Summary

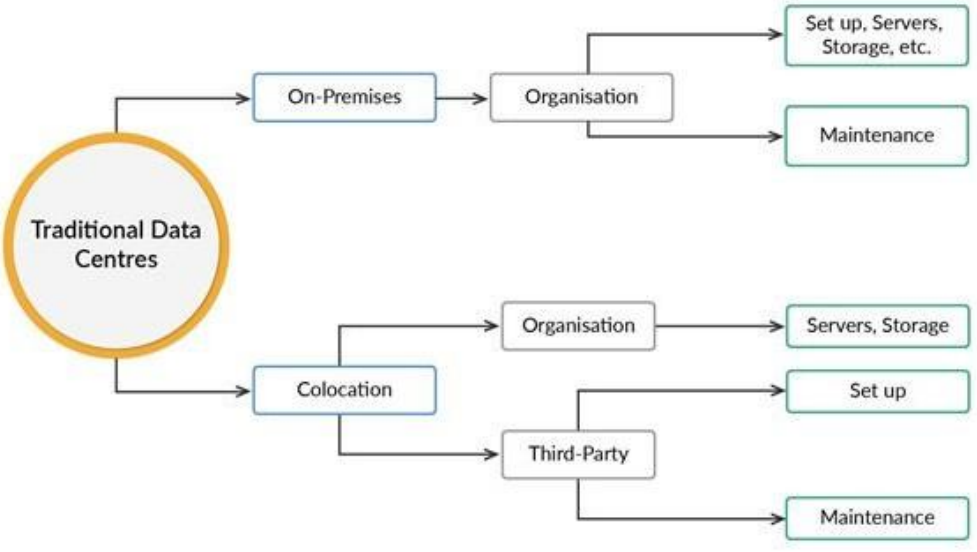
Why Cloud?

In this session, you learnt the procedure for storage and computing followed before the origin of Cloud computing. You also gained an understanding of the evolution of Cloud and explored the essential characteristics and benefits of Cloud computing.

Traditional Computing vs Cloud

Traditional data centres are deployed in two ways, which are as follows:

1. **On-premises**:
    a. In this set-up, the organisation establishes and maintains the data centre by itself. It also owns all the equipment.
    b. In some cases, networking between the data centre and the organisation's office is taken care of by the network service provider.
2. **Colocation**:
    a. In this set-up, the organisation ties up with a third-party firm to set up and maintain its data centres.
    b. The organisation provides the required computer servers, storage and networking, whereas the third-party firm provides the power, cooling and physical security.
    c. In some cases, networking between the data centre and the organisation's office is taken care of by the network service provider.



Disadvantages of traditional data centres:

1. Data centres require **high-cost investments,** which include power backup, telecommunication system, cooling system and racks.
2. Maintaining a high-density data centre could cost millions of dollars per year in **maintenance** alone. The major cost expenditure here would involve electricity cost, labour cost, real-estate cost and other safety costs.
3. A traditional data centre has **limited space**, and every time more servers are required, you need to get additional space to build the infrastructure and install the required set-up. This increases the investment cost; besides, having more servers requires more cooling, leading to an overall increase in cost. Procurement of new hardware usually requires a lead time of 1–2 months and is subject to multiple stages of approval, as these are significant costs for an organisation.
4. **Upgradation**, patching and scaling up of resources could add to the final cost and continue to increase the capital cost. Also, increasing the capacity of these data centres is usually a cumbersome task and can take weeks or months to realise.

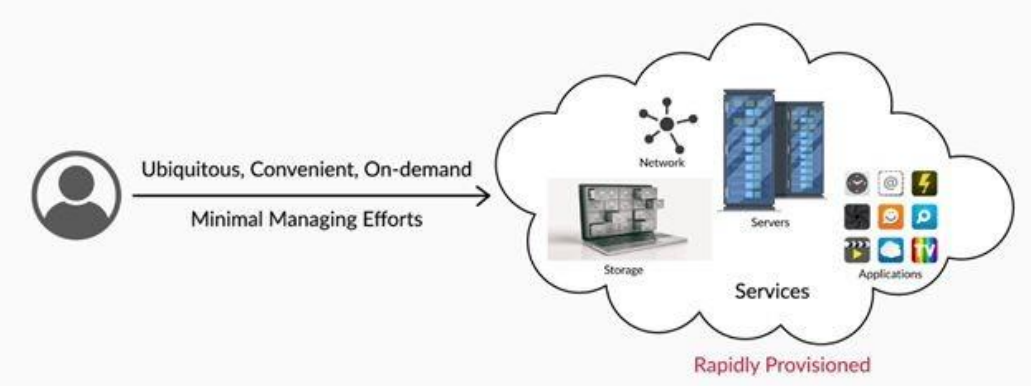
Evolution of Cloud computing:

In the early 2000s, with the Internet boom, the demand for computing resources was rising, which was becoming a major problem. The earliest breakthrough in Cloud computing as we know today happened in the late 2000s. In 2006, Amazon launched the Amazon Web Services, and in 2008, Google released the beta version of the Google App Engine. In 2010, NASA along with Rackspace introduced OpenStack, an open-source-based initiative for the development of a Cloud framework.

What is Cloud computing?

Definition of Cloud computing (NIST - National Institute of Standards and Technology):

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



The key aspect of a Cloud computing model is its ability to **provide on-demand resources**, which include servers, applications, network configurations and services, with minimal effort or interaction with any service provider. These systems get created rapidly and are usually part of a larger shared pool of resources.

The entity that provides such an ability to the end-user for a cost is called a **cloud service provider**.

Benefits of using Cloud over traditional data centres:

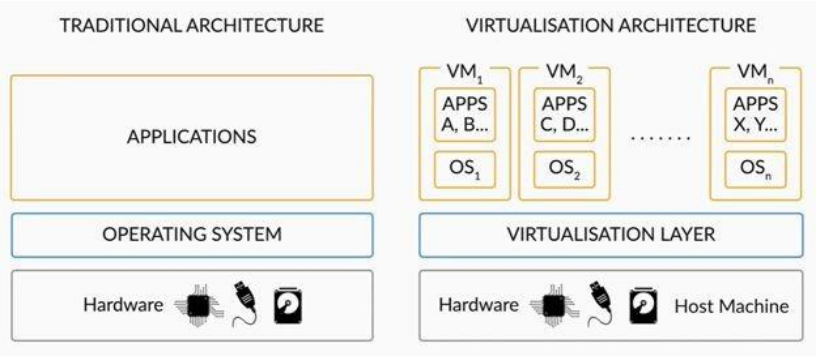
1. Cloud has the ability to serve on-demand resources such as servers, applications, network configurations and services. Users can provision these resources when required with just a single click.
2. Users can instantaneously (manual or automatic way) increase or decrease the number of resources that they require from the Cloud.

Cloud services, in general, can refer to any IT service that can be **accessed from a Cloud service provider over a network**. For example, storage services for storing files, services that provide fully managed databases and services that provide computing machines on demand can be called cloud services.

Virtualisation

Virtualisation can be defined as the process of **running a virtual instance** of a computer system in a **layer abstracted from the actual hardware**. Usually, only one operating system can be run on one hardware stack comprising CPU/network and memory. On the other hand, using virtualisation, you can **run multiple operating systems** on the same underlying hardware, making it easy to optimise the hardware usage. These multiple systems running on the same hardware are called virtual machines.

The architecture that is followed in building typical laptops is traditional architecture, whereas virtualisation architecture is used by some Cloud service providers to run multiple virtual machines on a single hardware.

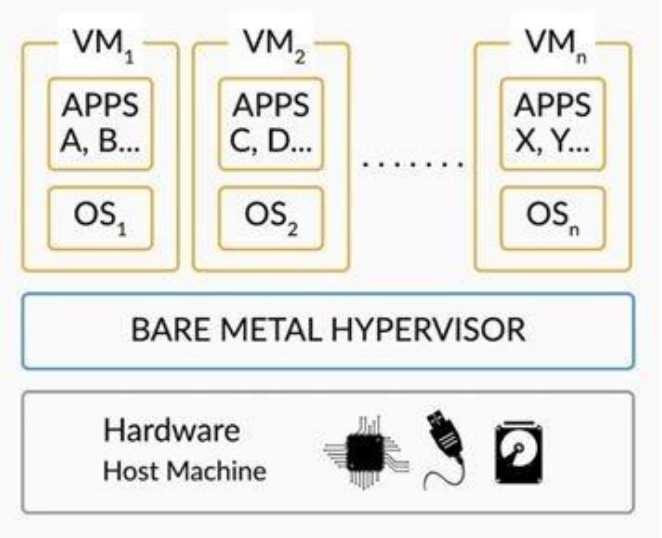


On the diagram on the right, you can see virtualisation architecture. Here, you have the hardware layer, which is a combination of CPU, physical memory and networking. Above this layer, you can see the virtualisation layer. This layer enables the decoupling of the hardware and the operating system, thereby removing the one-to-one mapping between the operating system and hardware. A system on which a virtualisation layer runs one or more virtual machines is called a **host machine**. Each of these **virtual machines is called a guest machine** and is running a different type of operating system, and different applications are running on top of each of the operating systems.

One of the key components of the virtualisation layer is the hypervisor. **Hypervisor** is the software that **manages, creates and deletes virtual machines** along with the memory and resource management of the hardware.

There are two types of hypervisors, namely, bare metal or Type-1 hypervisors, and hosted or type-2 hypervisors.

1. **Bare-metal hypervisors** run directly on the **host machine's hardware**, without the need for an underlying operating system. This means that the hypervisor has direct access to the hardware. Bare-metal hypervisors enable **efficient memory management** and are effective in creating and running servers that closely mimic physical hardware servers.



   a. A bare-metal hypervisor resides above the hardware and enables running multiple virtual machines on it. It allocates the ideal memory storage and bandwidth to each virtual machine.

   b. For example, an application such as Google Chrome would have to be tested on multiple operating systems such as Windows and Linux, as the browser runs on actual customer desktops and machines. Rather than bringing up a physical server for each environment, a

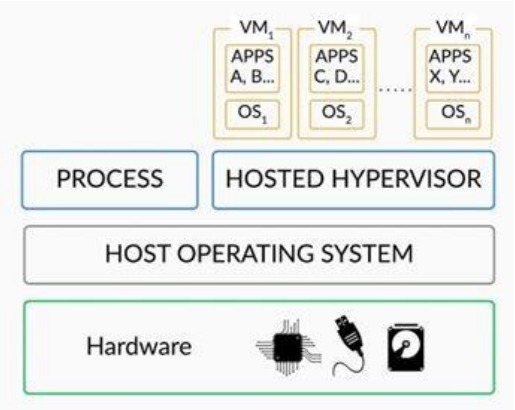
bare-metal hypervisor-based virtualisation would help in creating various operating systems and then bringing them down once the application has been tested. More importantly, it would also make these testing environment configurations close to real customer environments, as the configuration of a bare-metal hypervisor is limited only by the underlying hardware capabilities.

c. Some examples of bare-metal hypervisors are VMware's ESXi, Microsoft's Hyper-V and Xen hypervisors.

d. Bare-metal hypervisors are hard to set up and require dedicated hardware to run. In some cases, it is better to run the hypervisor as a simple application on an existing operating system, which makes bringing up the required OS faster. This is where type-2 or hosted hypervisors come into the picture.

2. **Type-2 or hosted hypervisors** run as a software using an operating system such as Windows or Linux. Also, they use the resources of the operating system to access the hardware.

   a. Hosted hypervisor systems are limited by the resources of the operating system, as the virtual machines created on the hosted hypervisor cannot access the hardware except through the operating system. Since a hosted hypervisor runs on a host operating system, it enables the host operating system user to switch between multiple virtual machines seamlessly.



   b. Suppose you suspect that a file that you want to open is infected. To tackle this problem, instead of opening the file in your host OS and potentially exposing yourself to security threats, you could install a hosted hypervisor on your OS, bring up a virtual machine within that hypervisor, open the file and examine it safely.

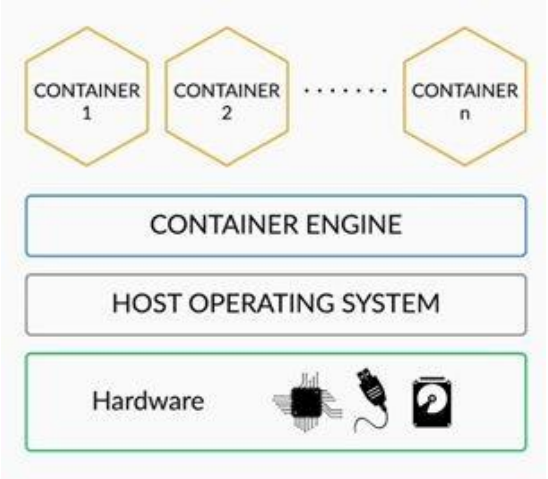
   c. Some examples of hosted hypervisors are Oracle Virtual Box and VMWare Workstation.

Bare-Metal Hypervisor (Type-1)	Hosted Hypervisor (Type-2)
It resides directly on the hardware.	It interacts with the hardware through the host operating system.
It is faster, as it directly interacts with the hardware.	It is a bit slower when compared with the bare-metal hypervisor, as it interacts with the hardware through the host operating system.

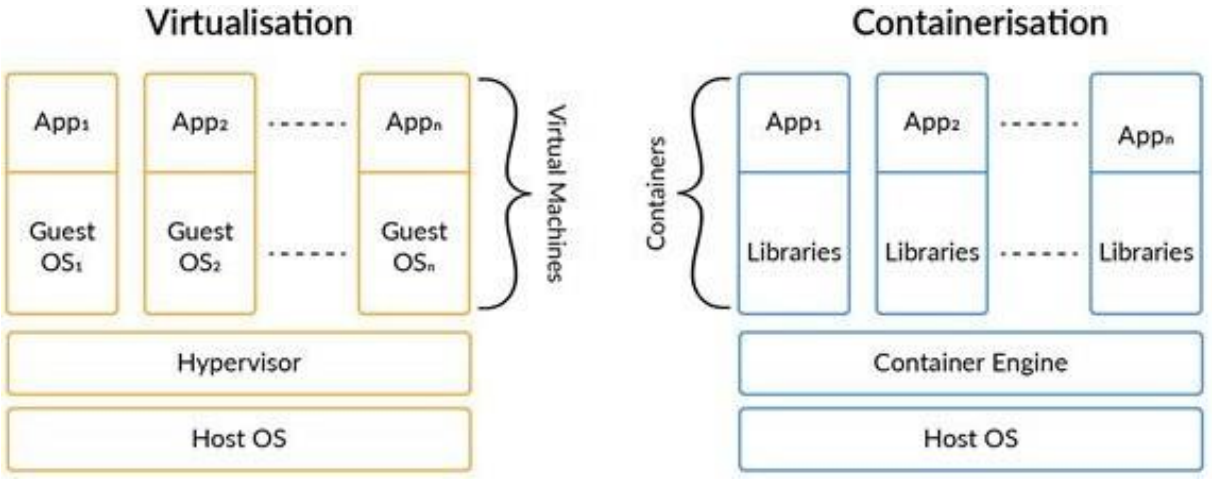
Containerisation

Consider the services of a food delivery app written in Java. A web server is one of the ways to host this web application. To host a single application in a virtual machine, you first need to load all the OS components and then install the web server on which the application is hosted, even though the web server might not need all the OS components that were loaded.

On the other hand, a container shares the OS kernel with just the essential libraries, where we can simply install the web server on top of the container. This makes it lightweight and also enables running multiple containers on the same operating system. This also makes it easier to create a new container in case one web server container goes down. You can see the container engine in the containerisation architecture; **it runs on the host operating system and shares the OS kernel among the containers. These containers are isolated from each other.**



Virtualisation	Containerisation
Using virtualisation, you can deploy applications on multiple virtual machines. Though you require only a few specific resources to run the application, you need to load the entire operating system in a virtual machine and then deploy the application.	Using containerisation, you can deploy the application in containers, where you will be provided with only those resources that are required for running it. This is faster than creating a virtual machine and deploying an application.



1. Overall, containerisation provides **dedicated isolated environments** within the operating system to help run the applications, without requiring an entire virtual machine to run each application.
2. Since containers do not contain a full OS layer, they are much **smaller in size**, and you can create a container within an operating system faster than a virtual machine.
3. From the perspective of a Cloud service provider, this would **help scale the system up and down faster** with much higher elasticity than scaling virtual machines. Containers also provide the flexibility to run a variety of operating systems on a single OS, thus improving the performance. This is the reason why many of the latest Cloud services are built using containers, as they scale faster and are lighter.

Essential Characteristics of Cloud

Why is cloud computing a better solution when compared with the infrastructure of traditional data centres?

1. **Cloud computing is cost-efficient**.

One of the prohibitive costs of traditional data centre infrastructure is the large upfront investment costs, where the infrastructure may or may not be used completely. On the other hand, cloud service providers usually offer flexible cost options such as pay as you go, steep discounts on upfront payments, post-usage payments, etc., which not only helps reduce the upfront investment costs but also works well with changing project requirements. With this service, resources such as storage, processing and bandwidth, etc. can be managed and controlled by the user. This enables the transparency of usage between the cloud provider and the user.

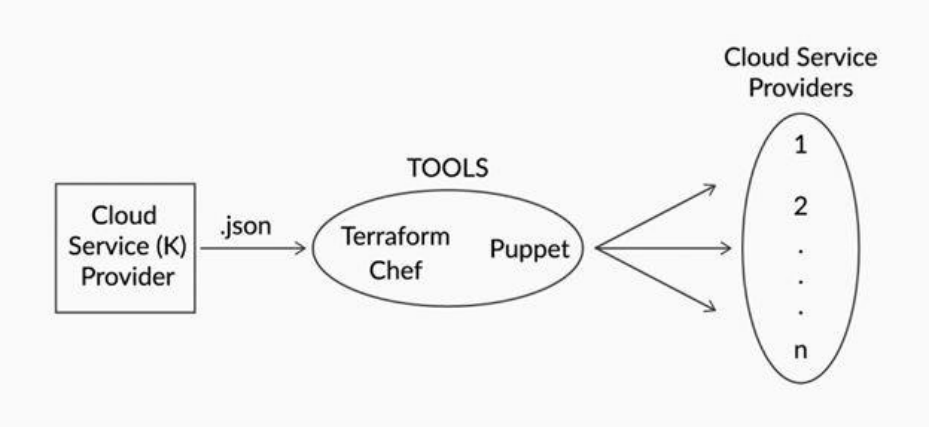
2. **Cloud computing provides scalability and rapid elasticity.**

In cloud computing, scalability refers to the ability to increase or decrease the capacity of the computing system based on the incoming demand, whereas elasticity is a measure of how quickly a system can be scaled up or down to closely match the incoming demand. With the underlying virtualisation and containerisation techniques, Cloud can help you scale up your application to a thousand instances easily during special days, such as sale day for an e-commerce website, and also scale down rapidly, so that you bear the cost of the additional servers only when they are needed.

Most cloud service providers offer an in-built mechanism called **autoscaling,** with which you can scale automatically based on system parameters such as website traffic or network load or increase in CPU load. It also lets you bring down machines when there is no longer a need for them due to reduced website traffic or decreased CPU utilisation below a certain threshold.

3. **Cloud computing has ushered in a better way of creating and managing infrastructure using Infrastructure as a Code.**

Infrastructure as a Code (IaC) is the process of managing and provisioning cloud resources through definition files such as JSON files. Once the JSON file is populated with the required infrastructure requirements, it can be fed into tools such as Terraform, Chef and Puppet that are connected to the cloud service provider to create and remove infrastructure. This helps to track changes in the infrastructure, destroy and recreate the infrastructure components easily with minimal effort. As many of the Infrastructure as a Code tools support multiple cloud services and operating systems, it is easy to port the definition file written for one cloud service provider to work seamlessly with another, thus simplifying the process of switching between multiple service providers.

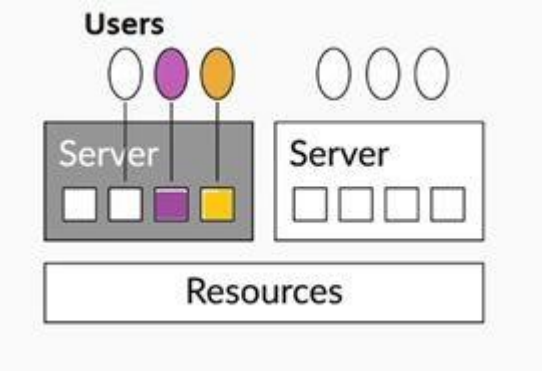


4. **Cloud allows multi-tenancy**

In a traditional data centre system, one data centre belongs to a single organisation/customer. On the other hand, cloud computing offers a multi-tenant cloud system.

A multi-tenant cloud is a cloud computing architecture that allows multiple consumers to share the same set of hardware resources through resource pooling. Resources are assigned and reassigned dynamically with the end user remaining completely unaware of the actual location of the resources. In a multi-tenant cloud system, the cloud providers usually allow a user to select a location (region) and the exact data centre in that location. However, while requesting a resource, the user remains completely unaware of the exact physical device behind that resource.

Each consumer's resources are kept isolated, with no scope of interaction with the resources of other consumers, even if they reside in the same physical server.



For example, a company storing financial data could run its applications alongside the applications of another company that deals with gaming software on the same underlying infrastructure. The resources of the two companies could still remain isolated, with no scope of accessing each other's data.

Cloud computing also helps in hosting applications in a variety of geographical regions without requiring physical presence in each of these regions. For instance, some of the data restrictions in the European Union necessitate web applications and user data to remain within the European Union. Prior to cloud computing, this would have been a cumbersome process, as a company based in the USA would have had to go through the painstaking process of setting up data centres in every such geographical location; however, this can now be done by a cloud-based approach.

Usually, most cloud providers support multiple regions across multiple physical data centres around the world, resulting in easy replication of resources across the world. While creating a cloud resource, the cloud service provider allows you to specify the region where it should be located. Hence, in this case, an end user attempting to create a web application for the European Union just needs to create the required resources, such as database and web servers, by selecting Europe as their cloud region. Once this is done, all the data stored in that database would reside on an actual physical server located in Europe, thus being compliant with local laws.

5. **Cloud computing resources have broad network access**

Cloud service providers' web consoles are usually accessible via the Internet through a wide array of clients such as mobile and web applications. The resources thus created can also be accessed from multiple clients as long as they adhere to the security restrictions of the end user.

6. **Cloud computing supports on-demand self service**

Almost all of the cloud service providers help in creating a computing resource that just involves interacting with a web console or interacting programmatically through API calls. There is no need to go through a cumbersome approval process or wait for assistance of a support personnel. If you want to start a new machine in the cloud and require the storage, then you need not ask anyone to grant the storage. You can simply create an account with any of the cloud service providers and then start the machine and purchase the storage on your own with just a few clicks.

Summary

Deployment and Service Models

In this session, you understood the different cloud deployment models that will help you deploy your applications on the cloud and also learnt about some of the popular cloud service providers. You also gained an understanding of cloud service models.

Cloud Deployment Models

Depending on how and where the cloud computing resources are deployed, there are four main types of cloud deployment models:

1. Public cloud
2. Private cloud
3. Hybrid cloud
4. Community cloud

Public cloud

A public cloud platform offers services via a third party to all its users by allowing them to access the resources over the Internet. The cloud resources and the backend hardware are completely owned by a cloud service provider, and the services are delivered over the network. The public cloud is usually multi-tenant. Most widely used public web applications use public cloud.

Advantages:

1. From a pricing perspective, public cloud would be the most affordable, as it could take full advantage of the cloud computing technology, including common resource pooling.
2. It provides almost unlimited scalability, along with high reliability.
3. It is location-independent, meaning you can access the public cloud from any location via the Internet.

Disadvantages:

1. If not secured properly, the public cloud is vulnerable to malicious attacks. Since it is hosted on multi-tenant data centers, a malicious or ignorant user might expose the hardware to security threats. If one data centre goes down, it might affect multiple users who are dependent on that data centre.

2. The flexibility of the users to customise the public cloud as per their requirements is limited.

Private cloud

In a private cloud, the resources are provisioned to only an authorised user. Here, the user can be an organisation with multiple consumers. The **user can own and manage the resources, or a third party** can manage them for the user, or any combination of these is possible. It can exist on or off premises of the organisation.

1. This kind of cloud platform is **commonly used in the organisations** that maintain their own infrastructure of servers, storage devices and networking devices. **OpenStack** is a free and open source cloud computing software platform that organisations often use for creating a private cloud using their own infrastructure.

2. With a private cloud, the organisation either sets up its **own data centre or ties up with a vendor to set up the cloud for it**. For example, the US Department of Defense uses a private cloud, which is built, managed and owned by AWS.

Advantages:

1. It provides **high security** and also restricts access to only authorised users. Hence, this kind of infrastructure is generally preferred in financial institutions such as **banks and insurance firms.**

2. It provides **high control over the resources**.

Disadvantages:

1. It is **not cost-effective** when compared with a public cloud.
2. It has **limited scalability** and can be scaled only up to the internal hosted resources.

How is a private cloud different from an on-premise data centre?

1. An **on-premise data centre does not fulfil the cloud characteristics**. An on premise data center has the **same disadvantages as any traditional data center**, along with not having any key characteristics of a cloud computing model such as on-demand service, rapid elasticity, resource pooling and broad network access.
2. Certain industries require a high level of uptime and strong security, and at times, these industries may also have to adhere to regulatory compliance, which mandates them to maintain the resources on premises. A private cloud would be an ideal solution for such industries
3. Using a private cloud is more expensive than using a public cloud. The scalability of a private cloud is limited by the internal infrastructure. In other words, a private cloud provides more security and flexibility at a higher cost.

Community cloud platform

A community cloud platform offers services to **a group of organisations** that belong to the **same community or geographical area to access the resources.**

Examples of use cases

1. Healthcare

A community cloud can be used by industries that need to perform specific activities on their cloud set-ups to be compliant with various regulatory requirements. For example, healthcare industries need to be compliant with regulatory requirements such as HIPAA (Health Insurance Portability and Accountability Act). HIPAA mandates strict security standards for healthcare-related data. Since public clouds do not provide this level of security, each company in the healthcare domain would have to invest in a private cloud to adhere to the requirements. This could lead to higher costs for each of the organisations. Instead, if all the healthcare companies form a community and create a cloud that implements the regulatory requirements, then the investment for each company will be reduced.

2. Financial institutions

In a similar manner, financial institutions may require a cloud with ultra-low latency so that stock trading could be done effectively and fast. Hence, when there are common restrictive cloud requirements, the community cloud model can be used. It is more flexible than a public cloud but cheaper than a private cloud.

Advantages:

1. The cost of maintenance can be shared among the organisations in the community.
2. It is more secure than a public cloud and less expensive than a private cloud.

Disadvantages:

1. It is difficult to distribute the responsibilities among the organisations in a community.
2. It is difficult to segregate the data among the organisations in a community.

Hybrid cloud

A hybrid cloud is a **combination of a public cloud and a private one**. It is composed of two or more distinct cloud infrastructures (private, public or community), which communicate among themselves to form a hybrid cloud.

Advantages:

1. A private cloud is **secure**, and hence, a hybrid cloud is secure as well.
2. **Scalability**: A public cloud is scalable. Therefore, a hybrid cloud, which is the combination of public and private clouds, is also scalable.
3. Users can access both the private and the public clouds as per their requirements; thus, a hybrid cloud offers **flexibility**.
4. Public cloud is **cost-effective**; hence, a hybrid cloud is also cost-effective if the user wants to use the public cloud properties.
5. A hybrid cloud can also be used to enable **'cloud bursting'**. It is a technique that is used when an organisation in a private cloud can use resources from the public cloud when they have reached their peak capacity. Such a situation can arise during special sale days; then, an organisation can use the public cloud for managing the additional resource requirements.

Disadvantages:

1. **Complex networking problems**: Due to the complexity of having the public and the private cloud, there would be an issue in configuring the network.
2. **Organisation's security compliance**: Both the public and the private clouds should comply with the organisation's security norms, and it is not easy to set up the clouds to meet this requirement.

Cloud service providers

Currently, there are about 250 cloud service providers in the market. Of these, Amazon Web Services, Microsoft Azure and Google Cloud together account for almost 65% of the market share.

1. Amazon Web Services

   Amazon Web Services was started in 2006 and was one of the first cloud service providers. As of 2019, it offers 149 services, spanning 22 geographic regions across the world. Some of the most popular AWS services include Amazon S3, Amazon EC2, Amazon RDS,Amazon Kinesis and Amazon Lambda.

2. Microsoft Azure

   The next popular cloud service provider is Microsoft Azure. It was launched in 2010. As of 2019, it offers over 100 services. More than 95% of Fortune 500 companies use Azure. Some of the top Azure services include Azure App Service, Azure Web Apps, Azure Functions and Azure Cosmos db.

3. Google Cloud

   Another popular cloud is Google Cloud. It started its beta testing in 2008 and started offering full services in 2011. Like AWS and Azure, Google Cloud also provides an array of services in the fields of big data, machine learning, networking and computing. It offers over 90 services across these areas.

Other noteworthy players in the field are Digital Ocean, Heroku, IBM Cloud, Alibaba Cloud and Oracle Cloud.

Cloud Service Models

Today, cloud service providers offer fully managed services for a wide variety of technologies, ranging from blockchain, caching, message queues, SQL databases, and NoSQL databases to large data warehouse solutions. They also offer ready-to-use services such as facial recognition APIs, object recognition APIs, text-to-speech APIs and frameworks to perform machine learning activities.

This wide array of services can be categorised into three standard cloud service models, namely:

1. Infrastructure as a service (IaaS)
2. Platform as a service (PaaS)
3. Software as a Service (SaaS)

Infrastructure as a Service (IaaS)

Deploying an application using this service model is similar to deploying applications in a data centre, the only difference being that the hardware maintenance part is handled by the cloud service provider in the case of the former. This model provides only the fundamental building blocks of infrastructure such as virtual machines. The onus of installing all necessary updates, software and patches required by the application is on the user of the cloud service.

Among all the service models, **IaaS offers the most flexibility**. The virtual machines built on this model can be either **multi-tenant or bare-metal instances with dedicated hardware**. These machines can come with a variety of combinations of CPU cores, RAM and memory.

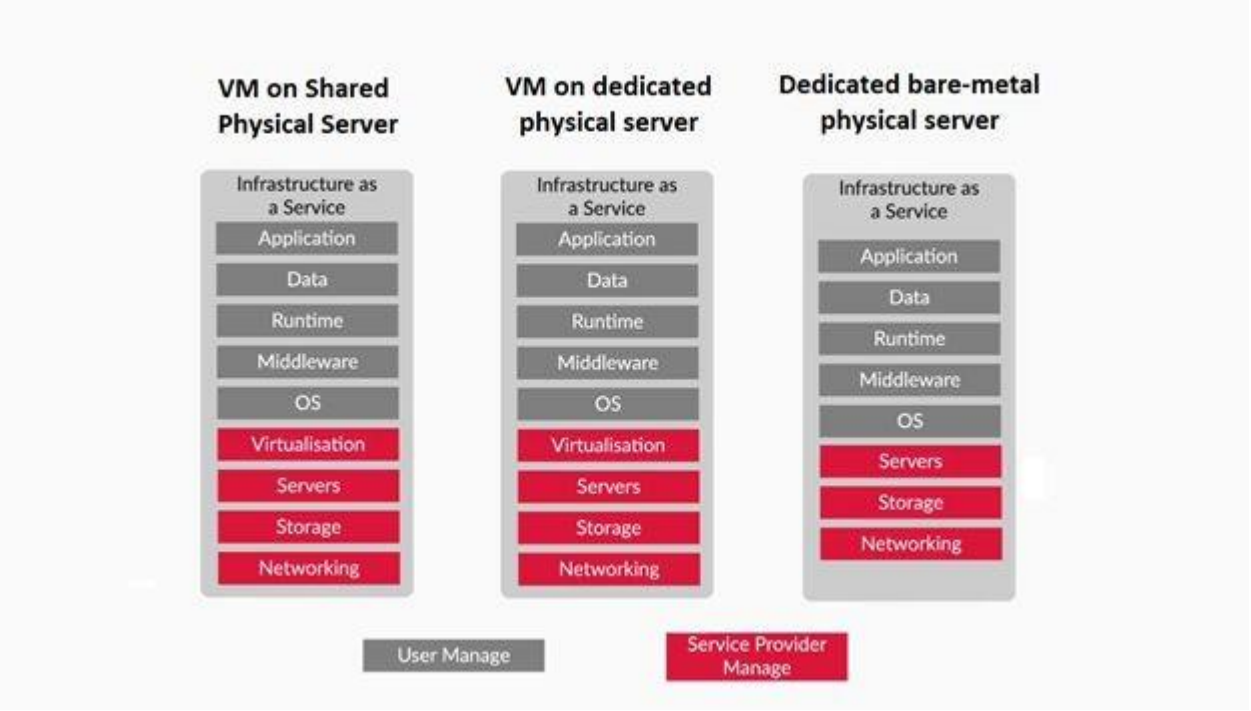
1. IaaS is best used for activities that require custom modification of the server by installing specialised software or using specific hardware variants to run your application.

2. There are three types of IaaS, namely:

    a. Virtual machines on a shared physical server
        - In the case of a virtual machine on a shared physical server, different users will have their virtual machines on the same shared physical storage. This is the most affordable and common of all types of IaaS.
        - In the IaaS model, the service provider manages the networking, storage, servers, and virtualisation, while the user would be responsible for the rest.

    b. Virtual machines on a dedicated physical server
        - In the case of virtual machines on a dedicated physical server, the components of the data centre stack that are managed by the service provider and the user are the same as in the case of virtual machines on a shared physical server. The only difference here is that the physical server on which the virtual machine is hosted is dedicated to the user.
        - This is a bit costlier when compared with the virtual machines on a shared physical server.

    c. Dedicated bare-metal physical server
        - Here, the user gets direct access to the physical server; no hypervisor or virtual machine is installed on this physical server.
        - In a data centre stack, there would be no virtualisation part, but the service provider would be responsible for the networking, storage and servers, while the users would be responsible for the rest.
        - This is the most expensive of all types of IaaS and is offered by very few cloud service providers.
        - Bare-metal physical servers are suited for applications that require direct access to the hardware feature set and for running applications the licenses of which restrict them from working only on physical servers. Certain applications that require finer levels of optimisation of performance, such as training complex machine learning algorithms, benefit from such bare-metal servers.



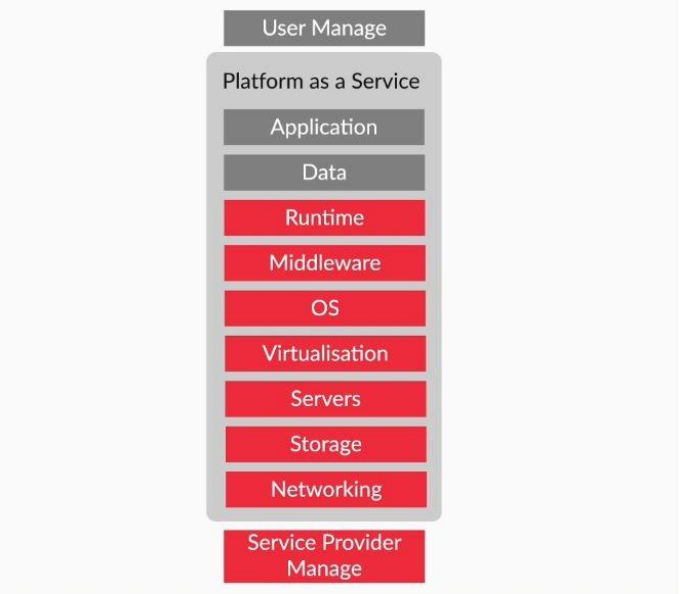
PaaS and SaaS

Platform as a Service (PaaS)

In the PaaS model, the service provider manages the required hardware, software and custom environment, while the user is responsible for the rest. Thus, Platform as a Service relieves the organisation of the additional burden of maintaining the software and installing patches and updates. So, it allows them to focus just on the deployment and coding of their applications.

1. Platform as a Service is a great option when you are using standard technologies to develop your applications. For example, most cloud service providers now offer application management services, which simplifies the process of deploying web applications.

2. The end user simply uploads the code to the app engine service, which takes care of creating multiple virtual machines and deploying the code to them. Most cloud service providers today have managed database services of industry-leading database engines such as mySQL, SQL server and PostGreSQL.



Advantages:

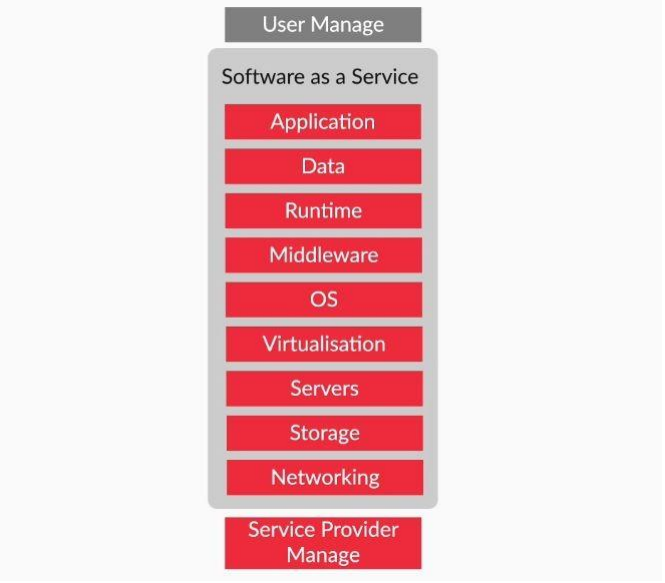
1. Prebuilt platform: PaaS provides an already built platform for users to build and run their applications.
2. User-friendly: It is a simple model to use and deploy applications.
3. Low cost: Since the platform is already built, the user needs to create only their applications. This reduces the costs related to hardware and software.

Disadvantages:

1. Migration issues: Migrating the user applications from one PaaS vendor to another might raise some issues.
2. Platform restrictions: The platforms provided by some vendors might have certain restrictions, for instance, the user can use only certain specified languages.

Software as a Service (SaaS)

The cloud service provider manages the required hardware, software and custom environment, including the application. The user's job is simply to have an Internet connection to use those applications.



Advantages:

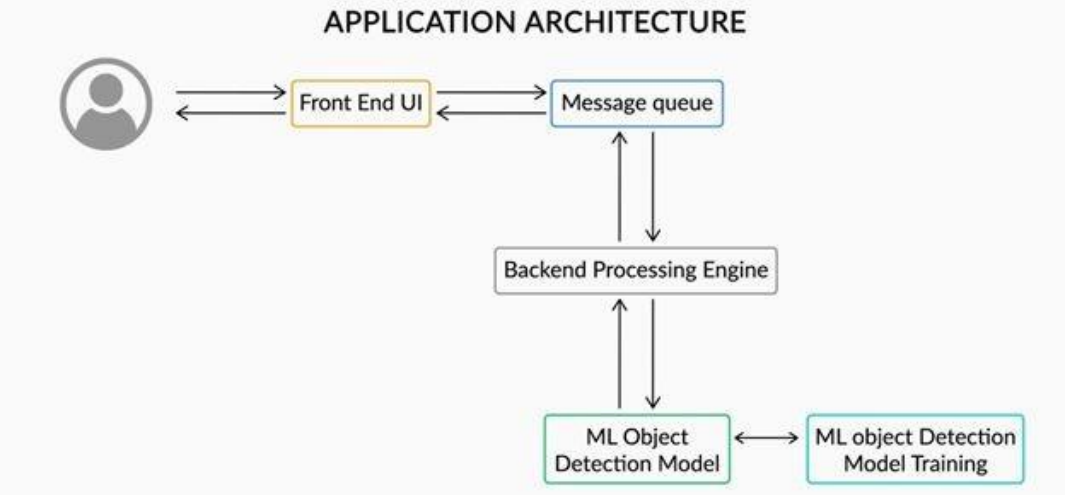
1. Ease of access: Users can access the applications on the server from anywhere using any Internet-connected device. Most types of internet-connected devices can access SaaS applications.
2. Low maintenance: Users need not update an application. The application is on the server, and it is the service provider's responsibility to maintain the application.
3. Quick set-up: Users do not require any hardware to install the application. The SaaS application is already present on the cloud.

Disadvantages:

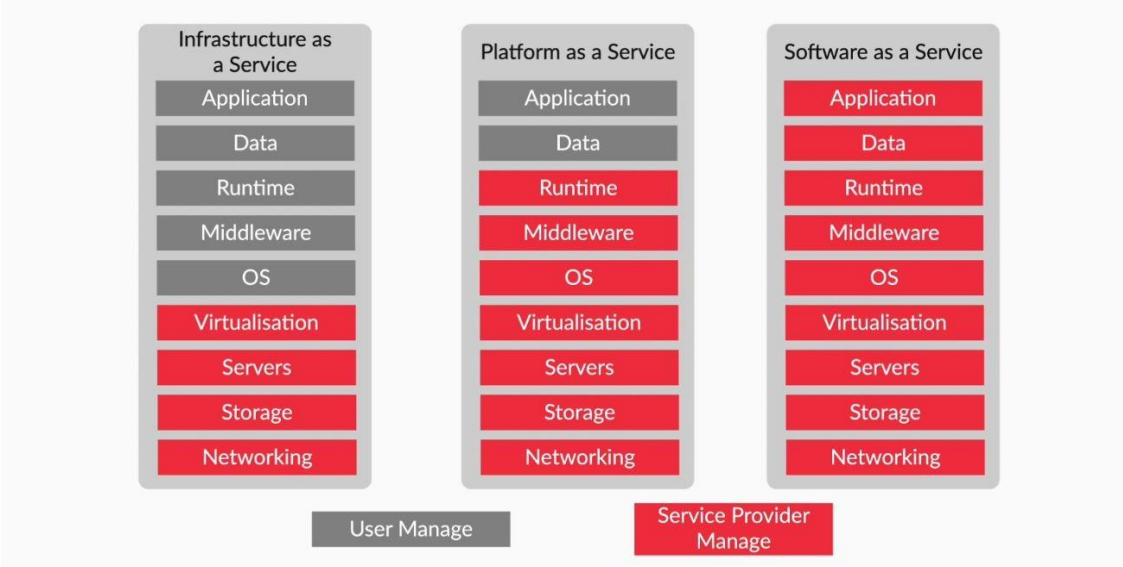
1. Lack of control: Users do not have control over SaaS applications; only the vendor has full control of SaaS applications.
2. Connectivity issue: The applications can only be accessed only via the Internet. Hence, if there is no Internet, then the users cannot access the applications.

Example on Service Models

An end user of your application uploads their picture using the front-end UI. The image is then passed on to the back-end processing engine through a message queue, which allows the different parts of the software to communicate with each other. The back-end processing engine interacts with another machine learning system that detects the objects in the image and returns the result to the user through back-end processing engine, message queue and the front-end UI.



An overview of the different cloud service models from the perspective of the data centre stack

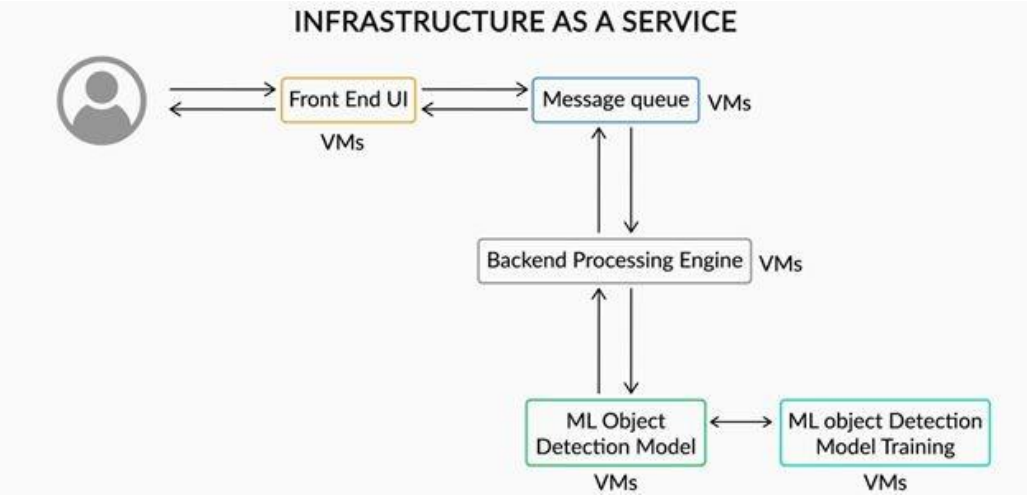


1. **How would this application be deployed in the IaaS model?**

    a. Here, we will need virtual machines to host the front-end UI, messaging servers, back-end processing engine and the machine learning system. We may need to have additional virtual

machines as a backup in case the main servers go down. It is a major activity to maintain these systems and install the necessary software on each of the virtual machines. Also, you need to ensure that patches and updates are applied regularly.

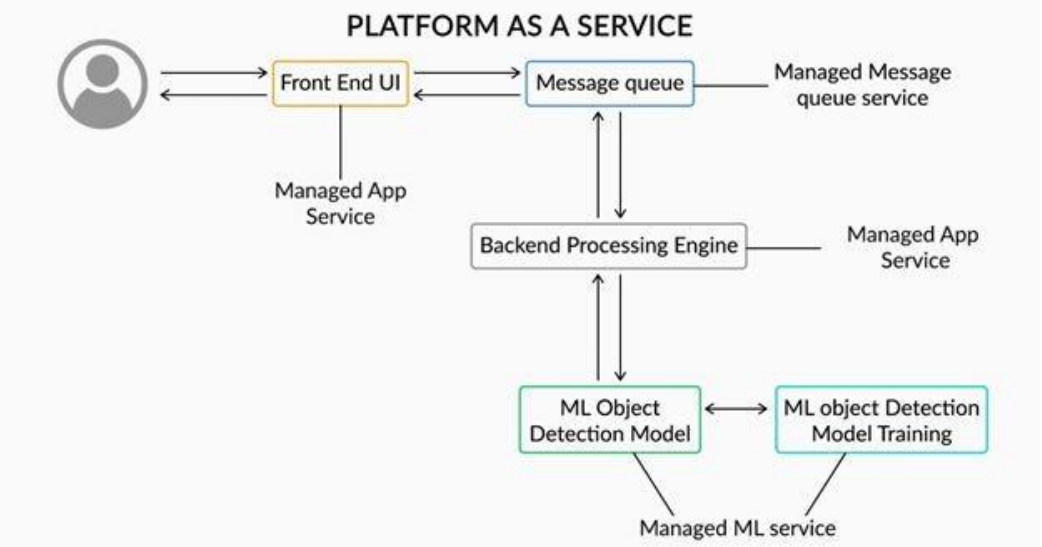
b. Hence, we require a dedicated cloud administrator to manage the virtual machines, along with creating almost 12–13 individual instances. Hence, implementing this solution in an IaaS-only model would require a team of dedicated cloud administrators, along with a front-end team, a back-end team and data scientists to build the application.



2. How would this application be deployed in the PaaS model?

a. In the PaaS model, cloud service providers offer managed solutions to most of the components of our application. To host a web service, you can use a managed app service, wherein you, as the cloud service user, need to simply upload the compiled code to the service. The managed application service takes care of scaling and maintaining the underlying servers for deployment.

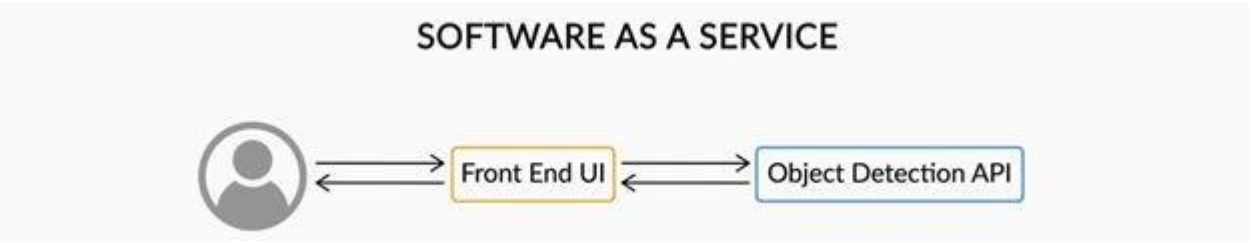
b. Hence, the front-end and back-end processing engines can be deployed on such a managed application service. There also exist managed message queue services that eliminate the need to install the message queue on a dedicated virtual machine. Finally, there also exist well-managed machine learning services that eliminate the need to manage the underlying machine learning servers individually.



c. As you can see, the number of servers to be managed individually has come down drastically, which eliminates the need for a dedicated cloud administrator in the team.

3. How would this application be deployed in the SaaS model?

Here, the cloud service provider provides the object detection API to the customer. This API will completely take care of the backend and machine learning tasks of detecting the object. All that the customer has to do is maintain a front-end team to connect with the API. Now, this API connects with the frontend team and delivers the output to the customer based on the image provided.



Summary

Amazon Web Services

In this session, you were introduced to Amazon Web Services and some of its popular services. By now, you also have some experience working with these services.

By now, you should be familiar with the following Amazon services.

1. AWS IAM
2. AWS S3
3. AWS EC2
4. AWS RDS
5. AWS Lambda

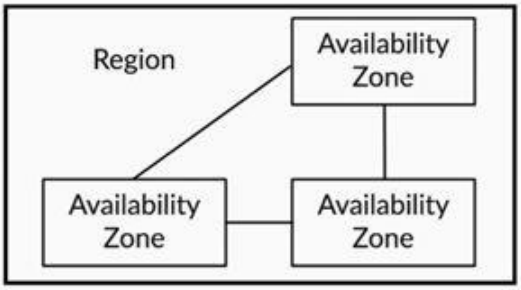
Why AWS?

Though most cloud service providers have many overlapping services addressing similar use cases, Amazon Web Services, or AWS, has emerged as a market leader. The key reasons for its success are its early presence in this field and a rapid pace of innovation, which has led them to **offering more than 149 services**. AWS is consistently ranked as a market leader in all categories of cloud services. Amazon Web Services has a **market share of more than 49%**. This is almost three times its closest rival's share, which makes it a widely adopted cloud service across industries.

Identity Access Management Service

1. The data centers of AWS are physically situated in various locations across the world.

a. A single geographical region is called ‘region’. Every region has multiple isolated locations or data centers known as availability zones. For example, Asia Pacific (Mumbai) is a region that has three availability zones.



b. Regions are represented in AWS services by a region code. For example, the Mumbai region has a region code of ap-south-1. The availability zones are usually represented in the format shown, where the letter identifier such as ‘a’, ‘b’ or ‘c’ stands for different availability zones. For example, an availability center in the Mumbai region would be represented as ap-south-1a. It is always important to create your AWS resources in appropriate regions. The closer the region is to the end application users, the better the performance would be.

2. **Identity and Access Management,** or IAM, is a management service from AWS that enables you to manage who accesses your AWS resources and how they can access it securely. Knowledge of this service is essential for creating any other services in AWS.

   a. Since AWS provides a lot of services, it is essential to control which user or resource has access to other services. For example, the ability to delete a virtual machine must rest with only a restricted few users; else, there is a risk of loss of data or malicious users taking advantage of the system. Hence, IAM provides a unified way to control the access to each of the services and resources.

   b. You will not be charged for using this IAM feature of AWS.

Users, Groups, Roles and Policies

Users

A 'user' is an entity in AWS that represents a person or an application that interacts with the AWS services. User access is usually authenticated with user credentials, namely, username and password.

With IAM, you can create users and grant or deny them access to AWS resources and services.

1. AWS root user: Once you create a new AWS account, you will be provided with single login credentials, namely, your registered email ID and password. This account is called a root user and it has complete access to all AWS services. (However, the accounts that you get from NuvePro are not root users.)
2. Once you create an AWS account, you will begin with a single sign-in identity that is, your email id and password. This account will have complete access to all AWS services. This identity is called an AWS account root user. IAM enables you to create new users and grant or deny them access to the required resources. IAM also allows you to grant access to your users based on the day, time and IP address.

3. There are two ways in which an IAM user can access AWS. These are as follows:
    a. Console access:
        - Amazon Web Services provides a web console to access the services.
    b. Programmatic access:
        - AWS provides command line tools called AWS CLI and SDKs in various languages to enable accessing the AWS services without a console. The authentication in this case is done through keys known as secret key ID and access key ID.

4. There are three types of IAM users, namely:
    a. Privileged administrators
        - These require the console access for managing the AWS users and resources.
        - These are usually administrators who maintain the accounts for organisations.

    b. End users
        - These require access to the content in AWS.
        - These are the users who use just a limited set of services within AWS. They do not have access to other services. For example, an end user might be authorised to create and delete virtual machines, but he would not be able to access other services that enable him to create managed database or store objects.

c. Programmatic users

- They are required to programmatically access the data in AWS.
- This would be applications that internally call AWS managed services, such as managed message queue and managed logging mechanisms, as part of their application logic.

Groups

1. Groups allow granting the same permissions to a set of users.

2. Consider this scenario: There are 50 users in your organisation who require administrative access to a resource. It will take a long time to grant each of them access individually. So instead, you can create a group called 'admin' and grant all the administration permissions to it and then add all those 50 users to that 'admin' group. If a new user joins the organisation and requires administrative access, this can be done by just adding them to this admin group. In another case, if the user wants to shift their job within the organisation, then the permissions required for them also changes; this can be done by removing them from the existing group and adding them to the relevant group.

Policies and permissions

1. A policy is an object in AWS that, when associated with an identity or resource, defines their permission.

2. You can control the access of AWS services by assigning policies to IAM identities or resources.

3. IAM identities represent users, groups of users or roles. An AWS resource could be any entity created by AWS. For example, a virtual machine created by Elastic Compute Cloud service is a resource.

4. When a user or role requests access to an AWS resource, AWS evaluates the policies attached to them, based on which the request is allowed or denied.

5. Policies in AWS are JSON documents.

Types of policies

1. **Identity-based policies**: Identity-based policies are policies that you can attach to a user, a group or a role. These policies control what actions a user, group or role can perform, on which resources, and under what conditions.

2. **Resource-based policies**: Resource-based policies are attached to resources. A resource-based policy grants permission to the identity specified in the policy document. These can be other resources in the same or a different account.

Elements in a policy document structure

1. Version: This specifies the version of the language used in the policy document. The latest version is 2012-10-17.

2. Statement: The set of permissions for a specific resource or user is encapsulated in this element. Each statement has the following sub-elements:
    a. Sid is a unique ID used to represent each statement.
    b. It specifies whether the statement allows the action mentioned in the policy or denies it. It takes either of the two values: 'Allow' or 'Deny'.

3. Principal: This is required only with a resource-based policy. It represents the account, user or role that can be allowed or denied access to the resource. This does not apply for identity-based policies.

4. Action: This includes a list of actions that can be performed on a resource that the policy allows or denies.

5. Resource: In the case of an identity-based policy, this can be used to specify the list of resources on which allow or deny actions need to be performed.

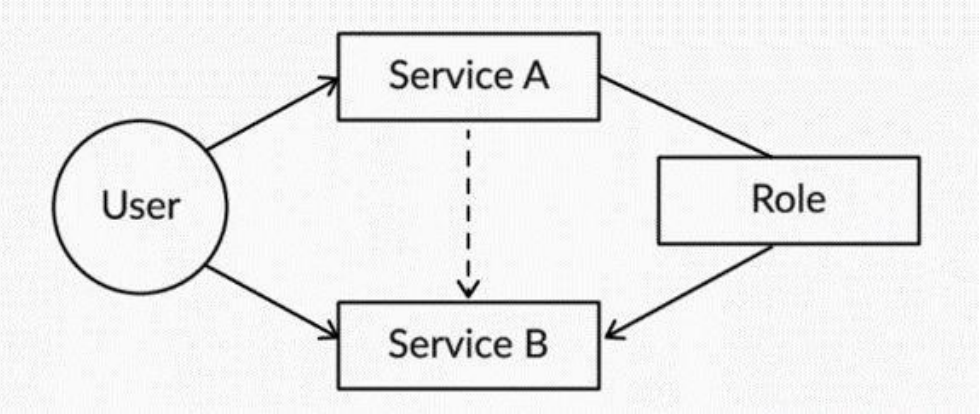
6. Condition: Instead of a blanket access or denial, it would help to specify certain circumstances in which the policy can grant or deny permission. This is an optional element.

Roles

IAM Roles are entities that specify policies and permission for making service requests.

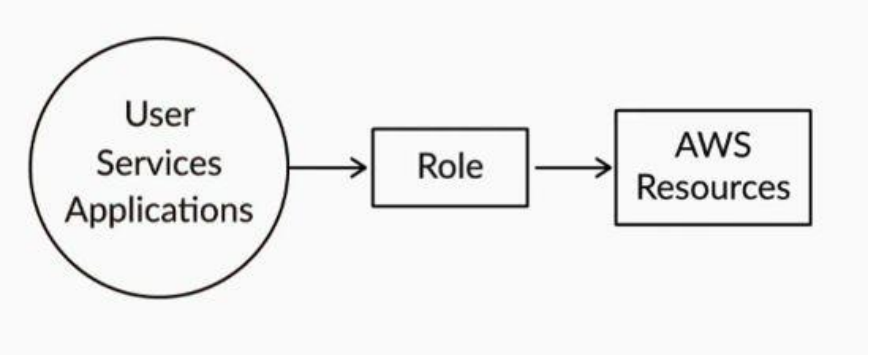
1. To access any service in AWS, the user is required to provide the credentials based on which the authentication is performed and the user is either allowed or denied access. It would be good for the user to delegate certain access to other entities without having to explicitly share credentials with them. IAM roles can be used to achieve this.

2. For example, consider a user who has permission to service A and service B. If service A needs to access another service B, this can be done by assigning the permissions for B through a role and assigning that role to service A. Now when service A needs to access service B, it need not provide the user credentials; it can access it directly through the role.



3. IAM role is an IAM identity similar to an IAM user. An IAM user is an AWS identity with the policies that determine the permissions of the user to perform their tasks. On the other hand, the role can be assumed by any user or any AWS service, if needed.

4. Using the IAM role, you can grant access to AWS resources to users, services and applications. Unlike an IAM user, the role does not have any credentials. An IAM role cannot directly access any service; it can only be assumed by an entity like user or service and access resources through that.



IAM: Best practices

1. Delete the root user access keys. A root access key has permission for all services within your account. To prevent misuse of the root access key, it is best to delete it. The best practice would be to create users for specific purposes and use their access keys.

2. Use multi-factor authentication for user authentication. This includes validation with an external device or application, such as OTPs through mobile phones or applications on the top of password-based authentication.

3. Use groups to assign permissions. This would reduce the overheads in user management.

4. Enforce rotation of credentials regularly through IAM. This would help to avoid the misuse of user credentials to access the cloud infrastructure, especially with accounts that have a lot of users.

5. Grant least privilege of access to each resource, and you can have a fine-grained access control system through policy documents.

Amazon Simple Storage Services

1. Amazon S3 is an object storage service, which can be used to store files with their metadata. Amazon S3 is highly available and scalable. It is an important service that can be used in backup and restore, for archiving purposes and hosting static websites.

2. Amazon S3 acts as an intermediate storage and also as long-term storage for many other AWS services.

3. Suppose you are running an image uploading website that is hosted on AWS. The uploaded images can be stored on AWS S3, which is the most cost-efficient and effective service for such use cases. Amazon S3 is often used in conjunction with other analytics and IoT services as well.

Advantages of Amazon S3:

1. AWS S3 is highly durable. There is a very low chance of losing your file in Amazon S3, as it automatically creates copies of your files across multiple physical systems.

2. AWS S3 is highly available. Amazon commits to an availability of 99.99% availability. This is possible due to the fact that when you specify a region for AWS S3, it automatically replicates the data to at least three availability zones within that region.

3. AWS S3 is highly scalable. It offers unlimited storage with low cost pricing.

Popular key concepts

1. **Objects**
   These are the core elements of Amazon S3. Any image file, text file or binary file that is stored in S3 can be considered an object. These objects contain two types of data:
   a) Object data
   b) Metadata
      - Metadata is a set of name-value pairs describing the attributes of an object.
      - Some examples for a S3 object metadata include the last modified date of an object and the size of the object.

2. **Key**
   This is the name that is assigned to an object. It is a unique object identifier that can be used to retrieve the object.

3. **Bucket**
   Buckets store the objects of Amazon S3. Every bucket in Amazon S3 should have a unique name globally. You can create buckets specific to the required region.

4. **Versioning**
   Buckets allow multiple versions of the objects to be stored. This can be done by enabling versioning on the bucket properties.

5. **Unique address**
   Every object in Amazon S3 can be uniquely addressed with the combination of the web service endpoint, bucket name, key and version.

Bucket policy

1. Bucket access can be configured by user policies or by bucket policy. A bucket policy is a **resource-based policy** that can be used to control which identities or resources can access and perform activities on that bucket.

2. By **default, all the buckets are private and accessible** either through programmatic access or console, based on their policy or role access levels. By modifying the bucket properties, a bucket can be made public as well; a bucket made public would be accessible by anyone.

3. **All uploads to Amazon S3 are atomic**, that is, if the upload of a new version of a file is in progress, and simultaneously, the request to download the same file is executed, then the user gets either the older version of the file or the latest version, only if the upload is complete. It never gives out an incomplete or corrupted version of the file.

4. Amazon S3 also provides **read-after-write consistency**, that is, any file uploaded would be immediately available for download. However, for an update or delete, S3 promises eventual consistency, which means that the changes would be seen only after all copies of the object located in different availability zones are updated or deleted. This happens over time, not instantaneously.

5. Depending on the use cases for which AWS S3 is used, S3 has **multiple storage classes.**
These include S3 standard, which is used for frequently accessed data and is the default option. Next, there is S3 with intelligent tiering for varying access patterns, S3 infrequent access for long-term but less frequently accessed data, and finally, S3 Glacier, that is used for archival purposes.

Amazon ELastic Compute Cloud

Amazon Elastic Compute Cloud provides virtual computing instances or virtual machines on Amazon Web Services. A virtual machine thus created with EC2 service is called an instance.

1. Major benefits of Elastic Compute Cloud (EC2):

    a. Amazon EC2 provides rapid scaling and elasticity. It enables you to create and destroy instances within minutes. Using the Amazon EC2 Auto Scaling feature, the instances can be scaled up and scaled down automatically based on demand.

    b. It provides complete control over the instances. It is very easy to shut down or stop the instances when not in use, or destroy or terminate them when no longer needed.

    c. It is reliable and secure. It assures 99.99% availability. The access to each of the EC2 instances is controlled by identity and access management roles and policies in conjunction with Amazon Virtual Private Cloud.

2. EC2 uses private public cryptography to encrypt and store the login information. The **private-public keys** together are known as key-pair. A key pair is required to log in to an EC2 instance through ssh.

3. Amazon EC2 instances can be used for a variety of use cases including web servers, code repositories, batch data processing job execution, engineering applications, high performance databases and much more.

4. Each of these use cases requires specific configurations of CPU, memory, storage and capacity. There are many such predefined configurations provided by Amazon EC2. These are called **instance types**.

Instance types

Instance types are broadly classified into the following:

1. **Compute instance type**
    a. This is suitable for compute intensive application that requires high performance processors.
    b. Certain use cases include high performance computing, such as dedicated game services, machine learning inferences and media transcoding.
    c. These instances provide choice of the high frequency processors such as Intel Xeon and AMD EPYC processors.
2. **General purpose instance type**
    a. General purpose instances are used in cases where an equal balance of compute, memory and networking resources is required.
    b. Typical use cases include low-latency interactive applications, small databases, virtual desktops, development environments, code repositories and microservices.
3. **Memory optimized instance type**
    a. Memory optimized instances are suited to run applications that perform large in-memory computations.
    b. Typical use cases include real-time big data analytics, web in memory cache applications and high performance databases.
4. **Storage optimized instance type**
    a. Storage optimized instances are suited for high, sequential, low latency read and write access to local storage.
    b. Typical uses cases for such instances would include running NoSQL databases such as MongoDb, hosting data warehousing systems and log processing applications.
5. **Accelerated computing instance types**
    a. The accelerated computing instance type is used mostly for graphics processing and performance intensive activities such as data pattern matching and complex computations.
    b. Typical use cases for such instances include advanced text analytics, deep learning training on images, object detection, image classification and computational finance-related applications.

Choosing the right instance type for a use case is important in any cloud deployment.

AWS AMI

Amazon EC2 provides various preconfigured operating system templates along with options of software such as Tensorflow and PyTorch pre-installed. These various pre-configured instance templates are called Amazon Machine Images, or AMIs.

1. Amazon **provides a set of standard AMIs covering standard operating systems**. This includes Ubuntu, Amazon Linux, and CentOS. It also provides variants of licensed Windows Servers as well.

2. AMI **defines what operating system and application configurations you would need** in that EC2 instance. You can choose a preferred AMI while creating the EC2 instance.

3. You can also build **custom AMIs** using the tools and SDKs provided by AWS. In a custom AMI, you could start with a standard AMI and add custom software above it and then convert it into an AMI.
    a. Now, any EC2 instance in your account can directly launch instances in the future with this AMI, without the need to reinstall all the custom software.
    b. For example, suppose you need to create an EC2 instance with Jupyter Notebook installed. You could create a custom AMI by selecting a standard Ubuntu-based AMI, install Jupyter Notebook on it, and then create a custom AMI of this with tools provided by Amazon. Now, whenever you want to launch an EC2 instance with Jupyter Notebook, you can just launch an EC2 instance with this custom AMI directly.
    c. Such custom AMIs can be shared with other AWS accounts as well.

EC2 instance pricing

There are four types of instances from a pricing perspective that enable the user to optimise their bills by using them as per their requirements.

1. **On-demand instances**
    a. This is the **default way in which instances are priced**. You pay only for the specific compute capacity by the hour or the second based on the instance type. This demands no long-term commitments or large upfront payments.
    b. On-demand instances are mostly used with short-term or unpredictable critical workloads. If you are trying out Amazon EC2 for the first time, this would be a good pricing method to use.
2. **Spot instances**
    a. They allow you to request **spare computing capacity**.
    b. These cost **90% less than the on-demand instance prices**.
    c. The spot instances will be **provided by Amazon only if spare capacity is available;** the request for spot instance will be in a pending state until then.

d. Amazon can **terminate a spot instance** by providing a spot instance interruption notice, where a two-minute warning is given to the user before it is interrupted.
e. Due to the unpredictability of the availability of instances, these are recommended in use cases that have **flexible start and stop times**. This could include background processing, data processing and optional tasks.

3. **Reserved instances**
    a. Some applications have a predictable usage, such as data warehouse needs. In such cases, you are already aware of what your requirement for an instance would be, what its configuration should be and for how long the instance needs to be run. In such cases, you could opt for reserved instances.
    b. Here, customers can commit to a 1-year or a 3-year term plan and choose the all upfront payment option or a partial upfront one with a monthly minimal amount. Based on the plan you chose, reserved instances could provide up to 75% discount to an EC2 user.

4. **Dedicated instances**
    a. As seen in the IaaS section, dedicated instances provide you with a **dedicated physical server**.
    b. These are a bit costlier when compared to the other instances. In most cases, depending on your instance type, specific instance sub-type and the pricing type, the price of an instance usually varies from 25 cents per hour to $40 per hour. Also, the incoming and outgoing data are charged at about $0–$1 per GB depending upon the consumption. Any additional hard disk storage would be priced between $0.002 and $0.1 per GB.

Virtual private cloud

In many cases, when you have multiple cloud resources created, such as multiple EC2 instances, it is good to isolate them so that the communication among these resources is controlled. This is required to ensure fine-grained security, and it also helps to create a logically isolated section within AWS. Amazon virtual private cloud lets you create a virtual network on cloud.

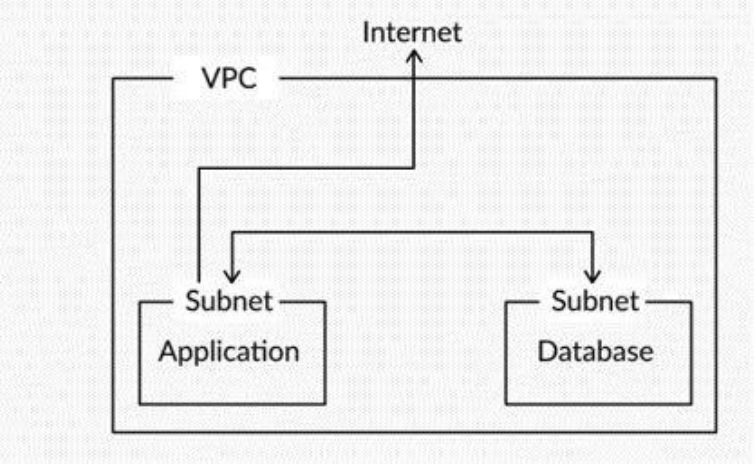
1. With VPC, you can create subnets, select your own IP address ranges and configure route tables based on the requirements. You would be able to also connect this VPC to other VPCs so that EC2 instances can communicate across VPCs.

2. Amazon VPC provides flexibility to configure the access to the resources.

3. Suppose you need to maintain a database that is accessible only via your application servers. Here, the subnet within the VPC in which the database resides can be made private-facing with no Internet access. Access to the database service can be provided only to the application servers. The

subnet where the application servers reside can be made public so that end users can access the application. Hence, external users will have access only to the application and not the database.

4. As with any networking infrastructure, Amazon VPC provides an internet gateway, end points, route tables, NAT gateway and subnets



AWS security groups

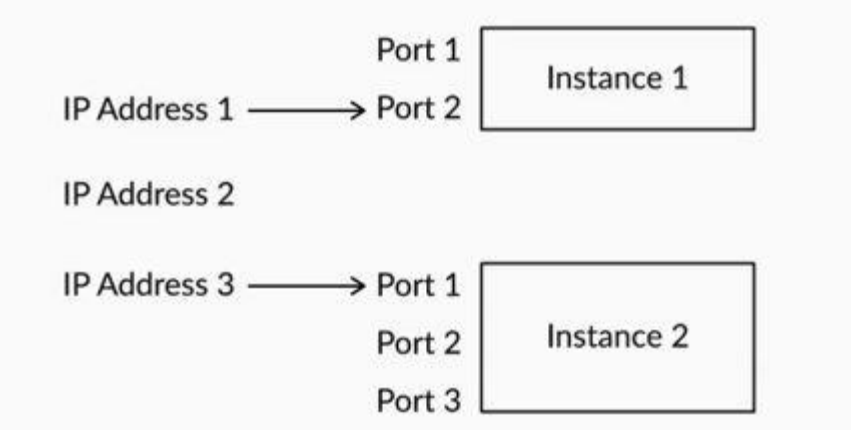
To ensure security, we need to make sure that only intended users from specific IP addresses and ports are able to access the instances. To enable this, AWS provides security groups.

1. With security groups, the end user can specify which specific IP addresses can be allowed to access which instance on which port. Rules can be added not only to control the incoming data, or Ingress, but also outbound data, or Egress.

2. Rules for a security group can be modified anytime, and the new rules will be applied to instances associated with the group immediately.

3. For each specified rule, you need to provide the port, the protocol and the source or destination IP that can be allowed to access the instance.

4. Each AWS account comes with a **default security group and a default VPC**.



Amazon RDS and AWS Lambda

Amazon RDS

1. Amazon RDS is a **PaaS offering from Amazon Web Services**. It is a managed relational database service. RDS currently supports MySQL, MariaDB, Oracle, Microsoft SQL Server and PostGreSQL.

2. Amazon RDS **relieves you of standard database tasks** such as backup, patching up and monitoring.

3. It also provides enhanced availability along with automated failover mechanism, using the **Multi-AZ deployment option**. Here, the primary database in one data center is synced up continuously with a replicated secondary database.

4. Amazon RDS also provides **read replicas**, wherein heavy read activities are redirected to a secondary database. This reduces the load on the primary database.

5. It provides **automated backups** for the data stored. It also ensures automatic instance replacement in case an instance in the database cluster goes down.

6. It provides **encryption** both in transit and at rest. On an RDS database running with Amazon RDS encryption, the data, the automated backups, read replicas and snapshots are all encrypted.

DB instances

The core part of an RDS is the DB instance.

1. A DB instance is an **isolated database environment** in AWS. It **can contain multiple databases**. It is the basic building block of Amazon RDS, and you could have **upto 40 DB instances**.
2. When a DB instance is created, it is **created with a master account** along with the password by default.
3. The DB instance **can be accessed using standard RDBMS client tools** such as MySQL Workbench and DBeaver. This is similar to accessing any other standalone database instance.

Storage types

Storage types determine the configuration with which your DB instance to be created. Based on your requirements, you can select one of the following three storage types in RDS.

1. **General purpose SSD**: This provides single-digit millisecond latencies and is usually suited for a wide range of workloads.
2. **Provisioned IOPS**: This is more suited for databases that require low latency and consistent IO throughput. This is specific for I/O intensive workloads.
3. **Magnetic**: This is just provided for backward compatibility. Prior to the advent of hard disks and SSDs, most large databases were stored in magnetic tapes. Hence, to support such applications that still use this technology, RDS with magnetic tapes was provided.

Instance types

Like Amazon EC2, Amazon RDS provides a wide range of instance types to choose from. This instance type comes in combinations similar to those of EC2 instances. These EC2 instance types are used to build the DB instance behind the scenes.

There are two main types of instance types for RDS:

1. **General purpose instance** types, which are usually used for a wide variety of smaller database applications; and
2. **Memory optimized instances,** which are optimized for memory-intensive database workloads.

RDS pricing

1. The cost of using RDS is based on the instance types, the DB instance configuration, storage types and the database engine used.

2. Based on the pricing type, there can be multiple types of DB instances:
    a. On-demand DB instance, where you can pay as you use and you would be charged per hour for the usage accordingly; and

b. Reserved instances, which give you an option to reserve a database for a 1–3 year term by paying full upfront or partial upfront.

3. In most cases, depending on the DB instance type and the storage size, the cost can vary from $0.10 per GB to $0.20 per GB.

4. Apart from this, RDS also charges for data transfer costs, for the data transferred in or out of the database.

5. Depending on the region, the data transfer costs around $0.01 per GB to $0.02 per GB.

6. Database backups in RDS are free for up to 100% of the storage size. Additional backups cost $0.095 per GB per month.

AWS Lambda

1. AWS Lambda is a **serverless computing offering from AWS**. It lets you run the code directly without having to deal with any of the infrastructure hassle.

   What is serverless computing?
   a. With serverless computing, modern applications can be developed with increased agility.
   b. In a serverless computing, AWS takes care of provisioning, patching, capacity adding besides managing code runtime.
   c. This comes at the cost of flexibility, which means that you cannot log in to compute instances or customise the operating system or language runtime.
   d. This would enable the developers to solely concentrate on code development. The developed code can be just directly uploaded and executed on AWS Lambda.

2. AWS Lambda plays a big role in architecting and implementing microservices-based applications on cloud.

3. AWS Lambda provides an **end-to-end automated administration,** thereby nullifying the need to manage instance types, runtimes and additional loggings. Unlike other services that come in fixed configurations, with AWS Lambda, the memory required and the runtime details can be modified anytime.

4. The code that you run on AWS Lambda is called a **Lambda function**.
   a. Every function comes with options to modify configuration details, such as memory, runtime, and configuration information such as name, description and function entry points.

b. Lambda functions can be used to execute minimal code upon external triggers. These external triggers could include addition of a file in S3, or activation of time-based triggers that automatically call the function at a specific time.
c. AWS Lambda, being lightweight, can be used as a tool to call other services when the function is invoked.
d. It provides you options to add or upload code within the service.
e. It also provides the function handler property, which provides the starting point of the execution.

AWS Lambda pricing

1. AWS lambda charges you on the basis of how many requests have been made to your functions. It also charges on the basis of the duration it takes for your function to serve those requests by executing the code internally.

2. The free usage tier includes 1 million free requests with 4,00,000 seconds compute time per month. Post this, depending on the region, $0.20 per million requests is charged.