

tances. This is made possible thanks to infrastructure management platforms offering on demand compute, storage, and, on top of these, more advanced services. There are several different options for building enterprise Cloud computing applications or for using Cloud computing technologies to integrate and extend existing industrial applications. An overview of a few prominent Cloud computing platforms and a brief description of the type of service offered by them are shown in Table 9.1. A Cloud computing system can be developed by using either a single technology and vendor or a combination of them.

This chapter presents some of the representative Cloud computing solutions offered as Infrastructure-as-a-Service (IaaS) and Platform-as-a-Service (PaaS) services in market. It provides some insights and practical issues around the architecture of the major Cloud computing technologies and their service offerings.

## 9.1 → AMAZON WEB SERVICES

Amazon Web Services (AWS) is a platform allowing the development of flexible applications by providing solutions for elastic infrastructure scalability, messaging, and file and data storage. The platform is accessible through SOAP or RESTful Web service interfaces and provides a Web based console where users can administrate and monitor the resources required as well as their expenses computed on a pay as you go basis.

**Table 9.1.** Some Examples of Cloud Computing Offerings.

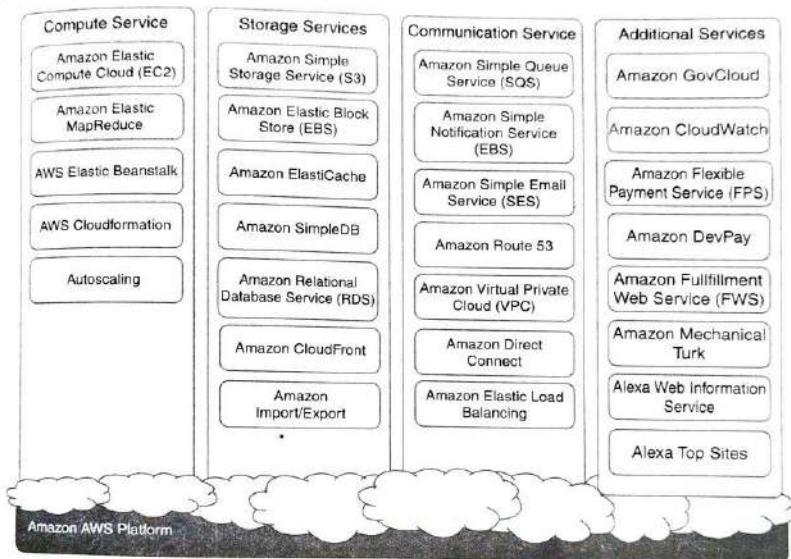
Vendor / Product	Service Type	Description
Amazon Web Services	IaaS, PaaS, SaaS	Amazon Web Services (AWS) is a collection of Web services providing developers with compute, storage, and more advanced services. AWS is mostly popular for IaaS services and primarily for its elastic compute service EC2.

(Continued)

Cloud Platforms in Industry

9.3

a specific image. Images come with already preinstalled operating system and a software stack, and instances can be configured for what regards memory, number of processors, and storage. Users are provided with credentials to remotely access the instance and further configure or install software, if needed.



**Fig. 9.1.** Amazon Web Services Ecosystem.

### 1. Amazon Machine Image (AMI)

AMIs are templates from which it is possible to create a virtual machine. They are stored into the Amazon S3 and identified by a unique identifier in the form of *ami-xxxxxx* and a manifest XML file. An AMI contains a physical file system layout with a predefined operating system installed. These two are specified by the *Amazon Ramdisk Image* (*ARI*, id: *ari-yyyyyy*) and the *Amazon Kernel Image* (*AKI*, id: *aki-zzzzzz*), which are part of the configuration of the template. AMIs are either created from scratch or "bundled" from existing EC2 instances running. A common practice is to prepare new AMIs to create an instance from a pre-existing AMI, log into it once it is booted and running, and install all the software needed and, by using the tools provided by Amazon, the instance is converted into a new image. Once an AMI is created, it is stored into an S3 bucket and the user can decide whether to make it available to other users or keep it for personal use. Finally, it is also possible to associate a product code with a given AMI, thus allowing the owner of the AMI to get revenue every time this AMI is used to create EC2 instances.

### 2. EC2 Instance

EC2 instances represent virtual machines. They are created by using AMI as templates, which are specialized by selecting the number of cores, their computing power, and the installed memory. The



ChatGPT ▾



Google App Engine (GAE) is a Platform as a Service (PaaS) offering from Google Cloud that allows developers to build and deploy applications without having to manage the underlying infrastructure.

### Key Features:

- Automatic scaling: GAE automatically adjusts the number of instances based on traffic.
- Supports multiple languages: Such as Python, Java, Node.js, Go, PHP, Ruby, and .NET.
- Fully managed: Google handles server provisioning, load balancing, patching, and scaling.
- Integrated services: Easy integration with other Google Cloud services like Cloud Datastore, Cloud SQL, and Cloud Storage.

Ask anything





ChatGPT ▾



- Integrated services: Easy integration with other Google Cloud services like Cloud Datastore, Cloud SQL, and Cloud Storage.
- Flexible environments:
  - Standard Environment: Uses sandboxed runtime environments and is optimized for quick scaling.
  - Flexible Environment: Runs on custom Docker containers and supports more control over the environment.

### Use Cases:

- Web and mobile app backends
- APIs and microservices
- Event-driven applications



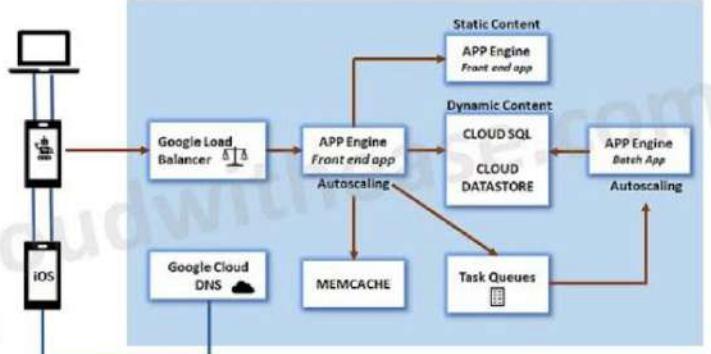
Would you like a simple example of how an app runs

Ask anything





## Google App Engine Architecture



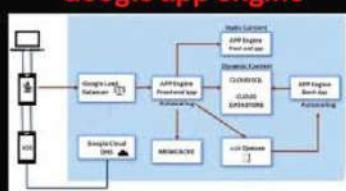
## Google App Engine vs Google Compute Engine ...

[Visit >](#)

Images may be subject to copyright. Learn More

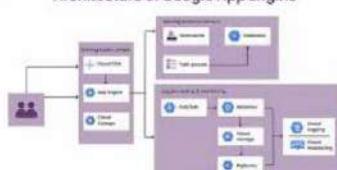
[Share](#)[Save](#)

### Google app engine

[Medium](#)

Exploring the Architect...

### Architecture of Google App Engine

[The Knowledge Academy](#)

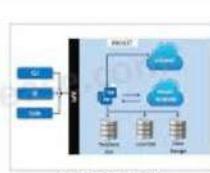
Google app engine in ...

### Google Compute Engine Architecture

[Cloudwithease](#)

Google App Engine vs ...

### Google App Engine vs Google Compute Engine

[Cloudwithease](#)

Google App Engine vs ...



with distributed, Scalable Cloud architectures. ②

23/4/25

### Virtualization

Virtualization is a large umbrella of technologies & concepts that are meant to provide an abstract environment - whether it's virtual hardware / operating system — to run applications. It is one of the fundamental components of Cloud computing, specially in case of infrastructure based services.

Virtualization technologies have gained a renewed interest by the community.

### Reasons

↗ Increase Performance & Computing capacity

NOW a days Super Computers can provide immense computing powers that can accommodate the execution of ~~tools~~ tool's / tool's machines.

↗ Underutilize of hardware / Software resources

H/W & S/W underutilization is occurring due to -

↗ increase Performance / Computing capacity

2) Effect of limited / sporadic use of resources.

(3) Computers today are so powerful that in most of the cases only a fraction of their capacity is used by an application (system.) //  
ii) Lack of space.

The continuous need for additional capacity like storage (Computing powers makes data centers grow quickly. Companies like google / microsoft expand their infrastructure by building data centers that are able to host thousands of nodes.) //

iv) Greening initiatives - It aims to reduce energy consumption & env. impact by running multiple virtual machines on fewer physical servers. [This improves resource efficiency, lowers power & cooling needs & reduces e-waste, supporting sustainable & eco-friendly IT operations.] //

v) Rise of administrative cost -

Virtualization can raise admin costs because it needs skilled staff & special tools. [Managing many virtual systems can be more complex.] //

character

i) Increased security backups

ii) Max sharing resources

• simulating virtual environments

virtual resources

virtualization

Physical resources

• Energy

with virtualization

## Characteristics of virtual environment

i) Increase security - virtual env's increase security by keeping systems separate, making backups easy & controlling access better.

## ii) Manage execution - sharing

sharing of resources, aggregation of resources, emulation, isolation.

- simulate - Executing the plan from real to virtual env.

virtual resources	multiple resources	Sharing	Aggregation	Emulation	Isolation
virtual resources	□□□□	□	□	□	□
Physical resources	□	□□□□□	□	□	□

- Emulation - Guests are ~~exec~~ executed

within an env. that is controlled by the virtualization layer which is a program that is allowed for controlling & ~~tun~~ tuning the env. that is exposed to guest.

which

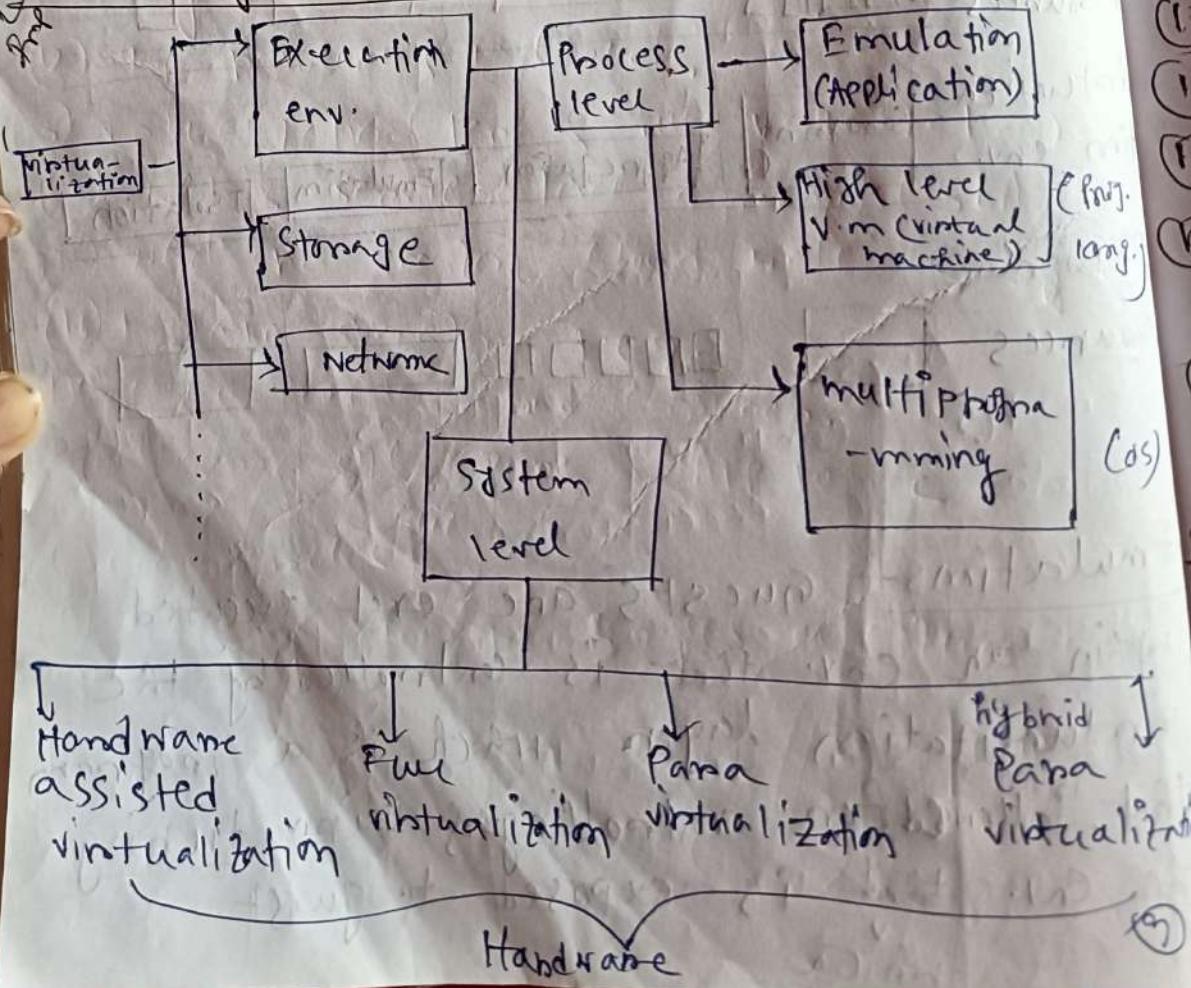
For instance, a complete diff. env. with respect to the host can be emulated for allowing the execution of guest requirement requiring specific characteristics that are not present in the physical machine.] ||

### • Isolation

Virtualization allows guest whether their applications/ other entities with a complete separate env. in which they are executed.

Secondly it provides separation b/w the host & the guest.

### • Taxonomy of virtualization technology





## Benefits of Virtualization

- Encapsulation - VMs can be described in a file
  - Possible to 'snapshot'
  - Easy to move
- Enables running multiple operating systems
- Consolidation & use of unused computation power
- Resource management
- High availability & disaster recovery
- Create "Base Environment"
- Safe testing of new software
- Easy Management



6

vmware®

## Virtualization and Cloud Computing - ppt video...

[Visit >](#)

Images may be subject to copyright. Learn More

[Share](#)[Save](#)

### Benefits of Virtualization

- Reduced capital and operating costs
- Minimized or eliminated downtime
- Increase efficiency and productivity
- Faster Backup
- Seamless migration of resources



What is Virtualization &...

### Advantages of Virtualization

- Minimize hardware costs (Capital) Multiple virtual servers on one physical hardware
- Easily move VMs to other data centers
  - Promote cluster roaming, Hardware maintenance,
  - Move the use (active usage) or reduce the usage (cheap power)
- Consolidate idle workloads, Usage is bursty and asynchronous, Increase device utilization
- Conserves power Free up unused physical resources
- Easier administration (Lower OpEx) Simplified provisioning - administration of hardware and software
- Scalability and Flexibility: Multiple operating systems

- Enhanced utilization of resources
- Better availability for company applications
- Lower maintenance costs
- Lesser personnel needed for the IT department



### Virtualization Techniques...

#### OS-Level Virtualization

- Not full machine virtualization
- Multiple instances of the same operating system
  - Each has its own environment



Home



Search



Saved



Notifications

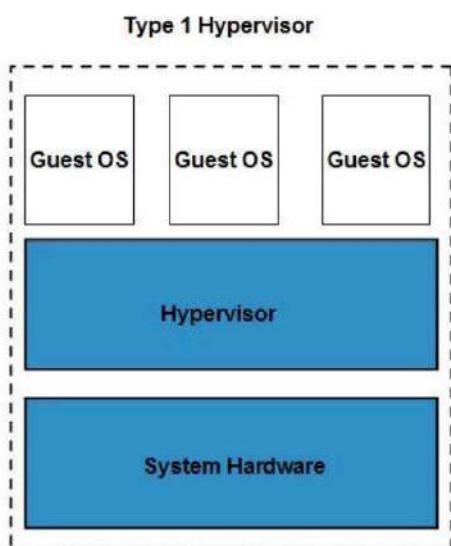
Creating a virtual machine over existing operating system and hardware is referred as Hardware virtualization. Virtual Machines provide an environment that is logically separated from the underlying hardware.

The machine on which the virtual machine is created is known as **host machine** and **virtual machine** is referred as a **guest machine**. This virtual machine is managed by a software or firmware, which is known as **hypervisor**.

## HYPERVISOR

**Hypervisor** is a firmware or low-level program that acts as a Virtual Machine Manager. There are two types of hypervisor:

Type 1 hypervisor runs on bare system. LynxSecure, RTS Hypervisor, Oracle VM, Sun xVM Server, VirtualLogic VLX are examples of Type 1 hypervisor. The following diagram shows the Type 1 hypervisor.

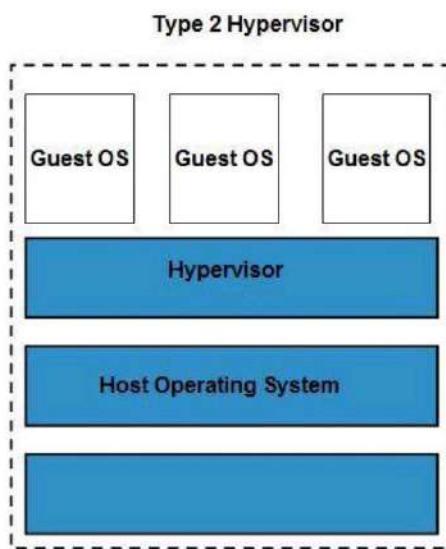


The type1 hypervisor does not have any host operating system because they are installed on a bare system. Type 2 hypervisor is a software interface that emulates the devices with which a system normally interacts. Containers, KVM, Microsoft Hyper V, VMWare Fusion, Virtual Server 2005 R2, Windows Virtual PC and VMWare workstation 6.0 are examples of Type 2 hypervisor. The following diagram shows the Type 2 hypervisor.

---

The type1 hypervisor does not have any host operating system because they are installed on a bare system. Type 2 hypervisor is a software interface that emulates the devices with which a system normally interacts. Containers, KVM, Microsoft Hyper V, VMWare Fusion, Virtual Server 2005 R2, Windows Virtual PC and VMWare workstation 6.0 are examples of Type 2 hypervisor. The following diagram shows the Type 2 hypervisor.

59 / 78



## Types of Hardware Virtualization

Here are the three types of hardware virtualization:

1. Full Virtualization
2. Emulation Virtualization
3. Paravirtualization

### FULL VIRTUALIZATION

In **Full Virtualization**, the underlying hardware is completely simulated. Guest software does not require any modification to run.



ChatGPT ▾



hypervisor clustering  
architecture in cloud  
computing

Hypervisor Clustering Architecture in Cloud

Computing refers to the use of multiple hypervisors (virtual machine monitors) working together in a coordinated cluster to provide high availability, scalability, and resource management for virtualized environments. Here's a breakdown of the architecture and its components:

## 1. Key Concepts



- Hypervisor: Software that creates and runs

Ask anything





## 2. Components of Hypervisor Clustering Architecture

### 1. Clustered Hypervisors:

- Multiple physical hosts running hypervisors.
- Connected via a high-speed network.

### 2. Centralized Management Server:

- Manages and monitors the cluster (e.g., VMware vCenter, Microsoft SCVMM).
- Handles VM migration, load balancing, and policy enforcement.

### 3. Shared Storage:

- Network-attached or SAN storage accessible by all hypervisors.
- Enables VM migration and data redundancy.

Ask anything





ChatGPT ▾



redundancy.

#### 4. VMs and Workloads:

- Deployed across the hypervisor cluster.
- Can move between hosts using technologies like Live Migration or vMotion.

#### 5. Heartbeat and Failover Mechanism:

- Ensures health monitoring of hypervisors.
- Automatically fails over VMs if a host goes down.

### 3. Benefits

- High Availability: If one hypervisor fails, VMs can automatically restart on another.

Ask anything





ChatGPT ▾



### 3. Benefits

- High Availability: If one hypervisor fails, VMs can automatically restart on another.
- Scalability: Easy to add more hosts to the cluster.
- Load Balancing: Even distribution of workloads across hosts.
- Maintenance Without Downtime: Live migration allows updates and maintenance with minimal service interruption.

### 4. Example Use Case



VMware vSphere HA Cluster

Ask anything





ChatGPT ▾



allows updates and maintenance with minimal service interruption.

#### 4. Example Use Case

VMware vSphere HA Cluster:

- Hosts running ESXi are clustered and managed by vCenter.
- Shared storage allows for vMotion and DRS (Distributed Resource Scheduler).
- If one ESXi host fails, VMs are automatically restarted on healthy hosts.



Would you like a diagram to visualize the

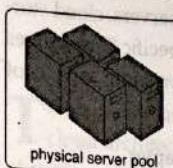
Ask anything



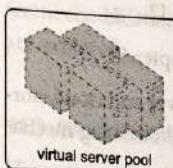
## 11.2 Resource Pooling Architecture

A *resource pooling architecture* is based on the use of one or more resource pools, in which identical IT resources are grouped and maintained by a system that automatically ensures that they remain synchronized.

Provided here are common examples of resource pools:



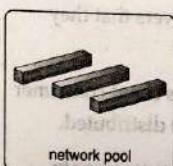
Physical server pools are composed of networked servers that have been installed with operating systems and other necessary programs and/or applications and are ready for immediate use.



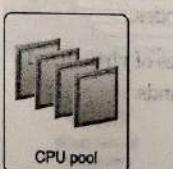
Virtual server pools are usually configured using one of several available templates chosen by the cloud consumer during provisioning. For example, a cloud consumer can set up a pool of mid-tier Windows servers with 4 GB of RAM or a pool of low-tier Ubuntu servers with 2 GB of RAM.



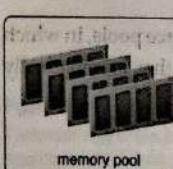
Storage pools, or cloud storage device pools, consist of file-based or block-based storage structures that contain empty and/or filled cloud storage devices.



Network pools (or interconnect pools) are composed 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, or link aggregation.

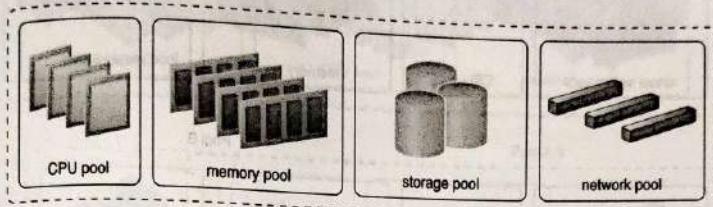


CPU pools are ready to be allocated to virtual servers, and are typically broken down into individual processing cores.



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 IT resource and individual pools can be grouped into a larger pool, in which case each individual pool becomes a sub-pool (Figure 11.2).



**Figure 11.2**

A sample resource pool that is comprised of four sub-pools of CPUs, memory, cloud storage devices, and virtual network devices.

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 (Figure 11.3).

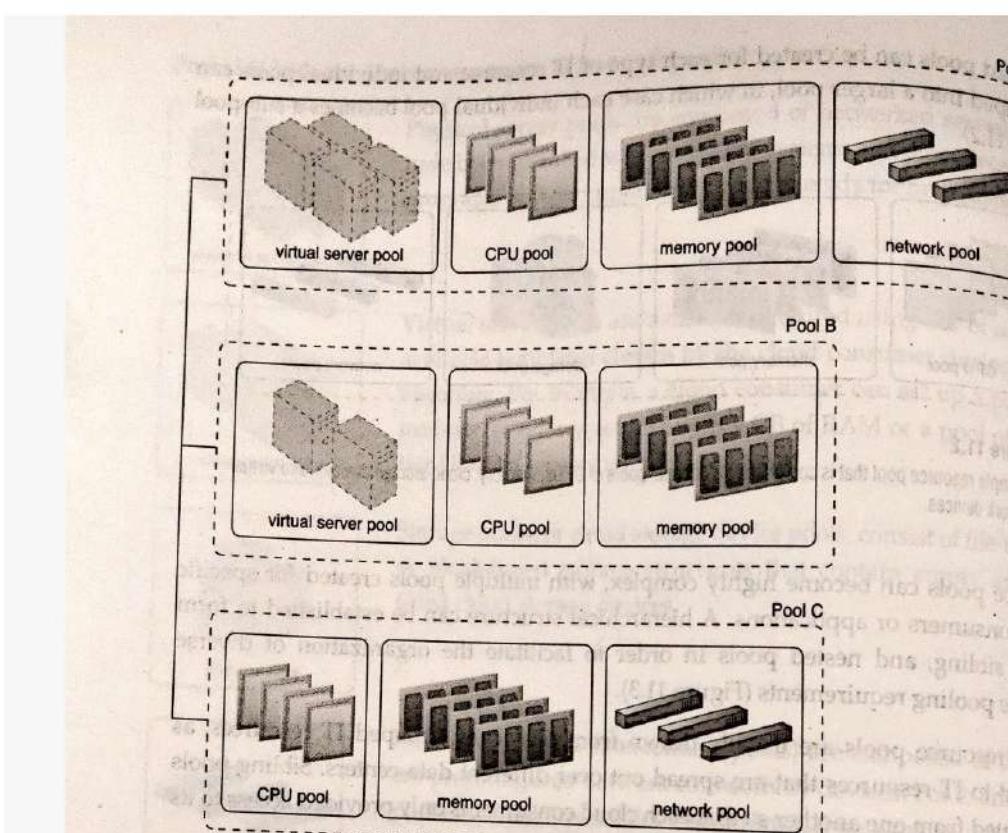
Sibling resource pools are usually drawn from physically grouped IT resources, as opposed to IT resources that are spread out over different data centers. Sibling pools are isolated from one another so that each cloud consumer is only provided access to its respective pool.

In the nested pool model, larger pools are divided into smaller pools that individually group the same type of IT resources together (Figure 11.4). Nested pools can be used to assign resource pools to different departments or groups in the same cloud consumer organization.

After resources pools have been defined, multiple instances of IT resources from each pool can be created to provide an in-memory pool of “live” IT resources.

In addition to cloud storage devices and virtual servers, which are commonly pooled mechanisms, the following mechanisms can also be part of this cloud architecture:

- **Audit Monitor** – This mechanism monitors resource pool usage to ensure compliance with privacy and regulation requirements, especially when pools contain cloud storage devices or data loaded into memory.

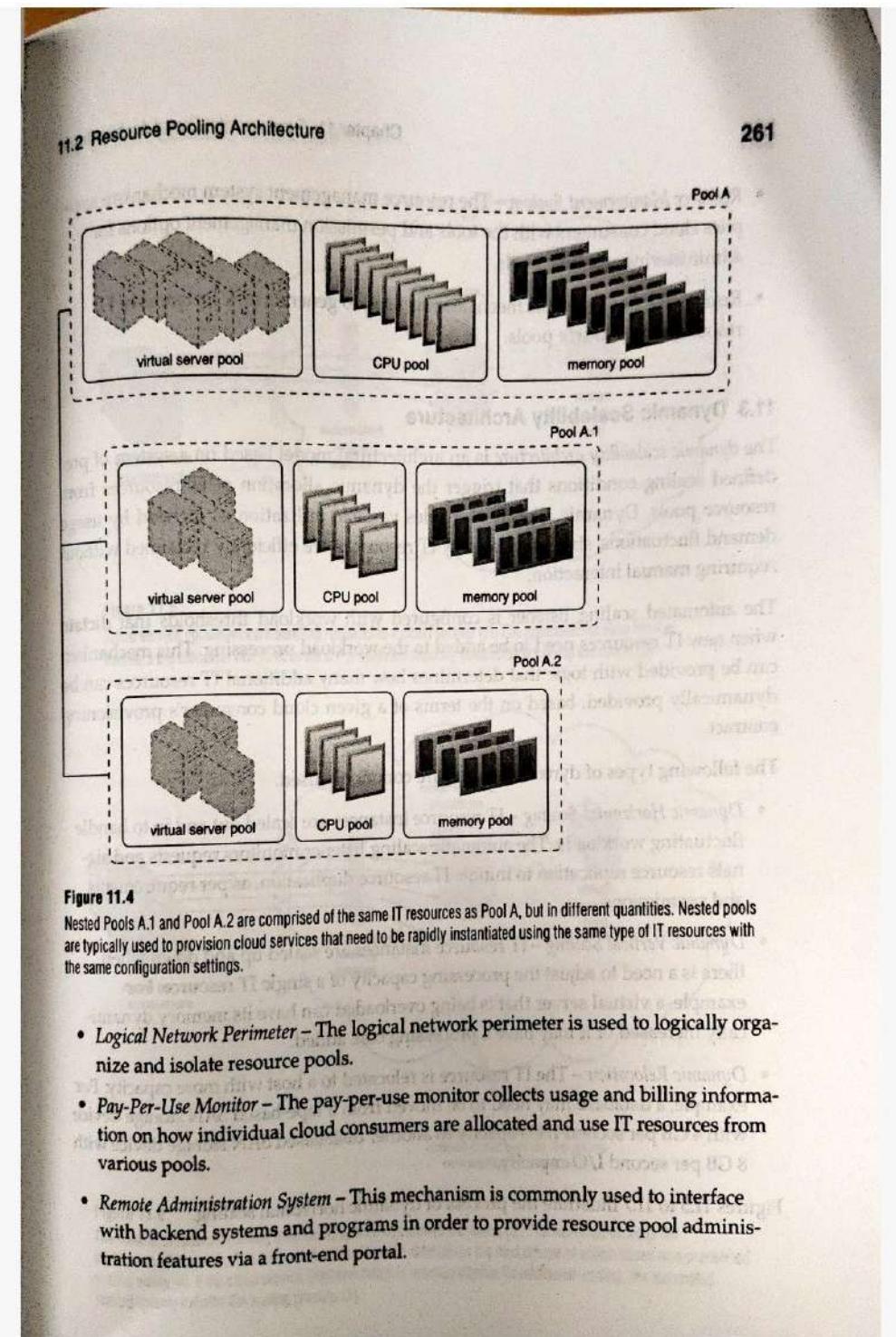


**Figure 11.3**

**Figure 11.3**

Pools B and C are sibling pools that are taken from the larger Pool A, which has been allocated to a cloud consumer. This is an alternative to taking the IT resources for Pool B and Pool C from a general reserve of IT resources that is shared throughout the cloud.

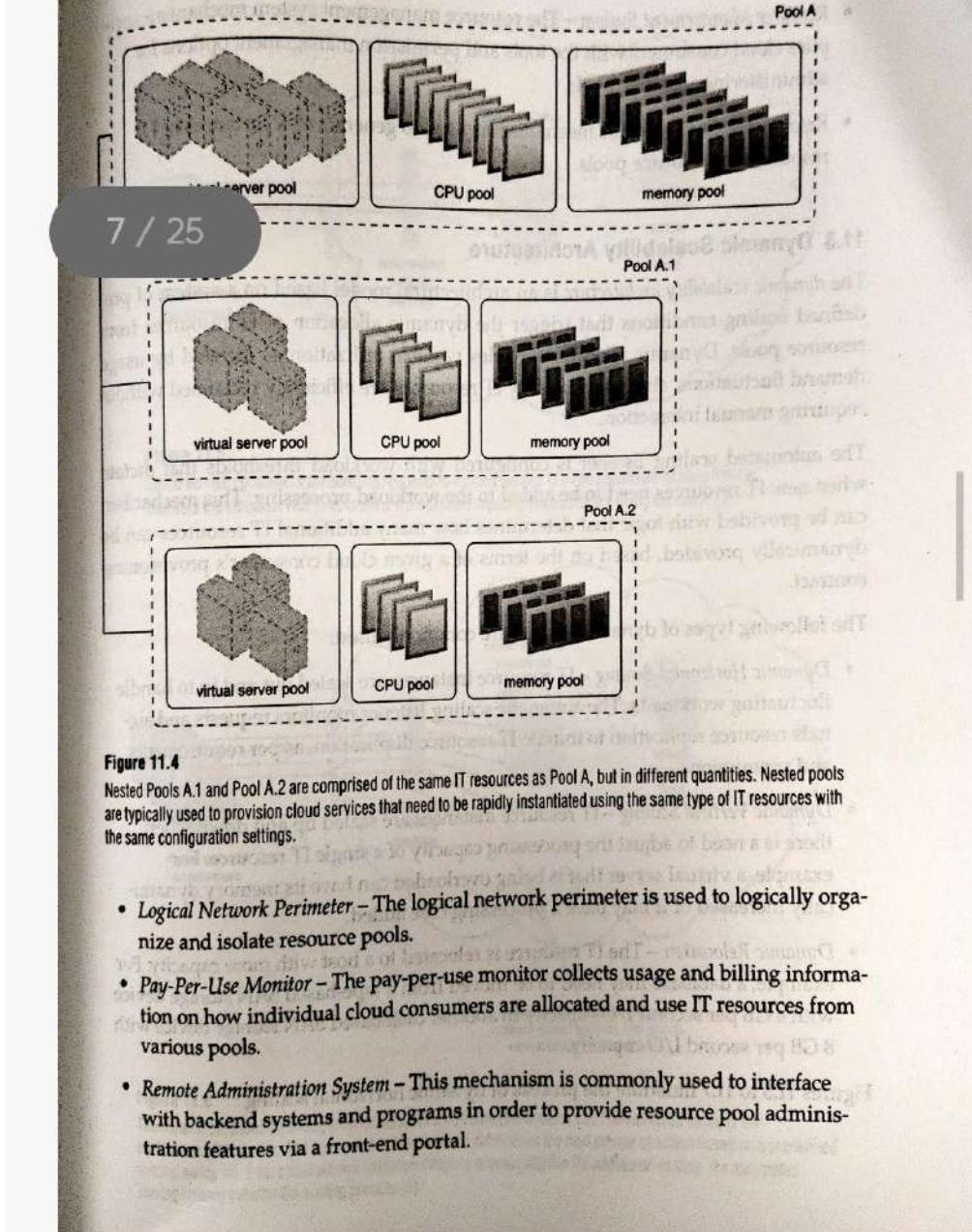
- *Cloud Usage Monitor* – Various cloud usage monitors are involved in the runtime tracking and synchronization that are required by the pooled IT resources and any underlying management systems.
- *Hypervisor* – The hypervisor mechanism is responsible for providing virtual servers with access to resource pools, in addition to hosting the virtual servers and sometimes the resource pools themselves.



**Figure 11.4**

Nested Pools A.1 and Pool A.2 are comprised of the same IT resources as Pool A, but in different quantities. Nested pools are typically used to provision cloud services that need to be rapidly instantiated using the same type of IT resources with the same configuration settings.

- *Logical Network Perimeter* – The logical network perimeter is used to logically organize and isolate resource pools.
- *Pay-Per-Use Monitor* – The pay-per-use monitor collects usage and billing information on how individual cloud consumers are allocated and use IT resources from various pools.
- *Remote Administration System* – This mechanism is commonly used to interface with backend systems and programs in order to provide resource pool administration features via a front-end portal.



- **Logical Network Perimeter** – The logical network perimeter is used to logically organize and isolate resource pools.
- **Pay-Per-Use Monitor** – The pay-per-use monitor collects usage and billing information on how individual cloud consumers are allocated and use IT resources from various pools.
- **Remote Administration System** – This mechanism is commonly used to interface with backend systems and programs in order to provide resource pool administration features via a front-end portal.

- **Resource Management System** – The resource management system mechanism supplies cloud consumers with the tools and permission management options for administering resource pools.
- **Resource Replication** – This mechanism is used to generate new instances of IT resources for resource pools.

### 11.3 Dynamic Scalability Architecture

The *dynamic scalability architecture* is an architectural model based on a system of pre-defined 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.

The following types of dynamic scaling are commonly used:

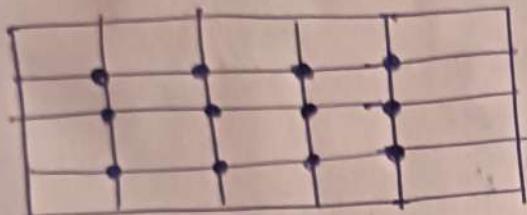
- **Dynamic Horizontal Scaling** – IT resource instances are scaled out and in to handle

## Adv -

Redundancy & failure.

## Grid Computing

Grid Computing provides a platform in which computing resources are organised into one or more logical pools.



These pools are collectively coordinated to

provide high performance distributed grids.  
It is also called super virtual computer.

## Cloud

A cloud refers to distinct IT environment that is designed for the purpose of remotely provisioning scalable & major IT resources.

The term originated from the visual rep. of the internet infrastructure in early n/w diagrams.

## ~~for~~ Components

~~and~~ IT resources - An IT resource is a physical or virtual IT related artifact that can be either software (such as virtual or custom software) or hardware based (such as physical servers or a physical device).

○ → service

□ → storage device

→ N/W device

cube → physical server

cube → virtual server

cloud → cloud

box → S/W programme

## ~~for~~ • On-Premise:

An IT resource that is hosted in a conventional IT enterprise within an organisational boundary is considered to be located on the premises of the IT enterprise or on-premise.

~~for~~ Cloud consumers v/s cloud providers

The party that provides cloud based

IT and  
the parties  
are called

~~the~~ Cloud

Here

~~water mode~~

SaaS  
PaaS  
IaaS

~~Info~~  
distri  
resour  
Automa

• PU

• R-

• A-

~~some~~  
Cloud

[ ]

1

2

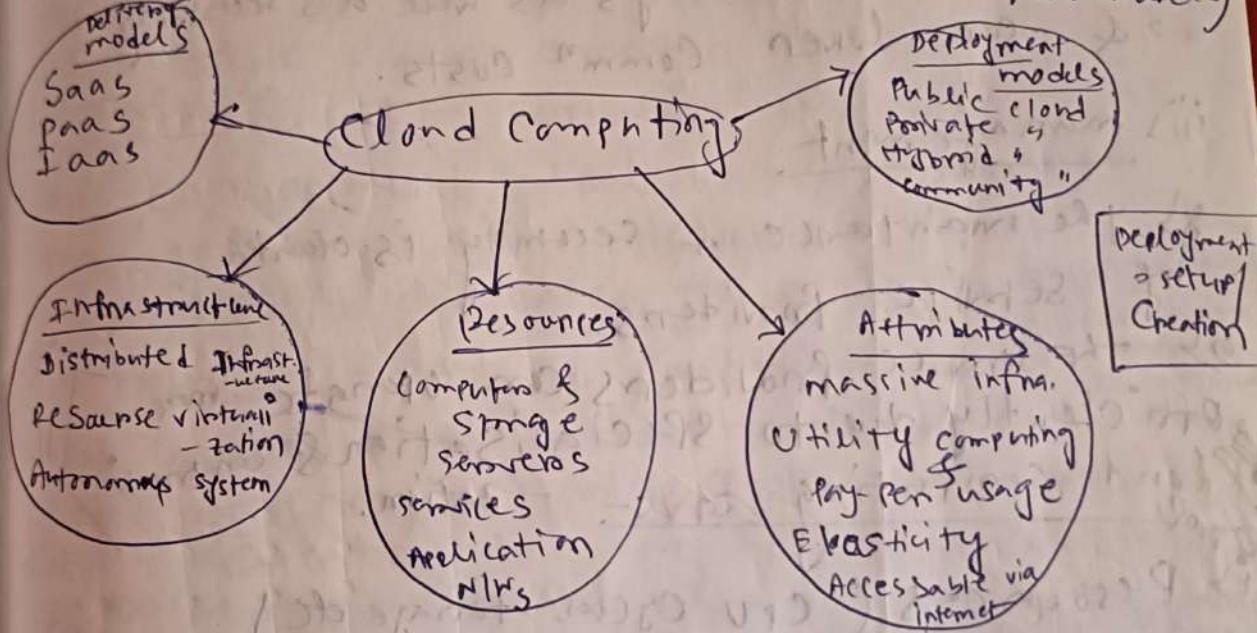
3

IT artifacts is the Cloud Providers, the party that uses cloud services that are called Cloud Consumers.

27/2/25

### Cloud computing models, resources & attributes

Here there are 3 models (they are old models, nowadays there are many new models)



- Public cloud ≈ SaaS / PaaS / IaaS (can be any)
- Resource virtualization ≈ Google Sheets (google drive)
- Autonomous system ≈ weather forecasting
- Cloud Computing - characteristics (P.T.O) ↗ for this

- ### Cloud Computing - characteristics (P.T.O)
- 1} Shared resources & resource management
  - 2} It uses shared pool of resources
  - 3} Scalable & elastic
  - 4} Elastic computing = The ability of dynamically provisioning computing resources & supporting

"iy Host

"iy guitar  
fan

"") Acc

### Some Cloud computing

It is the delivery of computing services ~~of computing services~~ <sup>in Sec</sup> (like servers, storage, databases, mining, SW) over the Internet <sup>by cos</sup> ("the cloud").

Characteristics - <sup>if</sup> on-demand access from <sup>ri</sup> or anywhere.

"pay-as-you-go" pricing model.

Scalable & flexible resources.

3 service models: PaaS, IaaS, SaaS.

Deployment models: Public, Private, Hybrid,  
Community cloud.

Real-time collaboration support.

Reduces hardware & infrastructure costs.

Data backup & recovery options.

There are certain services & models running  
behind the C.C.-I. Deployment model, 2. Service  
model.

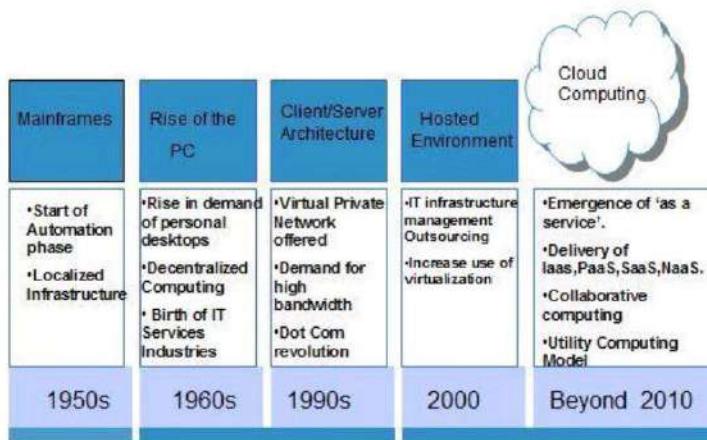
Deployment models - 4 types

### iii) management.

- 8) the maintenance & security is operated by service providers.
- 9) the service providers can operate more efficiently due to specialisation & centralization.  
Cloud computing advantages
- i) Resources (CPU cycles, storage etc.) are shared.
- ii) multiplexing leads to a higher resource utilization (when multiple users/applications share a system)
- iii) eliminates the initial investment costs.
- iv) cost reduction [concentration of resources creates the opportunity to pay as you go for computing].
- v) elasticity (the ability to accommodate workloads with very large peak-to-avg ratios)
- vi) user convenience (virtualisation allows users to operate in familiar envs [rather than idiosyncratic ones])
- vii) data sharing facilitates collaborative activities.

## History

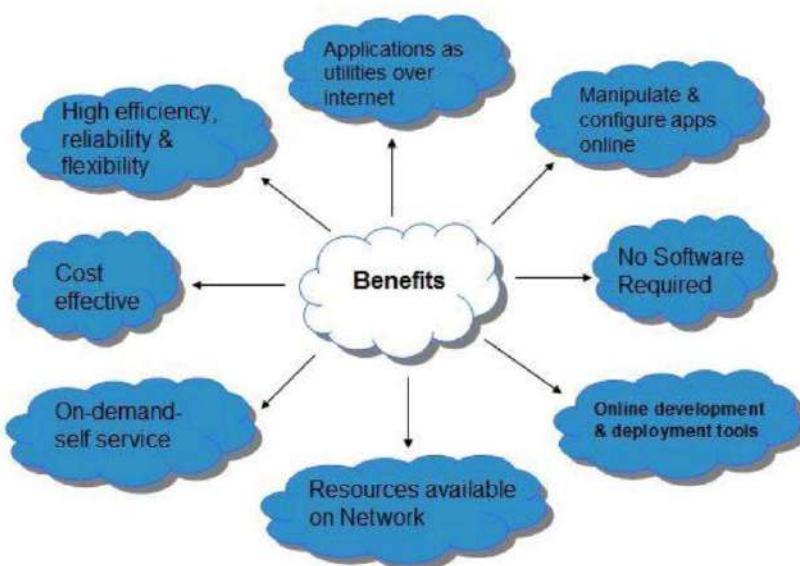
The concept of **Cloud Computing** came into existence in 1950 with implementation of mainframe computers, accessible via **thin/static clients**. Since then, cloud computing has been evolved from static clients to dynamic ones from software to services. The following diagram explains the evolution of cloud computing:



## Benefits

Cloud Computing has numerous advantages. Some of them are listed below:

- One can access applications as utilities, over the Internet.
- Manipulate and configure the application online at any time.
- It does not require to install a specific piece of software to access or manipulate cloud application.
- Cloud Computing offers online development and deployment tools, programming runtime environment through **Platform as a Service model**.
- Cloud resources are available over the network in a manner that provides platform independent access to any type of clients.
- Cloud Computing offers **on-demand self-service**. The resources can be used without interaction with cloud service provider.
- Cloud Computing is highly cost effective because it operates at higher efficiencies with greater utilization. It just requires an Internet connection.
- Cloud Computing offers load balancing that makes it more reliable.





Parameter	Grid Computing	Cloud computing
Goal	Collaborative sharing of resources	Use of service (eliminates the detail)
Computational focuses	Computationally intensive	Operations Standard and high-level instances
Level of abstraction	Low (more details)	High (eliminate details)
Degree of scalability	Normal	High
Multitask	Yes	Yes
Transparency	Low	High
Time to run	Not real-time	Real-time services

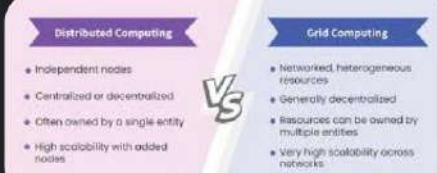
## Grid computing vs Cloud Computing | Download...

[Visit >](#)

Images may be subject to copyright. Learn More

[Share](#)
[Save](#)

Grid Computing	Cloud Computing
Means of utilization	Allocation of multiple servers onto a single task or job
Typical usage pattern	Typically used for job execution, i.e. the execution of a program for a limited time
Level of abstraction	Expose high level of detail
Task Size	Single Large
Computation/Service	Maximizes computing
Accessibility	Offers dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities.
Infrastructure	A decentralized system which spans across geographically distributed sites and lacks central control. It normally contains heterogeneous resources, such as hardware/software configurations, access interfaces and management policies.
Middleware	Self-tailored resilience with well-defined industry standards
Virtualization	Virtualization of data and computing resources



GeeksforGeeks

Distributed Computing...

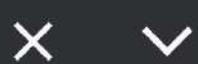
Criteria	Grid Computing	Cloud Computing
User Management	Decentralized user agreement	Centralized user agreement
Dependency	Other computers perform the work	Users depend on a common server
Operation	Business exists in a complex network	Can also operate through the internet
Availability	Through Grid middleware	Through standard Web protocols
Scalability	Multiple domains	Single domain
Architecture	Distributed computing architecture	Decentralized architecture
Virtualization	Data subpartitioning resources	Hardware and software partitions
Configuration	Machine-to-machine	Decentralized
Application Type	Batch	Interactive

DataFlair

Grid Computing Vs Clo...

Feature	Grid computing	Cloud computing
Business model	Resources sharing between users	Resources deliver services for every users according to their demands
Performance	Low	High
Scalability	Normal	High (resources are consumed according to users' demands)
Interoperability	Open Grid Forum Standards	Web services (SOAP/ReST)
Availability	Low	High
User management	Decentralized	Centralized or might be outsourced





## Cloud vs Grid vs Parallel Computing

chatgpt.com



ChatGPT

▼

Share



BL

Feature	Cloud Computing	Grid Computing	Parallel Computing
Definition	Delivery of computing services over the Internet	Distributed computing across multiple systems to complete a task	Simultaneous processing using multiple processors within a single system



Re

Ask anything



Search



Reason



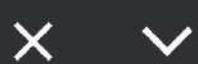
Deep research



Create image

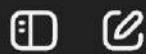


ChatGPT can make mistakes. Check important info. See [Cookie Preferences](#).



# Cloud vs Grid vs Parallel Computing

chatgpt.com



ChatGPT

Share



BI

Resource Location	Centralized or distributed over the Internet	Geographically distributed systems	Typically localized (same machine or cluster)
Scalability	Highly scalable	Moderately scalable	Limited by hardware
Architecture	Client-server, virtualized	Peer-to-peer or decentralized	Shared memory or distributed

Ask anything



Search



Reason



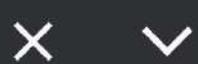
Deep research



Create image



ChatGPT can make mistakes. Check important info. See [Cookie Preferences](#).

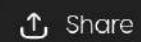


## Cloud vs Grid vs Parallel Computing

chatgpt.com



ChatGPT

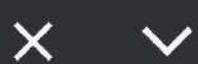


Resource Management	Managed by cloud provider	Managed by grid middleware	Managed by application or OS
Usage Model	Pay-as-you-go (utility model)	Shared resources for specific tasks	Single task split across cores/processors
Fault Tolerance	High (due to redundancy)	Medium (depends on grid)	Low to medium

Ask anything

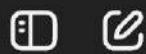


ChatGPT can make mistakes. Check important info. See [Cookie Preferences](#).



# Cloud vs Grid vs Parallel Computing

chatgpt.com



ChatGPT

Share



BI

Example Use Cases

Web hosting, SaaS, data storage

Scientific computing, academic research

Real-time simulations, image processing

Example

AWS, Azure, Google Cloud

BOINC, Globus Toolkit

MPI, OpenMP

Providers/Systems



Ask anything



Search



Reason



Deep research



Create image



ChatGPT can make mistakes. Check important info. See [Cookie Preferences](#).



# Virtualized Resource

In subject area: [Computer Science](#)

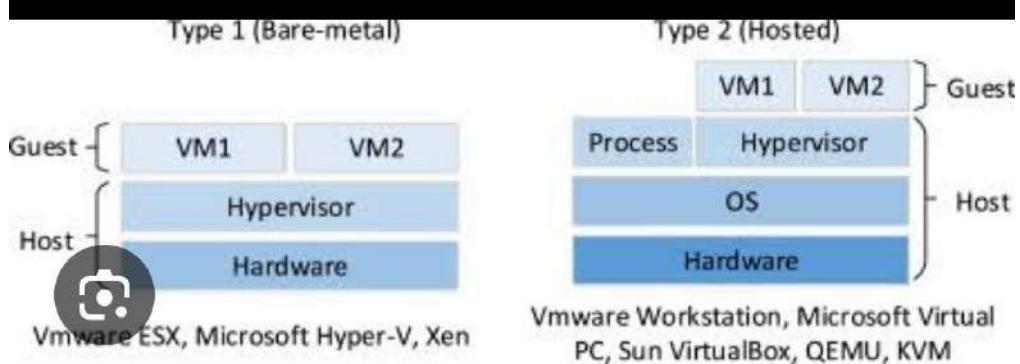
A 'Virtualized Resource' refers to a resource that is abstracted from the physical infrastructure through virtualization technologies, allowing it to be dynamically allocated and migrated across different physical machines in Cloud Computing environments.

AI generated definition based on: [Computer Networks, 2013](#)

## On this page ▾

## Chapters and Articles

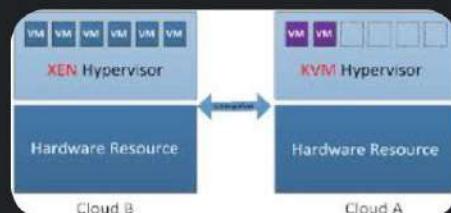
You might find these chapters and articles relevant to this topic.



## Virtualized Resource - an overview | ScienceDirect...

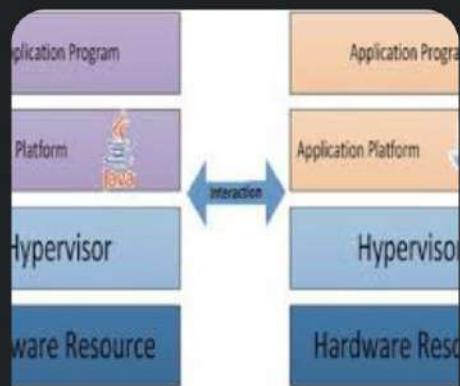
[Visit >](#)

Images may be subject to copyright. Learn More

[Share](#)[Save](#)

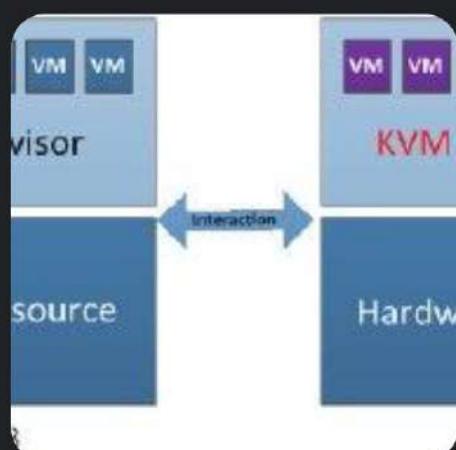
R<sup>®</sup> ResearchGate

Open Virtualization For...



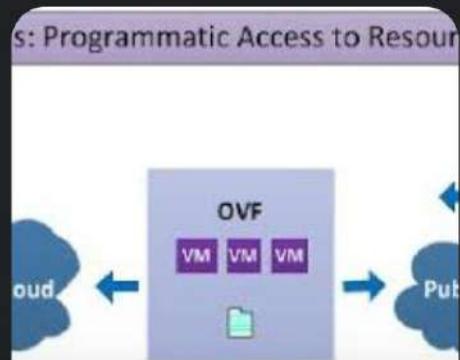
R<sup>®</sup> ResearchGate

Interoperability in Infra...



R<sup>®</sup> ResearchGate

Open Virtualization For...



~~processing~~

Chap - 3  
 $Pg = 26 - 44$  (Book from which we need to study)  
ITP  
Scaling

Scaling, from an IT resource perspective  
refers to the ability of the IT resource  
to handle increased or decreased usage

demands. There are 2 types of scaling; horizontal & vertical.

i) Horizontal — Scaling out & Scaling in

= Scaling up & Scaling down

~~Imp~~ ✓  
Horizontal

i) Scaling out & scaling in.

ii) Allocating & releasing of IT resources that if the ~~the~~ type is referred to as horizontal scaling.

The horizontal allocation of resources is referred to as scaling out & h.

Releasing of resources is referred to as scaling in. It is a common form of scaling within the cloud environment.

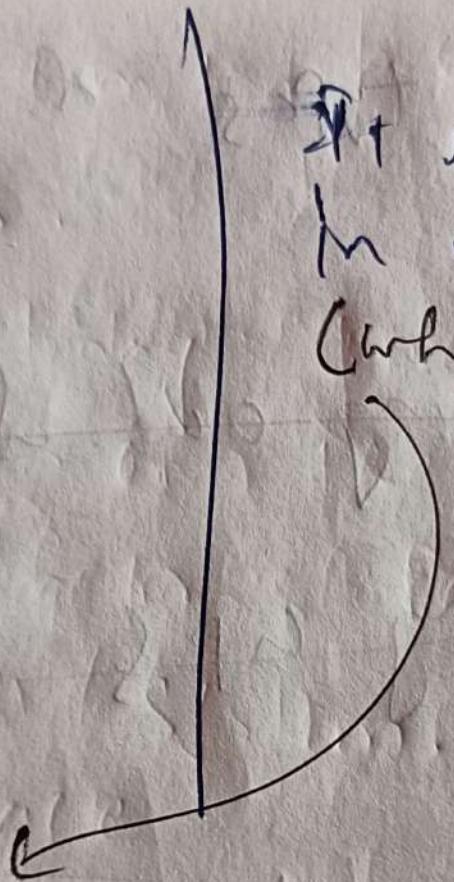
vertical

i) Scaling up & scaling down

ii) when an existing IT resource is replaced by another with higher or lower capacity, then it's called vertical scaling. The replacing of an IT resource with another that has a higher capacity is referred to as scaling up &

the replacing of the IT resource with another that has a lower capacity is referred to as scaling down.

It is less common  
in c. environment.  
(why?)



- H. S. i vertical Scanning is less common in cloud environment.  
Why?

This is less common in cloud envt.

-rs for intrastucture management.

- Integration challenges - Compatibility with existing on-premise systems is low.
- Performance issues - Latency & speed variations based on n/w conditions.
- Cloud delivery (deployment) model

(i) IaaS, ii) PaaS, iii) SaaS

The three main cloud delivery models are Infrastructure as service (IaaS), PaaS, SaaS.

i) IaaS - It provides the ability to use the computing & storage resources.  
Eg - Google drive etc.

ting ; its adoption by  
s, government etc eng.  
economy.

### Segment reasons

consists of a mostly  
hardware & S/W

in a single administrative

breaches,

& compliance issues.

- Potential service  
operations.

affected costs due to

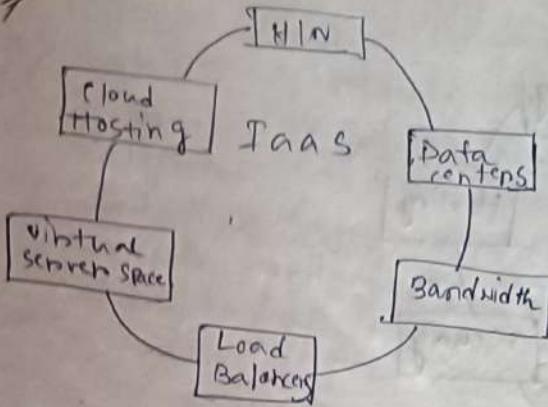
dependence on cloud provider  
management.

Compatibility with  
existing systems is low.

& speed variations  
model based on W/N  
conditions.

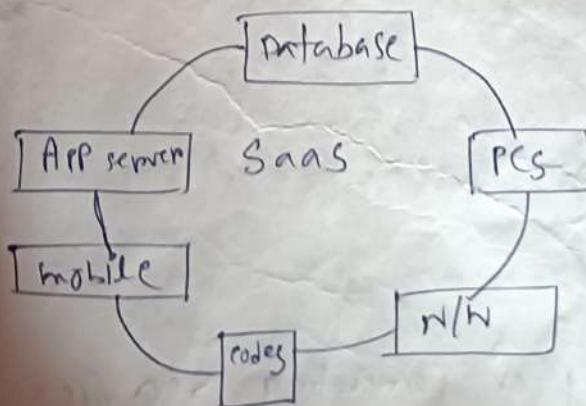
Delivery models are  
SaaS, PaaS,

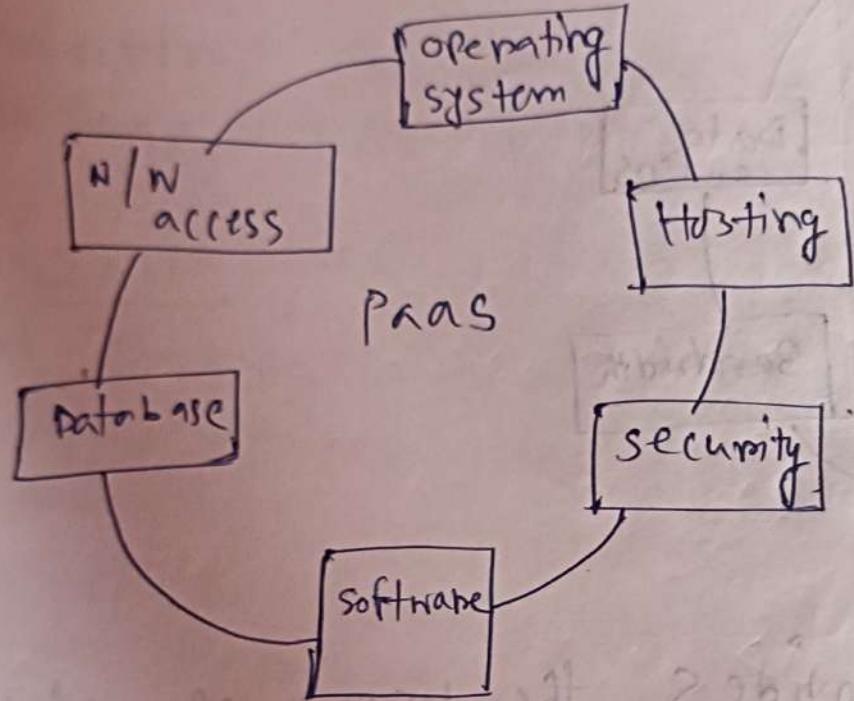
ability to use  
resources.



i) PaaS - It provides the users the ability to develop & deploy application in clouds using the development tools, APIs, SW libraries & services provided by the cloud service provider (CSP). Eg - ~~Ans~~ IBM cloud, Red Hat OpenShift etc.

ii) S/W as a service (SaaS) - Provides users a complete SW application / the user interface to the application. Eg - google sheet, Powerpoint etc.





Qn

16/4/25

- Cloud → manipulation  
↓  
(or)  
now → accessing the services provided  
configuration by cloud



Variables	SaaS	PaaS	IaaS
Full form	Software as a service	Parameter as a service	Infrastructure as a service
Used for	It is used for cloud computing software development	It is used for application development in cloud computing	It is used for any cloud computing development
Type of users	Any users who use it	It is used by software and application developers	Network architects use it
Access	SaaS gives access to any software that you want to use	It provides the necessary tools for development	You can store data and also use virtual machines
Technical know-how	If you want to use SaaS, you do not have to be a technical expert	You need to have a piece of proper knowledge of developing applications	You need to have excellent technical knowledge
Popularity	Most consumers and businesses focus on SaaS	PaaS is primarily popular among application developers	IaaS is popular among researchers and developers
Cost	The cost of using SaaS is minimum as you use the software according to your requirements	The cost of using PaaS is low	The cost of using IaaS is the highest



## SaaS Vs PaaS Vs IaaS: What's the Difference...

[Visit >](#)

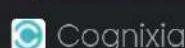
Images may be subject to copyright. Learn More

[Share](#)
[Save](#)

Difference between SaaS, IaaS, & PaaS			
Parameters	SaaS	IaaS	PaaS
Abbreviation	Software as a Service	Infrastructure as a Service	Platform as a Service
Potential Users	It is used by end users	Network architects use it	Mainly used by developers
Prerequisite-Technical knowledge	Does not need technical knowledge	Basic technical knowledge is needed	Requires advanced technical knowledge
Conception	Total abstraction	Abstraction only for underlying hardware resources	Abstraction of the underlying hardware, software, and application services
Level of control	Total control and flexibility can be exercised	Minimum control	Modest control can be exercised
Costs	Maintenance cost is avoided	High costs are involved	Low costs are involved
Examples-Cloud Computing	Facebook, Microsoft Office, and Dropbox	AWS, Azure, Google Cloud, Microsoft Azure and Google Cloud	Oracle Cloud, OpenShift, and AWS Elastic Beanstalk

Attribute	IaaS	PaaS	SaaS
Benefits	<ul style="list-style-type: none"> <li>Highly flexible and scalable</li> <li>Easily accessible by multiple users</li> <li>Cost effective</li> <li>Ideal for businesses of all sizes</li> <li>Provides complete &amp; disconnected private user infrastructure</li> <li>Can help slash hardware costs by 50% or more</li> </ul>	<ul style="list-style-type: none"> <li>It's a whole platform designed to work with preferred vendors like Amazon, Microsoft, Oracle, SAP, Salesforce, etc. to get custom solutions</li> <li>No vendor dependency effort</li> <li>Doesn't need an internal development team to build application codes from the scratch</li> <li>Highly beneficial for large enterprises with specialized applications &amp; requirements</li> </ul>	<ul style="list-style-type: none"> <li>No requirement for infrastructure deployment, software development or delivery, maintenance or operations</li> <li>Follows fault-tolerance</li> <li>Easy scaling</li> <li>Requires minimal input from users</li> </ul>
Limitations	<ul style="list-style-type: none"> <li>Requires a mature operations model</li> <li>Needs rigorous vendor integration</li> <li>Needs understanding cloud provider technologies</li> <li>Requires skill &amp; competency in resource management</li> </ul>	<ul style="list-style-type: none"> <li>Entails a vendor lock-in in terms of features</li> <li>Cross-vendor extreme compatibility</li> <li>Hard to migrate from one platform to another</li> </ul>	<ul style="list-style-type: none"> <li>Hard to do custom workflows</li> <li>Costs depending on the service provider to make any improvements in features and reliability</li> </ul>



Learn the differences ...



Differences between S...



Home



Search



Saved



Notifications

Types of clouds - Computing deployment models

- deployment models are described below:

FaaS, PaaS,  
SaaS =  
C.C. models  
= delivery n

↳ means deployment + models

i) Public - It's a commonly adopted cloud model, where the cloud services provider owns the infrastructure & openly provides access to it for the public to consume.

ii) Private - It can be thought of as an environment that is fully owned & managed by a single tenant.

iii) Hybrid - It's a computing env. that is also known as corporate on internal cloud. It's a c.c. env. that is dedicated to a single organization.

iv) Others - Eg - Netflix uses a hybrid c. m. to store & manage some on-premises data centers. It allows data & applications to move b/w the different envs.

v) Others types; Community / Federated - Eg - Azure stack.

It's a c. infrastructure in which multiple organizations share resources & services based on common operational & regulatory requirements. It's a semi-public cloud. Eg - Heroku etc.

• Why C.C. is successful, when others failed?

i) It's in a better position to exploit recent advances in SW, h/wing, storage & processor technologies promoted by the same companies who provide cloud services.

There are certain services behind the C.C. - 1. Deployment model, 2. Service model.

Deployment models - 4 types

- i) Public cloud, ii) Private cloud, iii) Hybrid cloud,  
iv) Community cloud (Done Pm)

Diff. b/w Private & Public cloud (vice versa)  
Adv & disadv of each deployment models  
(other clouds)

Features	Public	Private	Hybrid	Community
i) Host	service provider	Enterprise	Enterprise	Community (3rd Party)
ii) Suitable for	Large Enterprise	Large enterprise	Small & mid-size	Financial, health & legal companies.
iii) Access	Internet	Internet, VPN, Intranet	Intranet, VPN	Intranet, VPN
iv) Security	Low	most Secured	Moderate	Secured
v) Cost	cheapest	High cost	cost effective	Cost effective
vi) Owner	Service Provider	Enterprise	Enterprise	Community

	Very reliable	Moderate	Very High	Medium-High	Very High
VII) Scalability	Very High	Limited	Very High	Limited	
VI) Users	Organization, public etc.	Business organizations	Business org.	Community members	
V) Deployment Models					

### Adv & disadvantages of each deployment models -

#### i) Public

Adv - i) Cost effective, ii) Scalable & flexible, iii) No maintenance is needed by users, iv) Fast setup & deployment

Disadv - i) Less control over infrastructure, ii) Potential security concerns, iii) Limited customization, iv) Shared resources can affect performance.

#### ii) Private

Adv - i) Greater control & customization, ii) Enhanced security & privacy, iii) Better performance, iv) Strict regulations.

Disadv - i) High cost, ii) Requires in-house expertise, iii) Longer deployment time, iv) Scalability can be limited.

#### iii) Hybrid

Adv - i) Flexible & Scalable, ii) Balanced cost & control, iii) Improved business continuity, iv) Allows sensitive data to stay on private cloud.

Disadv - i) Complex management, ii) Integration challenges, iii) Security risks in data transfers, iv) Higher setup costs than Public cloud.

#### iv) Community

Adv - i) Cost effective, ii) Better security than Public cloud, iii) Shared responsibilities & costs, iv) Customizable for group needs.

Disadv - i) Limited Scalability, ii) Governance challenges, iii) Costly than Public, iv) Slow adoption & updates.

Service model  
the cl  
types.

Prash  
Rahul

Usin  
Sen

W

Bon  
Am  
21



ChatGPT ▾



Hypervisor Type 1 (Bare Metal Hypervisor):

- Installed directly on physical hardware (no operating system needed).
- Manages virtual machines (VMs) directly.
- Examples: VMware ESXi, Microsoft Hyper-V, Xen.

How it works:

1. Installed on the server hardware.
2. Creates and runs multiple VMs.
3. Each VM runs its own operating system.
4. Offers high performance and better security.

Hypervisor Type 2 (Hosted Hypervisor):



- Installed on top of a regular operating system

Ask anything





ChatGPT



## Hypervisor Type 2 (Hosted Hypervisor):

- Installed on top of a regular operating system (like Windows or Linux).
- Runs VMs like any other software application.
- Examples: VMware Workstation, Oracle VirtualBox.

### How it works:

1. Installed on an existing OS (e.g., Windows).
2. Creates VMs as applications on the OS.
3. Easier to set up but not as fast or secure as Type 1.



W

Ask anything





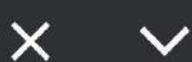
ChatGPT ▾



Feature	Amazon SimpleDB	Amazon RDS
Type	NoSQL (Key-Value Store)	Relational Database Service
Data Model	Schema-less, attribute-value pairs	Structured, schema-based tables
Query Language	Proprietary SimpleDB query language	Standard SQL
Use Case	Lightweight, non-relational data storage	Complex, relational data workloads
Scalability	Automatically scales horizontally	Vertical and limited horizontal scaling (read replicas)

Ask anything





ChatGPT ▾



Scalability	Automatically scales horizontally	Vertical and limited horizontal scaling (read replicas)
-------------	-----------------------------------	---

Transaction Support	Limited support	Full ACID transaction support
---------------------	-----------------	-------------------------------

Consistency	Eventual consistency	Strong consistency
-------------	----------------------	--------------------

Indexing	Automatic indexing on all attributes	User-defined indexes
----------	--------------------------------------	----------------------

Backup & Recovery	Manual via application	Automated backups, snapshots, point-in-time recovery
-------------------	------------------------	--

Availability	Designed for high availability	Supports Multi-AZ deployments for high availability
--------------	--------------------------------	---



S'

Ask anything





ChatGPT ▾



## 1. Standard Environment

- Languages supported:
  - Python
  - Java
  - Node.js
  - Go
  - PHP
  - Ruby
- Fast startup, automatic scaling, limited customization.

## 2. Flexible Environment

- Languages supported:
  - Same as Standard (Python, Java, Node.js, etc.)



Ask anything





ChatGPT ▾



- Languages supported:
  - Same as Standard (Python, Java, Node.js, etc.)
  - Plus: .NET Core and any language using Docker
- More control, supports custom runtimes, slower startup, scales with containers.

### 3. Custom Runtimes

- Use your own Dockerfile
- Deploy any language or tool (as long as it runs in Linux)

Want a quick table comparing them?



Ask anything



### 3. Storage

AppEngine provides different types of storage, which operate differently according to the volatility of the data they are designed for. There are three different levels of storage: in memory-cache, storage for semi-structured data, and long-term storage for static data. In this section, we describe DataStore and the use of static file servers and we cover MemCache in the application services section.

**(a) Static File Servers.** Web applications are composed by dynamic and static data. Dynamic data is a result of the logic of the application and the interaction with the user. Static data often is mostly constituted by the components that define the graphical layout of the application (css files, plain html files, javascript files, images, icons, and sound files) or data files. These files can be hosted on static file servers, since they are not frequently modified. Such servers are optimized for serving static content and users can specify how dynamic content should be served when uploading their applications to AppEngine.

**(b) DataStore.** DataStore is a service allowing developers to store semi-structured data. The service is designed to scale and optimized to quickly access data. DataStore can be considered as a large object database where to store objects that can be retrieved by a specified key. Both the type of the key and the structure of the object can vary.

With respect to the traditional Web applications backed by a relational database, DataStore imposes less constraint on the regularity of the data but, at the same time, does not implement some of the features of the relational model (such as reference constraints and join operations). These design decisions originated from a careful analysis of data usage patterns for Web applications and have been taken in order to obtain a more scalable and efficient data store. The underlying infrastructure of DataStore is based on Bigtable [93], which is a redundant, distributed, and semi-structured data store that organizes data in the form of tables (See Section 8.2.1).

DataStore provides high-level abstractions that simplify the interaction with Bigtable. Developers define their data in terms of *entity* and *properties*, and these are persisted and maintained by the service into tables in Bigtable. An entity constitutes the level of granularity for the storage and it identifies a collection of properties, which define the data it stores. Properties are defined according to one of the several primitive types supported by the service. Each entity has associated a key, which is either provided by the user or created automatically by AppEngine. An entity is associated with a *named kind* that is used by AppEngine to optimize its retrieval from Bigtable. Although entities and properties seem to be similar to rows and tables in SQL, there are a few differences that have to be taken account. Entity of the same kind might not have the same properties, and properties of the same name might contain values of different types. Moreover, properties can store different version of the same values. Finally, keys are immutable elements and once created, they cannot be changed anymore.

DataStore also provides facilities for creating indexes on data and to update data within the context of a transaction. Indexes are used to support and speed-up queries. A query can return zero or more objects of the same kind or simply the corresponding keys. It is possible to query the data store either by specifying the key or conditions on the values of the properties. Returned result sets can be sorted either by key value or properties value. Even though the queries are quite similar to SQL queries, their implementation is substantially different. DataStore has been designed to be extremely fast in returning result sets. In order to do so, it needs to know in advance all the possible queries that can be done for a given kind because it stores for each of them a separate index. The indexes are provided by the user while uploading the application to AppEngine and can be automatically defined by the development server: when the developer tests the application, the server monitors all the different types of query made against the simulated data store and creates an index for them. The structure of the indexes is saved into a configuration file and can be further changed by the developer before uploading the application. The use of pre-computed indexes makes the query execution time independent from the size of the stored data, but only influenced by the size of the result set.

The implementation of transaction is limited in order to keep the store scalable and fast. AppEngine ensures that the update of a single entity is performed automatically. Multiple operations on the same entity can be performed within the context of a transaction. It is also possible to update multiple entities automatically. This is only possible if these entities belong to the same *entity group*. The entity group to which an entity belongs is specified at the time of entity creation and cannot be changed later. With regards to concurrency, AppEngine uses an *optimistic concurrency control*: if one user tries to update an entity that is being updated, the control returns and the operation fails. Retrieving an entity never incurs into exceptions.

### 4. Application Services

Applications hosted on AppEngine take the most from the services made available through the runtime environment. These services simplify most of the common operations that are performed in Web applications: access to data, account management, integration of external resources, messaging and communication, image manipulation, and asynchronous computation.

**(a) UrlFetch.** Web 2.0 has introduced the concept of composite Web applications. Different resources are put together and organized as meshes within a single Web page. Meshes are fragment of HTML generated in different ways: they can be directly obtained from a remote server, they rendered from an XML document retrieved from a Web services or rendered by the browser as the result of an embedded and remote component. A common characteristic of all these examples is the fact that the resource is not local to the server, and often not even in the same administrative domain. It is therefore fundamental for Web applications to be able to retrieve remote resources.

The sandbox environment does not allow applications to open arbitrary connections through sockets, but provides developers with the capability of retrieving a remote resource through HTTP/HTTPS by means of the *UrlFetch* service. Applications can make synchronous and asynchronous Web requests and integrate the resources obtained in this way into the normal request handling cycle of the application. One of the interesting features of *UrlFetch* is the ability to set a deadline for a request, so that they can be completed (or aborted) within a given time. Moreover, the ability of performing such requests asynchronously allows the applications to continue with their logic while the resource is retrieved in background. *UrlFetch* is not only used to integrate meshes into a Web page, but also to leverage remote Web services in accordance with the SOA reference model for distributed applications.

**(b) MemCache.** AppEngine provides developers with access to a fast and reliable storage, which is used to store the results of the execution of the pipeline in a persistent manner.



ChatGPT ▾



## 1. Primitive Data Types

- String
- Integer
- Float (Double)
- Boolean (True/False)
- Null

## 2. Date and Time

- Timestamp (date and time with timezone info)

## 3. Complex Types

- List/Array (of values of the supported types)
- Embedded Entity (a nested dictionary/object)
- GeoPoint (latitude and longitude)
- Blob (binary data like images, files)



Ask anything





ChatGPT ▾



- Timestamp (date and time with timezone info)

### 3. Complex Types

- List/Array (of values of the supported types)
- Embedded Entity (a nested dictionary/object)
- GeoPoint (latitude and longitude)
- Blob (binary data like images, files)

### 4. Key

- Used to identify and reference entities (like primary key)

Each entity in Datastore is like a flexible JSON object  
— it doesn't need a fixed schema.

Would you like an example entity structure?



Ask anything



25/2/25

### • cluster computing

A cluster is a group of independent IT resources that are interconnected & work as a single system.

#### DisAdv -

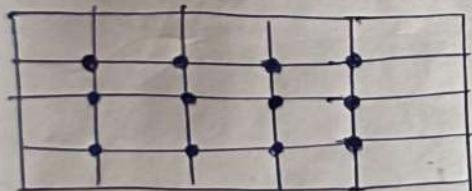
Availability of the resources.

#### DisAdv -

Redundancy & failure.

### • grid computing

Grid Computing provides a platform in which computing resources are organised into one or more logical pools.



These pools are collectively coordinated to

provide high performance distributed grids.  
It is also called super virtual computer.

### • cloud

A cloud refers to distinct IT environment that is designed for the purpose of remotely provisioning scalable & major IT resources.

The term is originally from the visual



Feature	Internet	Cloud
Definition	A global network connecting computers	A service that provides computing resources via the internet
Function	Enables communication and data sharing	Offers storage, computing power, and software services
Access	Used to access websites, email, social media, etc.	Used to access online services like Google Drive, AWS, Microsoft Azure
Infrastructure	Physical network of servers, cables, and devices	Virtual infrastructure managed by cloud providers



Depend on me

Ask anything





Infrastructure	Physical network of servers, cables, and devices	Virtual infrastructure managed by cloud providers
Dependency	Independent; cloud depends on the internet	Dependent on the internet to function
Example	Browsing websites, sending emails	Storing files on OneDrive or using Google Docs

Let me know if you'd like this in PDF or image format for easier sharing.

Ask anything





S.NO	PARALLEL COMPUTING	DISTRIBUTED COMPUTING
1.	In a parallel system, many operations can be performed at the same time, or simultaneously.	In a distributed system, the various components are located at different locations or places.
2.	Single computer is required.	Uses multiple computers.
3.	Multiple operations are performed by multiple processors.	Several computers carry out multiple operations.
4.	Communication between processors occurs through a bus.	Communication between computers occurs through message passing.
5.	It may have shared or distributed memory	It has only distributed memory
6.	Enhances the efficiency of the system.	Enhances the system's ability to handle more workload, recover from failures, and share resources effectively.

## Parallel and Distributed Computing | by Sumedh...

[Visit >](#)

Images may be subject to copyright. Learn More

[Share](#)
[Save](#)

Parallel Computing	Distributed System
Parallel computing is a type of computation where multiple operations or processes are performed simultaneously.	Distributed systems is a network of systems where components communicate with each other to coordinate their work together.
Only one computer performs all the available processes no action is common memory pool, while distributed memory provides local memory.	Multiple computers work together at the same time.
Multiple computers work together at the same time, it is used to share processes to increase availability with each other by creating regular connections between them.	Each computer has own memory.
Improves the performance of the system.	Computers communicate with each other via a network by sending and receiving data packets over a shared or dedicated communication link.

DISTRIBUTED SYSTEM	PARALLEL COMPUTING
Different system components are located in different locations that perform assigned tasks.	Many processes are performed concurrently using multiple processing components.
Multiple computers in a network are used for distributed computing.	A single computer is needed for parallel computing.
Multiple computers systems perform multiple processes.	Multiple processors perform multiple processes.
Each computer in a distributed system has their own memory.	This system may have shared or distributed memory.
Multiple Computers inter communicate through message passing via a network.	Inter Processor communication is done through system buses.
Allows resource sharing to perform computations efficiently.	Parallel computing improves the system's performance.
Mainly used for Resource sharing and increasing scalability.	Mainly used for Scientific, Complex Computing.

Medium

Distributed Systems an...

Medium

Distributed System and...

Distributed Computing	Grid Computing
<ul style="list-style-type: none"> <li>Independent nodes</li> <li>Centralized or decentralized</li> <li>Often owned by a single entity</li> <li>High scalability with added nodes</li> </ul>	<ul style="list-style-type: none"> <li>Networked, heterogeneous resources</li> <li>Generally decentralized</li> <li>Resources can be owned by multiple entities</li> <li>Very high scalability across networks</li> </ul>

Parallel Computing
<ul style="list-style-type: none"> <li>Shared memory system</li> <li>Multiprocessor (multiple processes share a single bus and memory unit)</li> <li>Symmetric Multiprocessing (SMP) supports in Operating System.</li> </ul>

Distributed Computing
<ul style="list-style-type: none"> <li>Distributed memory system</li> <li>Multicomputer (Autonomous computer nodes connected via network)</li> <li>Share any of the challenges</li> <li>More reliability and cheaper</li> </ul>





	Cloud Computing	Distributed Computing
Definition	Cloud computing defines a new way of computing based on the network technology. Cloud computing takes place over the common network like internet. It usually comprises of a collection of integrated and networked hardware, software and internet infrastructure resources.	Distributed computing contains multiple software components from multiple different computers which work together as a single system. Cloud computing can be referred as a virtualization achieved from distributed computing.
Goals	<ul style="list-style-type: none"> <li>* Reduced Initial Investment and Proportional Costs</li> <li>* Increased Scalability</li> <li>* Increased Availability</li> <li>* Increased Reliability</li> </ul>	<ul style="list-style-type: none"> <li>* Resource Sharing</li> <li>* Openness</li> <li>* Transparency</li> <li>* Scalability</li> </ul>
Types	<ul style="list-style-type: none"> <li>* Public Clouds</li> <li>* Private Clouds</li> <li>* Community Clouds</li> <li>* Hybrid Clouds</li> </ul>	<ul style="list-style-type: none"> <li>* Distributed Computing Systems</li> <li>* Distributed Information Systems</li> <li>* Distributed Pervasive Systems</li> </ul>
Characteristics	<ul style="list-style-type: none"> <li>* It provides a shared pool of configurable computing resources.</li> <li>* An on-demand network model is used to provide access</li> <li>* The clouds are provisioned by the Service Providers.</li> <li>* It provides broad network access.</li> </ul>	<ul style="list-style-type: none"> <li>* A task is distributed amongst different machines for the computation job at the same time.</li> <li>* Technologies such as Remote Procedure calls and Remote Method Invocation are used to construct distributed computations.</li> </ul>
	<ul style="list-style-type: none"> <li>* More elasticity means less control especially in the case of public clouds.</li> <li>* Restrictions on available services may be faced, as it depends upon the cloud provider.</li> </ul>	<ul style="list-style-type: none"> <li>* Higher level of failure of nodes than a dedicated parallel machine.</li> <li>* Few of the algorithms are not able to match with slow networks.</li> <li>* Nature of the computing job may present too much overhead.</li> </ul>

## Cloud versus distributed computing - Distributed...

[Visit >](#)

Images may be subject to copyright. Learn More

[Share](#)
[Save](#)

	Grid Computing	Cloud Computing
Means of utilization	Allocation of multiple servers onto a single task or job.	Virtualization of servers one server to compute several tasks concurrently.
Typical usage patterns	Typically used for job execution, i.e. the execution of a program for a limited time.	More frequently used to support long running services
Level of abstraction	Expose high level of detail	Provide higher-level abstractions
Task Size	Single Large	Small and medium
ComputationService	Minimum computing	On-demand
Accessibility	Offers dependable, consistent, periodic and inexpensive access to high-end computational capabilities	Offers customized, reliable and QoS guaranteed computing environments for users with an easy and pervasive access.
Infrastructure	A decentralized system, which spans across geographically distributed sites and lack central control. It normally consists heterogeneous resources, such as hardware/software configurations, access interfaces and management policies.	A central computer server with single access point and spans several computing centers, like Google and Amazon, in general consist homogeneous resources. Operated under central control.
Middleware	Full-fledged middleware with well-defined industry standards	still underdeveloped and lacks of standard
Virtualization	Virtualization of data and computing resources	Virtualization of hardware and software platforms

CLOUD COMPUTING	DISTRIBUTED COMPUTING
- CLOUD COMPUTING PROVIDES ON-DEMAND SERVICES.	- DISTRIBUTED COMPUTING REFERS TO DIVIDING A PROBLEM OVER DISTRIBUTED AUTONOMOUS COMPUTERS.
- IT HAS 4 DIFFERENT TYPES: PUBLIC CLOUD, PRIVATE CLOUD, COMMUNITY CLOUD, AND HYBRID CLOUD.	- IT HAS 3 DIFFERENT TYPES: DISTRIBUTED COMPUTING SYSTEMS, DISTRIBUTED INFORMATION SYSTEMS, AND DISTRIBUTED PERVERSIVE SYSTEMS.
- IT IS PROVIDED AS A SERVICE OVER THE INTERNET TO OTHER BUSINESS FIRMS.	- IT ALLOWS MANY COMPUTERS TO COMMUNICATE AND WORK TO SOLVE A SINGLE PROBLEM.
- IT PROVIDES SERVICES SUCH AS HARDWARE, SOFTWARE, STORAGE, RESOURCES, ETC.	- IT HELPS TO SOLVE COMPUTATIONAL TASKS FASTER THAN USING A SINGLE COMPUTER AS IT TAKES A LOT OF TIME.

H HitechNectar

Cloud vs Distributed C...

 ResearchGate

GRID COMPUTING VS. ...

Functionality, Applications	Computer Clusters [16,20,30]	Peer-to-Peer Networks [34,40]	Distributed Computational Grids [6,15,31]	Cloud Platforms [1,8,11,12,30]
Architecture	Network of	Peer-to-peer	Heterogeneous	Virtualized cluster





Parameter	Grid Computing	Cloud computing
Goal	Collaborative sharing of resources	Use of service (eliminates the detail)
Computational focuses	Computationally intensive	Operations Standard and high-level instances
Level of abstraction	Low (more details)	High (eliminate details)
Degree of scalability	Normal	High
Multitask	Yes	Yes
Transparency	Low	High
Time to run	Not real-time	Real-time services

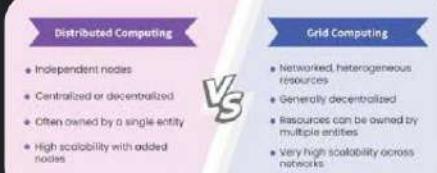
## Grid computing vs Cloud Computing | Download...

[Visit >](#)

Images may be subject to copyright. Learn More

[Share](#)
[Save](#)

Grid Computing	Cloud Computing
Means of utilization	Allocation of multiple servers onto a single task or job
Typical usage pattern	Typically used for job execution, i.e. the execution of a program for a limited time
Level of abstraction	Expose high level of detail
Task Size	Single Large
Computation/Service	Maximizes computing
Accessibility	Offers dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities.
Infrastructure	A decentralized system which spans across geographically distributed sites and lacks central control. It normally contains heterogeneous resources, such as hardware/software configurations, access interfaces and management policies.
Middleware	Self-tailored resilience with well-defined industry standards
Virtualization	Virtualization of data and computing resources



GeeksforGeeks

Distributed Computing...

Criteria	Grid Computing	Cloud Computing
User Management	Decentralized user agreement	Centralized user agreement
Dependency	Other computers perform the work	Users depend on a common server
Operation	Business exists in a complex network	Can also operate through the internet
Availability	Through Grid middleware	Through standard Web protocols
Scalability	Multiple domains	Single domain
Architecture	Distributed computing architecture	Decentralized architecture
Virtualization	Data subpartitioning resources	Hardware and software partitions
Configuration	Machine-to-machine	Decentralized
Application Type	Batch	Interactive

DataFlair

Grid Computing Vs Clo...

Feature	Grid computing	Cloud computing
Business model	Resources sharing between users	Resources deliver services for every users according to their demands
Performance	Low	High
Scalability	Normal	High (resources are consumed according to users' demands)
Interoperability	Open Grid Forum Standards	Web services (SOAP/ReST)
Availability	Low	High
User management	Decentralized	Centralized or might be outsourced



once  
share a  
system)

ness creates  
as you go

on clouds

vs members  
incapacitive ones]]||

i) Others types; Community / Federated

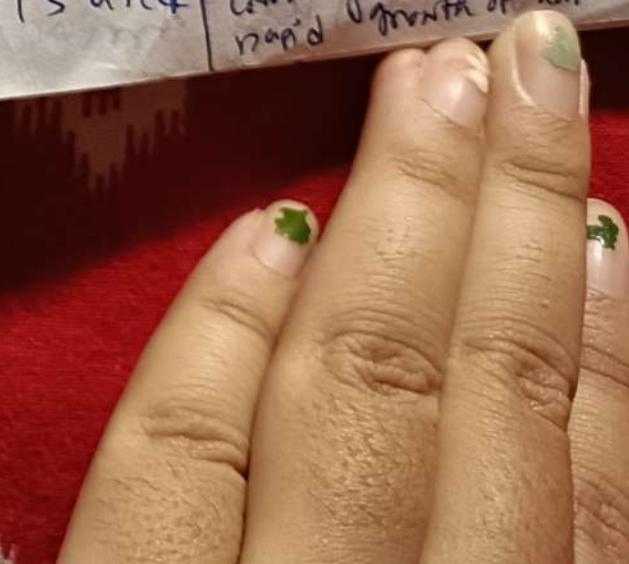
It's a c. infrastructure in which multiple organizations share resources & services based on common operational & regulatory requirements. It's a semi-public cloud. Eg - Heroku etc.

• Why C.C. is successful, when others paradigm have failed?

if it's in a better position to exploit recent advances in SW, hwning, storage & processing technologies promoted by the same companies who provide cloud services.

ii) Economical reasons - If it's used

SW requirement are  
coming due to the  
rapid growth of hardware



for enterprise computing ; its adoption by industrial organisations, government etc has a huge impact on the economy.

### ii) Infrastructure management reasons -

a) A single cloud consists of a mostly homogeneous set of hardware & s/n resources.

b) The resources are in a single administrative domain (AD).

### • Challenges of C.C

i) security risks - Data ~~has~~ breaches



what is para and full vi...



In cloud computing

In os

Geeksforgee

বা

En



Listen

In computer virtualization, **full virtualization** allows a guest operating system to run without modification in a virtual machine by emulating the underlying hardware. Paravirtualization, on the other hand, requires modifications to the guest OS kernel to communicate more efficiently with the hypervisor, often resulting in better performance.



Here's a more detailed comparison:

**Full Virtualization:**

**No OS Modification:**

Guest OS runs as is, without any



Home



Search



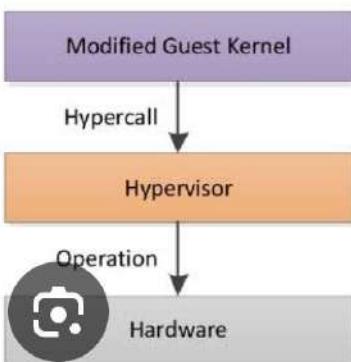
Saved



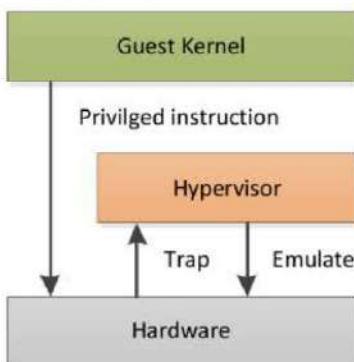
Notifications



## Para-virtualization



## “Classical” Full-virtualization



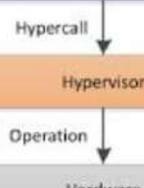
# Understanding Hypervisors: Exploring...

[Visit >](#)

Images may be subject to copyright. Learn More

[Share](#)[Save](#)

## Para-virtualization



## Full Virtualization

&lt;img alt="



ChatGPT ▾



A Service Level Agreement (SLA) in cloud computing is a promise between the cloud provider and the user about how well the service will work. It clearly states things like:

- How often the service will be available (like 99.9% uptime)
- How fast problems will be fixed
- What happens if the service doesn't meet these promises

It helps users know what to expect and what they'll get if things go wrong.



Ask anything



## 9.1.2 Storage Services

AWS provides a collection of services for data storage and information management. The core service in this area is represented by Amazon Simple Storage Service (S3). This is a distributed object store that allows users to store information in different. The core components of S3 are two: buckets and objects. Buckets represent virtual containers where to store objects, while objects represent the content that is actually stored. Objects can also be enriched with metadata that can be used to tag the content stored with additional information.

### 1. S3 Key Concepts

As the name suggest, S3 has been designed to provide a simple storage service accessible through a REST interface, which is quite similar to a distribute file system but that presents some important differences that allow the infrastructure to be highly efficient:

The storage is organized in a two level hierarchy. S3 organizes its storage space into buckets that cannot be further partitioned. This means that it is not possible to create directories or other kinds of physical groupings for objects stored in a bucket. Despite this, there are less limitations in naming objects, and this allows users to simulate directories and create logical groupings.

Objects stored cannot be manipulated like standard files. S3 has been designed to essentially provide storage for objects that will not change over time. Therefore, it does not allow renaming, modifying, or relocating an object. Once an object has been added to a bucket, its content and position is immutable, and the only way to change one of them is to remove the object from the store and add it again.

Content is not immediately available to users. The main design goal of S3 is to provide an eventually consistent data store. As a result, being a large distributed storage facility, changes are not immediately reflected. For instance, S3 uses replication to provide redundancy and efficiently serve objects across the globe; this introduces latencies when adding objects to the store—especially large ones—which are not available instantly across the entire globe.

Request will occasionally fail. Due to the large distributed infrastructure being managed, requests for object may occasionally fail. Under certain conditions, S3 can decide to drop a request by returning an internal server error. Therefore, it is expected to have a small failure rate during day-to-day operations, which generally it does not identify a persistent failure.

Access to S3 is provided with RESTful Web services. These express all the operations that can be performed against the storage in the form of HTTP requests (GET, PUT, DELETE, HEAD, and POST), which operate differently according to the element they address. As a rule of thumb, PUT/POST requests add new content to the store, GET/HEAD requests are used to retrieve content and information, while DELETE requests are used to remove elements or information attached to it.

**(a) Resource Naming.** Buckets, objects, and attached metadata are made accessible through a REST interface. Therefore, they are represented by *Uniform Resource Identifiers (URIs)* under the s3.amazonaws.com domain. All the operations are then performed by expressing the entity they are directed to in the form of a request for a URI.

Amazon offers three different ways of addressing a bucket:

**Canonical Form:** `http://s3.amazonaws.com/bucket_name/`. The bucket name is expressed as a path component of the domain name s3.amazonaws.com. This is the naming convention that has less restriction in terms of allowed characters, since all the characters that are allowed for a path component can be used.

**Subdomain form:** `http://bucketname.s3.amazonaws.com/`. Alternatively, it is also possible to reference a bucket as a subdomain of s3.amazonaws.com. In order to express a bucket name into this form, the name has to:

- be between 3 to 63 characters long;
- contain only letters, numbers, periods, and dashes;
- start with a letter or a number;
- contain at least one letter; and
- have no fragments between periods that start with a dash, end with a dash, or are an empty string.

This form is equivalent to the previous one when it can be used but it is the one to be preferred since it works more effectively for all the geographical locations serving resources stored in S3.

**Virtual hosting form:** `http://bucket-name.com/`. Amazon also allows referencing its resources with custom URLs. This is accomplished by entering a CNAME record into the DNS that points to the subdomain form of the bucket URL.

Since S3 is logically organized as flat data store, all the buckets are managed under the s3.amazonaws.com domain. Therefore, the names of buckets must be unique across all the users.

Objects are always referred as resources local to a given bucket. Therefore, they always appear as part of the resource component of a URI. Since bucket can be expressed in three different ways, objects indirectly inherit this flexibility:

- Canonical form: `http://s3.amazonaws.com/bucket_name/object_name`.
- Subdomain form: `http://bucket-name.s3.amazonaws.com/object_name`.
- Virtual hosting form: `http://bucket-name.com/object_name`.

Except for the '?', which separates the resource path of a URI from the set of parameters passed with the request, all the characters that follow the '/' after the bucket reference constitute the name of the object. For instance, path separator characters expressed as part of the object name do not have any meaning within the bucket store. Despite this, these can still be used to create logical



ChatGPT ▾



Amazon CloudWatch is a monitoring service from AWS that helps you track and understand what's happening in your cloud resources.

In simple words, CloudWatch watches your AWS services (like EC2, RDS, Lambda) and gives you:

- Metrics (e.g., CPU usage, memory, disk activity)
- Logs (detailed records of what your applications are doing)
- Alarms (alerts when something goes wrong or crosses a limit)

It helps you keep your apps running smoothly by spotting problems early.

Would you like an example of how CloudWatch is used?



Ask anything





ChatGPT ▾



Amazon EC2 (Elastic Compute Cloud) is a service from AWS that lets you run virtual computers (called instances) in the cloud.

In simple words, EC2 is like renting a computer on the internet. You can use it to:

- Run websites or apps
- Store files
- Process data
- Test software

You choose how powerful the computer is, how long you want it, and what software runs on it.

Would you like to know how to launch an EC2 instance?



Ask anything



Compute, storage, and middleware services constitute the core components of the Windows Azure platform. Beside these, other services and components simplify the development and the integration of applications with the Azure Cloud. An important area for these services is applications connectivity, including virtual networking and content delivery.

**(a) Windows Azure Virtual Network.** Networking services for applications are offered under the name of *Windows Azure Virtual Network*, which includes: *Windows Azure Connect* and *Windows Azure Traffic Manager*.

*Windows Azure Connect* allows easily setting up IP-based network connectivity among machines hosted on the private premises and the roles deployed on the Azure Cloud. This service is particularly useful in case of VM roles, where machines hosted in the Azure Cloud become part of the private network of the enterprise and can be managed with the same tools used in the private premises.

*Windows Azure Traffic Manager* provides load balancing features for services listening the HTTP or HTTPS ports and hosted on multiple roles. It allows developers to choose from three different load-balancing strategies: Performance, Round-Robin, and Failover.

Currently, the two services are still in beta phase and available for free only by invitation.

**(b) Windows Azure Content Delivery Network (CDN).** *Windows Azure CDN* is the content delivery network solution that improves the content delivery capabilities of *Windows Azure Storage* and several other Microsoft services such as *Microsoft Windows Update* and *Bing maps*. The services allows to serve Web objects (images, static HTML, css, and scripts) as well as streaming content by using a network of 24 locations distributed across the world.

### 9.3.2 SQL Azure

SQL Azure is a relational database service hosted on Windows Azure and built on the SQL Server technologies. The service extends the capabilities of SQL Server to the Cloud and provides developers with a scalable, highly available, and fault-tolerant relational database. SQL Azure is accessible from either the Windows Azure Cloud or any other location that has access to the Azure Cloud. It is fully compatible with the interface exposed by SQL server, so that applications built for SQL server can transparently migrate to SQL Azure. Moreover, the service is fully manageable by using REST API, allowing developers to control databases deployed in the Azure Cloud as well as the firewall rules set up for their accessibility.

Cloud Platforms in Industry

9.29

Fig. 9.4 shows the architecture of SQL Azure. Access to SQL Azure is based on the Tabular Data Stream (TDS) protocol, which is the communication protocol underlying all the different interfaces used by applications to connect to a SQL Server based installation such as ODBC and ADO.NET. On the SQL Azure side, access to data is mediated by the service layer, which provides provisioning, billing, and connection routing services. These services are logically part of server instances, which are managed by SQL Azure Fabric. This is the distributed database middleware that constitutes the infrastructure of SQL Azure and that is deployed on Microsoft data centers.

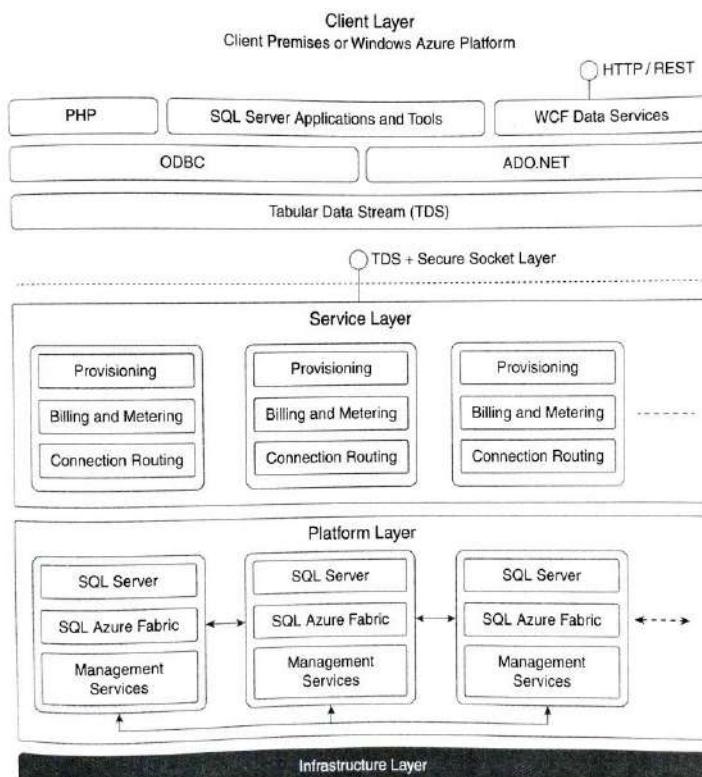


Fig. 9.4. SQL Azure Architecture.

Developers have to sign up for a Windows Azure account in order to use SQL Azure. Once the account is activated, they can either use the Windows Azure Management Portal or the REST APIs to create servers, logins, and configure access to servers. SQL Azure servers are abstractions that closely resemble physical SQL Servers: they have a fully qualified domain name under *database.windows.net* (i.e., *server-name*.

delete, and select a subset of entities. There are no constraints on the properties of entities and there is no facility for representing relationships among entities. Because of this, tables are more similar to spreadsheets rather than SQL tables.

The service is designed to handle large amounts of data and query returning huge result sets. Two main features provide support in this sense: partial result sets and table partitions. A partial result set is returned together with a continuation token allowing the client to resume the query for large result sets. Table partitions allow tables to be divided among several servers for load balancing purposes. A partition is identified by a key, which is represented by three of the columns of the table.

At the time of writing, a table can contain up to 100 TB of data and rows can have up to 255 properties, with a maximum of 1 MB for each row. The maximum dimension of row keys and partition keys is 1 KB.

**(c) Queues.** Queue storage allows applications to communicate by exchanging messages through durable queues, thus preventing messages from getting lost or remaining unprocessed. Applications enter messages into a queue and other applications can read them in a FIFO style.

In order to ensure that messages get processed, when an application reads a message, the message is marked as invisible, thus being not available to other clients. Once the application has completed processing the message, it needs to explicitly delete it from the queue. This two-phase process ensures that messages get processed before they are removed from the queue, and that client failures do not prevent messages from being processed. At the same time, this is also a reason why the queue does not enforce a strict FIFO model: messages that are read by applications that crash during the processing are made available again after a time out, during which other messages can be read by other clients. An alternative to reading a message is peeking, which allows retrieving the message but letting it be visible in the queue. Messages that are peeked are not considered processed.

All the services described are geo-replicated three times in order to ensure their availability in case of major disasters. Geo-replication involves the copy of data into a different data center, which is hundred thousands of miles away from the original datacenter.

### 3. Core Infrastructure: AppFabric

AppFabric is a comprehensive middleware for developing, deploying, and managing applications on the Cloud or for integrating existing applications with Cloud services. AppFabric implements an optimized infrastructure supporting scaling out and high availability; sandboxing and multi-tenancy; state management; and dynamic address resolution and routing. On top of this infrastructure, the middleware offers a collection of services that simplify many of the common tasks in a distributed application such as communication, authentication and authorization, and data access. These services are available through language agnostic interfaces, thus allowing developers to build heterogeneous applications.

**(a) Access Control.** AppFabric provides the capability of encoding access control to resources within Web applications and services into a set of rules, which are expressed outside the application code base. These rules give a great degree of flexibility in terms of the ability of securing components of the application and defining access control policies for users and groups.

Access control services also integrate several authentication providers into a single coherent identity management framework. Applications can leverage Active Directory, Windows Live, Google, Facebook and other services to authenticate users. This feature also allows easily building hybrid systems, which part existing in the private premises and part are deployed in the Cloud.

**(b) Service Bus.** Service Bus constitutes the messaging and connectivity infrastructure provided with AppFabric for building distributed and disconnected in the Azure Cloud and between the private premises and the Azure Cloud. Service Bus allows applications to interact with different protocols and patterns over a reliable communication channel that guarantees delivery.

The service is designed to allow transparent network traversal and to simplify the development of loosely coupled applications, without renouncing to security and reliability, and letting the developers focus on the logic of the interaction, rather than the details of its implementation. Service bus allows services to be available by simple URLs, which are untied to their deployment location. It is possible to support publish-subscribe models, full duplex communications point to point as well as in a peer-to-peer environment, unicast and multi-cast message delivery in one-way communications, and asynchronous messaging to decouple applications components.

Applications in order to leverage these features need to be connected to the bus, which provides these services. A connection is the element that is priced by Azure on a pay-as-you-go basis for service bus. Users are billed on connection/month basis and they can buy in advance "connection packs", which have a discounted price, if they can estimate in advance their needs.

**(c) Azure Cache.** Windows Azure provides a set of durable storage solutions that allow applications to persist their data. These solutions are based on disk storage, which might constitute a bottleneck for applications that need to gracefully scale along the clients requests and dataset size dimensions.

Azure Cache is a service that allows developers to quickly access data persisted on the Windows Azure storage or in SQL Azure. The service implements a distributed in-memory cache whose size can be dynamically adjusted by applications according to their needs. It is possible to store any .NET managed object as well as many common data formats (table, rows, XML, and binary data) and control its access to it by applications. Azure Cache is delivered as a service and easily integrates with applications, since there is no need to implement or deploy specific components. This is particularly true for ASP.NET applications, which already integrate providers for session state and page output caching based on Azure Cache.

The service is priced according to the size of cache allocated by applications per month, despite their effective use of the cache. At the time of writing, several cache sizes are available, ranging from 128 MB (\$45/month) to 4 GB (\$325/month).

### 4. Other Services

Compute, storage, and middleware services constitute the core components of the Windows Azure platform. Beside these, other services and components simplify the development and the integration of applications with the Azure Cloud. An important area for these services is applications connectivity, including virtual networking and content delivery.

Large	2 x 1.6 GHz	7 GB	1000 GB	High	\$0.48
Extra Large	4 x 1.6 GHz	14 GB	2040 GB	High	\$0.96
	8 x 1.6 GHz				

With respect to the worker role and web role, the VM role provides a finer control on the compute service and resource that are deployed on the Azure Cloud, at the same time it also requires more administrative effort for services configuration, installation, and management.

## 2. Storage Services

Compute resources are equipped with local storage in the form of a directory on the local file system that can be used to temporarily store information useful for the current execution cycle of a role. If the role is restarted and activated on a different physical machine, this information gets lost.

Windows Azure provides for different types of storage solutions, which complement compute services with a more durable and redundant option if compared to local storage. Differently from local storage, these services can be accessed by multiple clients at the same time and from everywhere, thus becoming a general solution for storage.

**(a) Blobs.** Azure allows storing large amount of data in the form of Binary Large Objects (BLOBs) by means of the *blobs* service. This service is optimal to store large text or binary files. Two types of blobs are available:

**Block Blobs.** Block blobs are composed by blocks and they are optimized for sequential access and therefore they are appropriate for media streaming. At the time of writing (during 2011-2012), blocks are of 4MB and a single block blob can reach 200 GB of dimension.

**Page Blobs.** Page blobs are made of pages that are identified by an offset from the beginning of the blob. A page blob can be split in multiple pages or constituted by a single page. This type of blob is optimized for random access and can be used to host data different from streaming. At the time of writing, the maximum dimension of a page blob is 1 TB.

Blobs storage provides users with the ability to describe the data by adding metadata. It is also possible to take snapshots of a blob for backup purposes. Moreover, in order to optimize its distribution, blobs storage can leverage the Windows Azure CDN, so that blobs are kept close to those requesting them and can be served more efficiently.

**(b) Azure Drive.** Page blobs can be used to store an entire file system in the form of a single Virtual Hard Drive (VHD) file. This can then be mounted as a part of the NTFS file system by Azure compute resources, thus providing persisting and durable storage. A page blob mounted as part of an NTFS tree is called an *Azure Drive*.

**(c) Tables.** Tables constitute a semi-structured storage solution, allowing users to store information in the form of entities having a collection of properties. Entities are stored as rows into the table and identified by a key, which also constitutes the unique index built for the table. Users can insert, update, delete, and select a subset of the rows stored in the table. Unlike SQL tables, there is no schema enforcing constraints on the properties of entities and there is no facility for representing relationships among entities. Because of this, tables are more similar to spreadsheets rather than SQL tables.

The service is designed to handle large amounts of data and query returning huge result sets. Two main features provide support in this sense: partial result sets and table partitions. A partial result set is returned together with a continuation token allowing the client to resume the query for large result sets. Table partitions allow tables to be divided among several servers for load balancing purposes. A partition is identified by a key, which is represented by three of the columns of the table.

At the time of writing, a table can contain up to 100 TB of data and rows can have up to 255 properties, with a maximum of 1 MB for each row. The maximum dimension of row keys and partition keys is 1 KB.

**(c) Queues.** Queue storage allows applications to communicate by exchanging messages through durable queues, thus preventing messages from getting lost or remaining unprocessed. Applications enter messages into a queue and other applications can read them in a FIFO style.

In order to ensure that messages get processed, when an application reads a message, the message is marked as invisible, thus being not available to other clients. Once the application has completed processing the message, it needs to explicitly delete it from the queue. This two-phase process ensures that messages get processed before they are removed from the queue, and that client failures do not prevent messages from being processed. At the same time, this is also a reason why the queue does not enforce a strict FIFO model: messages that are read by applications that crash during the processing are made available again after a time out, during which other messages can be read by other clients. An alternative to reading a message is peeking, which allows retrieving the message but letting it be visible in the queue. Messages that are peeked are not considered processed.

All the services described are geo-replicated three times in order to ensure their availability in case of major disasters. Geo-replication involves the copy of data into a different data center, which is hundred thousands of miles away from the original datacenter.

## 3. Core Infrastructure: AppFabric

AppFabric is a comprehensive middleware for developing, deploying, and managing applications on the Cloud or for integrating existing applications with Cloud services. AppFabric implements an optimized infrastructure supporting scaling out and high availability; sandboxing and multi-tenancy; state management; and dynamic address resolution and routing. On top of this infrastructure, the middleware offers a collection of services that simplify many of the common tasks in a distributed application such as communication, authentication and authorization, and data access. These services are available through language agnostic interfaces, thus allowing developers to build heterogeneous applications.

**(a) Access Control.** AppFabric provides the capability of encoding access control to resources.