additional user per month

Number of Transactions Users Metric

- ***Description*** - number of transactions served by the cloud service
- ***Measurement*** - number of transactions (request-response message exchanges)
- ***Frequency*** - continuous
- ***Cloud Delivery Model*** - **PaaS, SaaS**
- **Example** - ***$0.05 per 1,000 transaction***

5.2..4 Cost Management Considerations

Cost management is often centered around the lifecycle phases of cloud services, as follows:

- ***Cloud Service Design and Development*** - During this stage, the vanilla pricing models and cost templates are typically defined by the organization delivering the cloud service.
- ***Cloud Service Deployment*** - Prior to and during the deployment of a cloud service, the

backend architecture for usage measurement and billing- related data collection is determined and implemented, including the positioning of pay-per-use monitor and billing management system mechanisms.

- ***Cloud Service Contracting*** - This phase consists of negotiations between the cloud consumer and cloud provider with the goal of reaching a mutual agreement on rates based on usage cost metrics.
- ***Cloud Service Offering*** - This stage entails the concrete offering of a cloud service's pricing models through cost templates, and any available customization options.
- ***Cloud Service Provisioning*** - Cloud service usage and instance creation thresholds may be imposed by the cloud provider or set by the cloud consumer. Either way, these and other provisioning options can impact usage costs and other fees.
- ***Cloud Service Operation*** - This is the phase during which active usage of the cloud service produces usage cost metric data.
- ***Cloud Service Decommissioning*** - When a cloud service is temporarily or permanently deactivated, statistical cost data may be archived.

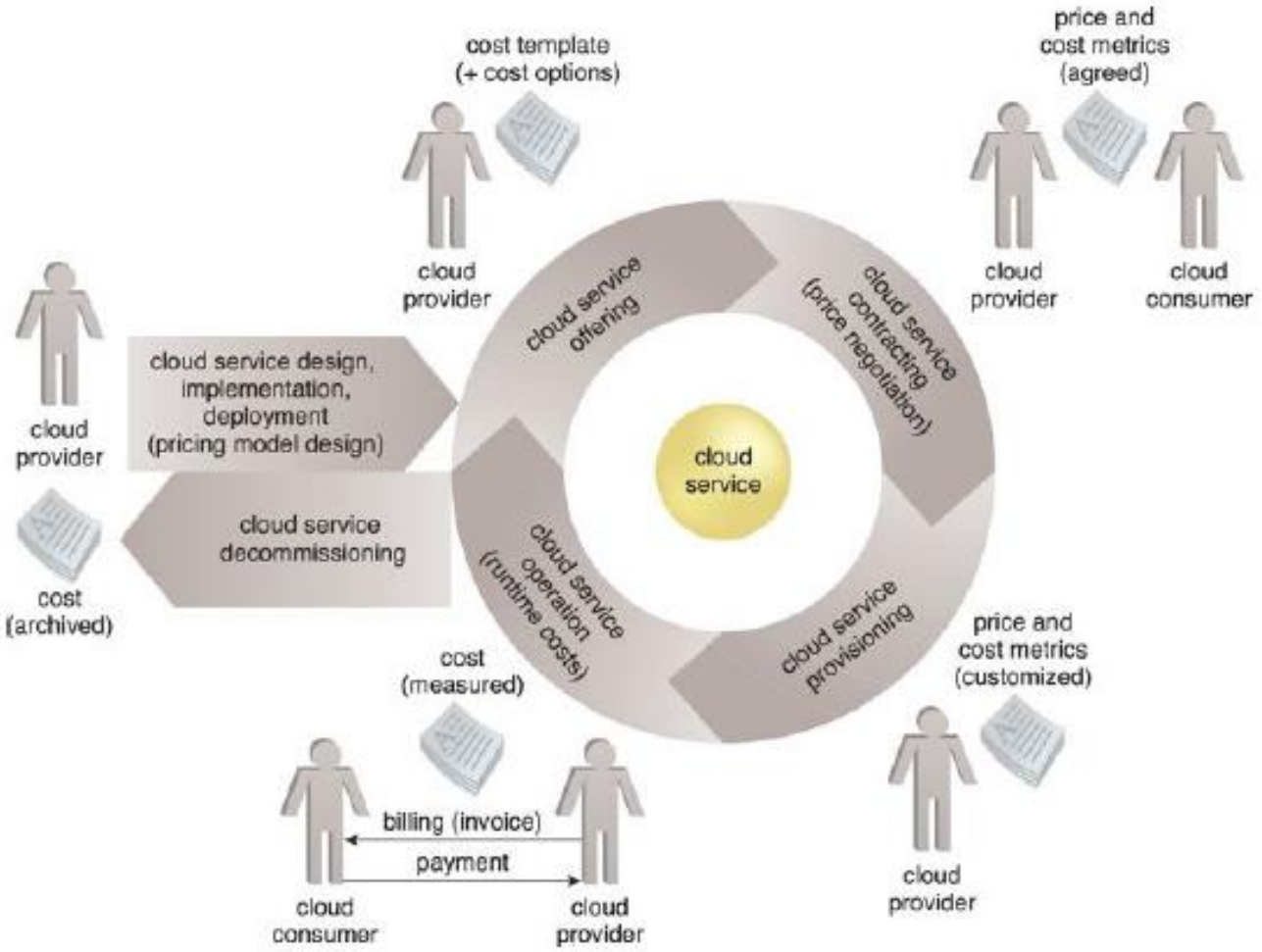


Figure 5.2..1 Common cloud service lifecycle stages as they relate to cost management considerations.

5.2..4.1 Pricing Models

The pricing models used by cloud providers are defined using templates that specify unit costs for fine-grained resource usage according to usage cost metrics. Various factors can influence a pricing model, such as:

- Market competition and regulatory requirements
- Overhead incurred during the design, development, deployment, and operation of cloud services and other IT resources
- Opportunities to reduce expenses via IT resource sharing and data center optimization

Most major cloud providers offer cloud services at relatively stable, competitive prices even though their own expenses can be volatile. A price template or pricing plan contains a set of standardized costs and metrics that specify how cloud service fees are measured and calculated. Price templates define a pricing model's structure by setting various units of measure, usage quotas, discounts, and other codified fees. A pricing model can contain multiple price templates, whose formulation is determined by variables like:

- ***Cost Metrics and Associated Prices*** - These are costs that are dependent on the type of IT resource allocation (such as on-demand versus reserved allocation).
- ***Fixed and Variable Rates Definitions*** - Fixed rates are based on resource allocation and define the usage quotas included in the fixed price, while variable rates are aligned with actual resource usage.
- ***Volume Discounts*** - More IT resources are consumed as the degree of IT resource scaling progressively increases, thereby possibly qualifying a cloud consumer for higher discounts.
- ***Cost and Price Customization Options*** - This variable is associated with payment options and schedules. For example, cloud consumers may be able to choose monthly, semi-annual, or annual payment installments.

5.2..4.2 Additional Considerations

- ***Negotiation*** - Cloud provider pricing is often open to negotiation, especially for customers willing to commit to higher volumes or longer terms. Price negotiations can sometimes be executed online via the cloud provider's Web site by submitting estimated usage volumes along with proposed discounts. There are even tools available for cloud consumers to help generate accurate IT resource usage estimates for this purpose.
- ***Payment Options*** - After completing each measurement period, the cloud provider's billing management system calculates the amount owed by a cloud consumer. There are two common payment options available to cloud consumers: pre-payment and post-payment. With pre-paid billing, cloud consumers are provided with IT resource usage credits that can be applied to future usage bills. With the post-payment method, cloud consumers are billed and invoiced for each IT resource consumption period, which is usually on a monthly basis.
- ***Cost Archiving*** - By tracking historical billing information both cloud providers and cloud consumers can generate insightful reports that help identify usage and financial trends.

5.2..5 Service-level agreements (SLAs)

Service-level agreements (SLAs) are a focal point of negotiations, contract terms, legal obligations, and runtime metrics and measurements. SLAs formalize the guarantees put forth by cloud providers, and correspondingly influence or determine the pricing models and payment terms. SLAs set cloud consumer expectations and are integral to how organizations build business automation around the utilization of cloud-based IT resources.

The guarantees made by a cloud provider to a cloud consumer are often carried forward, in that the same guarantees are made by the cloud consumer organization to its clients, business partners, or whomever will be relying on the services and solutions hosted by the cloud provider. It is therefore crucial for SLAs and related service quality metrics to be understood and aligned support of the cloud consumer's business requirements, while also ensuring that the guarantees can, in fact, be realistically fulfilled consistently and reliably by the cloud provider. The latter consideration is especially relevant for cloud providers that host shared IT resources for high volumes of cloud consumers, each of which will have been issued its own SLA guarantees.

5.2..6 Service Quality Metrics

SLAs issued by cloud providers are human-readable documents that describe quality-of-service (QoS) features, guarantees, and limitations of one or more cloud-based IT resources.

SLAs use service quality metrics to express measurable QoS characteristics.

For example:

- ***Availability*** - up-time, outages, service duration
- ***Reliability*** - minimum time between failures, guaranteed rate of successful responses
- ***Performance*** - capacity, response time, and delivery time guarantees
- ***Scalability*** - capacity fluctuation and responsiveness guarantees
- ***Resiliency*** - mean-time to switchover and recovery

SLA management systems use these metrics to perform periodic measurements that verify compliance with SLA guarantees, in addition to collecting SLA- related data for various types of statistical analyses.

Each service quality metric is ideally defined using the following characteristics:

- ***Quantifiable*** - The unit of measure is clearly set, absolute, and appropriate so that the metric can be based on quantitative measurements.
- ***Repeatable*** - The methods of measuring the metric need to yield identical results when repeated under identical conditions.
- ***Comparable*** - The units of measure used by a metric need to be standardized and comparable. For example, a service quality metric cannot measure smaller quantities of data in bits and larger quantities in bytes.
- ***Easily Obtainable*** - The metric needs to be based on a non-proprietary, common form of measurement that can be easily obtained and understood by cloud consumers.

5.2..6.1 Service Availability Metrics

Availability Rate Metric

The overall availability of an IT resource is usually expressed as a percentage of up-time. For example, an IT resource that is always available will have an uptime of 5.2.0%.

- ***Description*** - percentage of service up-time
- ***Measurement*** - total up-time / total time
- ***Frequency*** - weekly, monthly, yearly
- ***Cloud Delivery Model*** - IaaS, PaaS, SaaS
- ***Example*** - minimum 55.5% up-time

Availability rates are calculated cumulatively, meaning that unavailability periods are combined in order to compute the total downtime (Table 5.2..1)

Availability (%)	Downtime/Week (Seconds)	Downtime/Month (Seconds)	Downtime/Year (Seconds)
99.5	3024	216	158112
99.8	1210	5174	63072
99.9	606	2592	31536
99.95	302	1294	15768
99.99	60.6	259.2	3154
99.999	6.05	25.9	316.6
99.9999	0.605	2.59	31.5

Table 5.2..1 Sample availability rates measured in units of seconds

Outage Duration Metric

This service quality metric is used to define both maximum and average continuous outage service-level targets.

- ***Description*** - duration of a single outage
- ***Measurement*** - date/time of outage end - date/time of outage start
- ***Frequency*** - per event
- ***Cloud Delivery Model*** - IaaS, PaaS, SaaS
- ***Example*** - 1 hour maximum, 15 minutes average

5.2..6.2 Service Reliability Metrics

A characteristic closely related to availability; reliability is the probability that an IT resource can perform its intended function under pre-defined conditions without experiencing failure. Reliability focuses on how often the service performs as expected, which requires the service to remain in an operational and available state. Certain reliability metrics only consider runtime

errors and exception conditions as failures, which are commonly measured only when the IT resource is available.

Mean-Time Between Failures (MTBF) Metric

- ***Description*** - expected time between consecutive service failures
- ***Measurement*** - £, normal operational period duration / number of failures
- ***Frequency*** - monthly, yearly
- ***Cloud Delivery Model*** - IaaS, PaaS
- ***Example*** - 50 day average

Reliability Rate Metric

Overall reliability is more complicated to measure and is usually defined by a reliability rate that represents the percentage of successful service outcomes.

This metric measures the effects of non-fatal errors and failures that occur during up-time periods. For example, an IT resource's reliability is 5.2.0% if it has performed as expected every time it is invoked, but only 80% if it fails to perform every fifth time.

- ***Description*** - percentage of successful service outcomes under pre-defined conditions
- ***Measurement*** - total number of successful responses / total number of requests
- ***Frequency*** - weekly, monthly, yearly
- ***Cloud Delivery Model*** - SaaS
- ***Example*** - minimum 55.5%

5.2..6.3 Service Performance Metrics

Service performance refers to the ability on an IT resource to carry out its functions within expected parameters. This quality is measured using service capacity metrics, each of which focuses on a related measurable characteristic of IT resource capacity. A set of common performance capacity metrics is provided in this section. Note that different metrics may apply, depending on the type of IT resource being measured.

Network Capacity Metric

- ***Description*** - measurable characteristics of network capacity
- ***Measurement*** - bandwidth / throughput in bits per second
- ***Frequency*** - continuous
- ***Cloud Delivery Model*** - IaaS, PaaS, SaaS
- ***Example*** - 5.2. MB per second

Storage Device Capacity Metric

- ***Description*** - measurable characteristics of storage device capacity
- ***Measurement*** - storage size in GB
- ***Frequency*** - continuous
- ***Cloud Delivery Model*** - IaaS, PaaS, SaaS
- ***Example*** - 80 GB of storage

Server Capacity Metric

- ***Description*** - measurable characteristics of server capacity
- ***Measurement*** - number of CPUs, CPU frequency in GHz, RAM size in GB, storage size in GB
- ***Frequency*** - continuous
- ***Cloud Delivery Model*** - IaaS, PaaS
- ***Example*** - 1 core at 1.7 GHz, 16 GB of RAM, 80 GB of storage

Web Application Capacity Metric

- ***Description*** - measurable characteristics of Web application capacity
- ***Measurement*** - rate of requests per minute
- ***Frequency*** - continuous
- ***Cloud Delivery Model*** - SaaS
- ***Example*** - maximum 5.2.0,000 requests per minute

Instance Starting Time Metric

- ***Description*** - length of time required to initialize a new instance
- ***Measurement*** - date/time of instance up - date/time of start request
- ***Frequency*** - per event
- ***Cloud Delivery Model*** - IaaS, PaaS
- ***Example*** - 5 minute maximum, 3 minute average

Response Time Metric

- ***Description*** - time required to perform synchronous operation
- ***Measurement*** - (date/time of request - date/time of response) / total number of requests
- ***Frequency*** - daily, weekly, monthly
- ***Cloud Delivery Model*** - SaaS
- ***Example*** - 5 millisecond average

Completion Time Metric

- ***Description*** - time required to complete an asynchronous task
- ***Measurement*** - (date of request - date of response) / total number of requests
- ***Frequency*** - daily, weekly, monthly
- ***Cloud Delivery Model*** - **PaaS, SaaS**
- **Example** - ***1 second average***

5.2..6.4 Service Scalability Metrics

Service scalability metrics are related to IT resource elasticity capacity, which is related to the maximum capacity that an IT resource can achieve, as well as measurements of its ability to adapt to workload fluctuations. For example, a server can be scaled up to a maximum of 128 CPU cores and 512 GB of RAM, or scaled out to a maximum of 16 load-balanced replicated instances.
The following metrics help determine whether dynamic service demands will be met proactively or reactively, as well as the impacts of manual or automated IT resource allocation processes.

Storage Scalability (Horizontal) Metric

- ***Description*** - permissible storage device capacity changes in response to increased workloads
- ***Measurement*** - storage size in GB
- ***Frequency*** - continuous
- ***Cloud Delivery Model*** - IaaS, PaaS, SaaS
- ***Example*** - 1,000 GB maximum (automated scaling)

Server Scalability (Horizontal) Metric

- ***Description*** - permissible server capacity changes in response to increased workloads
- ***Measurement*** - number of virtual servers in resource pool
- ***Frequency*** - continuous
- ***Cloud Delivery Model*** - IaaS, PaaS
- ***Example*** - 1 virtual server minimum, 5.2. virtual server maximum (automated scaling)

Server Scalability (Vertical) Metric

- ***Description*** - permissible server capacity fluctuations in response to workload fluctuations
- ***Measurement*** - number of CPUs, RAM size in GB
- ***Frequency*** - continuous
- ***Cloud Delivery Model*** - IaaS, PaaS
- ***Example*** - 512 core maximum, 512 GB of RAM

5.2..6.5 Service Resiliency Metrics

The ability of an IT resource to recover from operational disturbances is often measured using service resiliency metrics. When resiliency is described within or in relation to SLA resiliency guarantees, it is often based on redundant implementations and resource replication over different physical locations, as well as various disaster recovery systems.

The type of cloud delivery model determines how resiliency is implemented and measured. For example, the physical locations of replicated virtual servers that are implementing resilient cloud services can be explicitly expressed in the SLAs for IaaS environments, while being implicitly expressed for the corresponding PaaS and SaaS environments.

Resiliency metrics can be applied in three different phases to address the challenges and events that can threaten the regular level of a service:

- ***Design Phase*** - Metrics that measure how prepared systems and services are to cope with challenges.
- ***Operational Phase*** - Metrics that measure the difference in service levels before, during, and after a downtime event or service outage, which are further qualified by availability, reliability, performance, and scalability metrics.
- ***Recovery Phase*** - Metrics that measure the rate at which an IT resource recovers from downtime, such as the meantime for a system to log an outage and switchover to a new virtual server.

Two common metrics related to measuring resiliency are as follows:

Mean-Time to Switchover (MTSO) Metric

- ***Description*** - the time expected to complete a switchover from a severe failure to a replicated instance in a different geographical area
- ***Measurement*** - (date/time of switchover completion - date/time of failure) / total number of failures
- ***Frequency*** - monthly, yearly
- ***Cloud Delivery Model*** - IaaS, PaaS, SaaS
- ***Example*** - 5.2. minutes average

Mean-Time System Recovery (MTSR) Metric

- ***Description*** - time expected for a resilient system to perform a complete recovery from a severe failure
- ***Measurement*** - (date/time of recovery - date/time of failure) / total number of failures
- ***Frequency*** - monthly, yearly
- ***Cloud Delivery Model*** - IaaS, PaaS, SaaS
- ***Example*** - 120 minutes average

5.2..7 SLA Guidelines

This section provides a number of best practices and recommendations for working with SLAs, the majority of which are applicable to cloud consumers:

- ***Mapping Business Cases to SLAs*** - It can be helpful to identify the necessary QoS requirements for a given automation solution and to then concretely link them to the guarantees expressed in the SLAs for IT resources responsible for carrying out the automation. This can avoid situations where SLAs are inadvertently misaligned or perhaps unreasonably deviate in their guarantees, subsequent to IT resource usage.
- ***Working with Cloud and On-Premise SLAs*** - Due to the vast infrastructure available to support IT resources in public clouds, the QoS guarantees issued in SLAs for cloud-based IT resources are generally superior to those provided for on-premise IT resources. This variance needs to be understood, especially when building hybrid distributed solutions that utilize both on on-premise and cloud-based services or when incorporating cross-environment technology architectures, such as cloud bursting.
- ***Understanding the Scope of an SLA*** - Cloud environments are comprised of many supporting architectural and infrastructure layers upon which IT resources reside and are integrated. It is important to acknowledge the extent to which a given IT resource guarantee applies. For example, an SLA may be limited to the IT resource implementation but not its underlying hosting environment.
- ***Understanding the Scope of SLA Monitoring*** - SLAs need to specify where monitoring is performed and where measurements are calculated, primarily in relation to the cloud's firewall. For example, monitoring within the cloud firewall is not always advantageous or relevant to the cloud consumer's required QoS guarantees. Even the most efficient firewalls have a measurable degree of influence on performance and can further present a

point of failure.

- ***Documenting Guarantees at Appropriate Granularity*** - SLA templates used by cloud providers sometimes define guarantees in broad terms. If a cloud consumer has specific requirements, the corresponding level of detail should be used to describe the guarantees. For example, if data replication needs to take place across particular geographic locations, then these need to be specified directly within the SLA.
- ***Defining Penalties for Non-Compliance*** - If a cloud provider is unable to follow through on the QoS guarantees promised within the SLAs, recourse can be formally documented in terms of compensation, penalties, reimbursements, or otherwise.
- ***Incorporating Non-Measurable Requirements*** - Some guarantees cannot be easily measured using service quality metrics, but are relevant to QoS nonetheless, and should therefore still be documented within the SLA. For example, a cloud consumer may have specific security and privacy requirements for data hosted by the cloud provider that can be addressed by assurances in the SLA for the cloud storage device being leased.
- ***Disclosure of Compliance Verification and Management*** - Cloud providers are often responsible for monitoring IT resources to ensure compliance with their own SLAs. In this case, the SLAs themselves should state what tools and practices are being used to carry out the compliance checking process, in addition to any legal-related auditing that may be occurring.
- ***Inclusion of Specific Metric Formulas*** - Some cloud providers do not mention common SLA metrics or the metrics-related calculations in their SLAs, instead focusing on service-level descriptions that highlight the use of best practices and customer support. Metrics being used to measure SLAs should be part of the SLA document, including the formulas and calculations that the metrics are based upon.
- ***Considering Independent SLA Monitoring*** - Although cloud providers will often have sophisticated SLA management systems and SLA monitors, it may be in the best interest of a cloud consumer to hire a third-party organization to perform independent monitoring as well, especially if there are suspicions that SLA guarantees are not always being met by the cloud provider (despite the results shown on periodically issued monitoring reports).
- ***Archiving SLA Data*** - The SLA-related statistics collected by SLA monitors are commonly stored and archived by the cloud provider for future reporting purposes. If a cloud provider intends to keep SLA data specific to a cloud consumer even after the cloud consumer no longer continues its business relationship with the cloud provider, then this should be disclosed. The cloud consumer may have data privacy requirements that disallow the unauthorized storage of this type of information. Similarly, during and after a cloud consumer's engagement with a cloud provider, it may want to keep a copy of historical SLA-related data as well. It may be especially useful for comparing cloud providers in the future.
- ***Disclosing Cross-Cloud Dependencies*** - Cloud providers may be leasing IT resources from other cloud providers, which results in a loss of control over the guarantees they are able to make to cloud consumers. Although a cloud provider will rely on the SLA assurances made to it by other cloud providers, the cloud consumer may want disclosure of the fact that

the IT resources it is leasing may have dependencies beyond the environment of the cloud provider organization.