**Introduction to Cloud Computing & AWS**(Service Models, Development Models, Fundamentals Of AWS Cloud)

* As the data is increases it motivated use of cloud to handle it more efficiently.
* Cloud let us to rents the services we need and use them.
* Server providers ensure the security , infrastructure etc, so that we can focus on business
* It connects the world
* It lets us use our applications from anywhere in the world and manage them according to us.

**-> SERVICE MODELS**

* Cloud providers provide users with service models that fit to our needs
* **SAAS:** software as a service , we only need to consume the applications and services without worrying about the scaling.
* **PAAS:**platform as a service, it allows us to develop applications. Eg. google cloud engine
* **IAAS:**infrastructure as a service, provides complete infrastructure such as os, softwares etc to develop applications.

**-> DEPLOYMENT MODES**

* it means different types of cloud available in the market
* Public Cloud
* Private Cloud
* Hybrid Cloud

**-> CLOUD PROVIDERS**

**Introduction to AWS**

* Reason for its to be best in the market : flexibility, scalability, and pricing, compute capacity

**-> Advantages:**

1. Flexibility
2. Cost Effective: pay as u go model, charges only for the services we are using
3. Scalability: scaling up and down is very easy
4. Security: great security mechanism

**-> AWS ARCHITECTURE**

* Widely spread and covers every area that needs to be covered
* Divided into 2 parts:
* 1. Region : areas of world where data centers are located, 1 region might contain more than 1 data centers. These are called “availability zones”.
* 2. Availability zone

**-> AWS DOMAINS**

* Compute : EC2 it is a clean slate I.e the machine is completely new without any libraries before, ElasticBean Stalk contains some pre defined libraries before hand
* Migration : physical migration is also provided I.e we can move data from 1 data center to other, service which provides this is “SnowBall”
* Security and Compliance : IMA - identification and authentication management tool, KMS - helps to create own public and private keys
* Storage : S3 works as a bucket - bucket is where data is stored and object which we store are files, CloudFront - content delivery network, Glacier - a place to store archives.
* Networking : VPC - virtual nw to move or launch services, Direct Connect - least internet connection that can be used within aws
* Messaging : Cloud Trial
* Databases : storage stores exe files byt database stores files sql like.
* Management Tools : CloudWatch - lets us set alarms

**-> INSTANCE**

**-> EC2 PRICING MODELS**

* On Demand instance : the instance u demand and u get it. We get it for a limited time frame say hourly basis. Once the hour is complete the instance gets terminated on its own
* Dedicated: given to organizations so that there data is secured and instance is not shared by other users
* On Spot : good for volatile data , cheap as available on the price we bid but only for limited time after which it gets terminated on its own
* Reserved : fixed set of rent and is available for longer duration

**-> EC2 INSTANCES BASED ON THEIR GENERAL FUNCTIONING**

* Burstable : category of general purpose instances with base cpu utilization. If there is more traffic and requires more performance from current performance to max performance. Use Case :Analysis of Customer Data
* EBS Optimized : app to process data at high speed, provide high input and output processing, Use Case : Auto Response E-Mail System
* Cluster Networking : forms clusters of instances, each cluster meets a particular need, Use Case: Search Engine and Browsing
* Dedicated : provides and ensure more security , Use Case : Confidential Data Processing

**-> Security Group :** allows the connection from instance to an external nw and also the connection to instance from external network

**-> IAM Role :** Access Management Policies

**-> Change Termination Protection :** it helps us to prevent accidental termination of an important instance, so what it does is when we try to terminate an instance it asks us to go into the instance and change the policy and then delete it.

**-> ELASTIC BEANSTALK :** automated version of ec2, don’t have control on os but there is control on other configuration , used to deploy applications , we just upload code on the aws infrastructure

**-> AWS LAMBDA**

* again its an automated version of ec2, no control over os, but here we cant even control and decide the configuration of the server
* Not used to deploy applications
* Used for executing background tasks
* Depending on the workload configuration is assigned to us
* Used with sdk’s
* Sdk’s are api which can be integrated in our code
* To install in eclipse
* Help -> install new software -> enter website name(aws.amazon.com/eclipse) -> hit enter -> all tools and sdk’s available will be listed -> select the tools and submit -> it will be installed
* S3 bucket is configured in this way that when its called it invokes the lambda function
* When data is stored to S3 its also stored in SQS(Simple Queue Service)
* Lambda can get the details of the latest file uploaded to S3
* DNS Server = Route53 service
* SES : used to send the emails
* my.dot.tk is the site which helps to create free domains for 3 months
* On changing the nameservers for the domain it will point to route 53

**-> AWS ELASTIC BEANSTALK**

* Computer warehouse : place where different computer components are laid like cpu motherboard, hard disk, etc.
* When we want to deploy an application on our own we have to take a number of decisions like how powerful ec2 instance we want, what amount of storage we want, and infrastructure stack
* When we try to deploy an application using elastic beanstalk we need to focus on just the code of the application. Here things like configuring ec2 instance, installing autoscaling group , security mechanism and management tools are done by elastic beanstalk
* Its a paas that is a platform where we can deploy and host our applications
* Offers quicker deployment, simplifies operations, cost effective, multi-tenant architecture - allows to share ur application on different platforms with security
* Paas provides security means it will provide us with a detailed report of application usage, different users using the application etc
* Paas provides to know whether user is getting a better experience out of it or not
* It helps to select the aws resources which are provided to the application by elastic beanstalk like iaas
* It will create a dedicated folder for our application containing all the softwares, files and other tools required for the application in that folder
* At a single point of time it can run only a single version of your application in one environment
* if we want our application to handle http request then we chose server environment tier
* if we want our application to handle background tasks then we chose worker environment tier
* **Architecture:** 
  + Elastic load balancer : it handle the load on our application, when an environment is set up for our application a url is assigned to our application and an alternative name ‘cname’ is also given, and this url in form of cname is used to point to elastic load balancer and it distributed these requests to various ec2 instances or auto scaling group
  + Auto scaling group : when there is high traffic and scarcity of ec2 instances then it adds up the ec2 instances , similarly if the load is less it terminated the underused instances
  + Container type : depending on the requirements of our application it installs the various softwares on our ec2 instances
  + Host manager : runs on all instances, provides detailed report of the performance of app, monitors log files, monitors app server
  + Security group : elastic beanstalk allows users to use our application from port 80
  + Worker : is a process that handles background tasks which are time intensive and resource intensive

**-> Storage Services(Cloud Storage)**

* cloud storage : storage which is made available as a service which is connected over a nw, it cal let u store emails, media(images, videos, files etc), services(websites), backup(large enterprises that let us backup our data). not a redundant or dead storage but smart storage
* Ensures that there is mechanism that allows storage of data and allows performing various operations on this data
* Cloud storage ensures disaster recovery
* Services on cloud ensure that users working on cloud becomes easy
* Cloud platforms are more secure
* Aws provides shar3ed security model where customers and aws team decide what accesses should be given, thus there is control of security in our hands
* Practices to be followed:
  + Scrutinize sla : vendors and customers decide on some rules, so we need to decide points which benefits our business
  + Follow business needs : chose proper storage acc to ur requirements(static data, real time data)
  + Ensure security : being monitored by org should be implemented properly
  + Plan ur storage future: do u want to scale up in future, max load time, etc
  + Beware of hidden costs : understand what we are paying for and how much we are paying for
* S3(Simple Storage Service) : object kinda storage, bucket where data is stored and data stored is an object, when we require faster retrieval of data
* EFS(Elastic File Storage) : used to store files and helps to connect to nw
* S3 Galcier : used to store archival data ie the data that we are not going to be used everyday, provides delayed retrieval of data, very cheap compared to S3
* Storage Gateway : let u move data from on premise software to cloud
* EBS(Elastic Block Storage) : lets create volumes snapshots and copies and attach them to instances
* Instances can be backed up with 2 kinds of storage that are S3 and EBS, EBS is important as it helps to create images and volumes(hard disks or ssds) also lets to manage the storages,
* Snapshots in instances help to replicate my instances and volumes attached to it
* We can create copy of the volumes so that if anything goes wrong we can use it to recover our data
* When we create a volume to attach an existing instance the region in which it is created should be the same as the region of the instance
* Volumes can’t be detached until the instance is first stopped and then we detach the volume from it and attach another volume to it
* When we attach a volume to an instance it will ask for device where we have to fill the details of the folder where volume is attached when it is created, it can be derived from root device name from instance details

**-> S3 SERVICE**

* Provides durability of 99.9999999999 %
* Durability is provided using checksums
* We can store 5gb of data for free
* works on buckets and objects
* Buckets are containers containing the data
* Objects are the files which are stored in a bucket
* Objects come with required metadata(key and its version id)
* Depending on the region where bucket is created our cost may vary
* We decide which region the bucket should be created depending on which region would provide an efficient access to the data for the users
* Also latency is there for far away regions
* It is a global service and not specific to a particular region
* S3 Storage Classes :
  + Standard : low latency , faster fetching of data, also storing for shorter duration
  + Infrequent Access : latency period would be low , retrieved quickly , data should be stored for a longer time
  + Glacier: retrieval speed is slow and data is stored for a longer duration and thus is cost efficient
* Versioning : it means keeping multiple copies of the data , multiple copies of the data are stored on different data centers so that If 1 data center goes down the data is still accessible from other data center, and this is done by making versions of the data
* by default versioning is disabled in S3
* It prevents overwriting and accidental deletion
* We can fetch a particular version of data by specifying its id
* Cross region Replication : using it we can copy the objects from the bucket in one region to the bucket in other region
* We can specify the id and delete all the versions
* We can also specify the life cycle of all the versions of a file(first 3 months in standard storage, 2 months in infrequent and then to glacier)
* Here if the data expires = expiration and transfer of data from 1 storage class to other = transfer
* We can replicate our data as well
* When we are performing replication from 1 bucket to other bucket but if sending bucket has versioning enabled then receiving bucket should have versioning enabled as well
* S3 Transfer Acceleration :
  + If we want to move data from 1 location to a far away location without high latency it can be used here
  + **Cloudfront** : used to cache our data close to our destination to provide faster retrieval

**-> Networking Services**

* VPC : virtual private cloud, if we launch all our services in 1 vpc then all the services will be able to communicate with each other easily, provides security
* Direct connect : it provides direct connection to the aws infrastructure from user side without the need of an internet connection
* Route 53 : domain name system, redirects url from ur location to the ip address where website is hosted

**-> AWS CLOUDFRONT**

* It sets an intermediate point where the data is cached first and data is provided at a faster pace
* Uses 3 notations
  + Routing : it ensures that there are quite a few edge locations that located near to the user and cache the data over there so that it is available to quickly
  + Edge locations : has regional cache that is the data which is more frequently queried in a particular region
  + More availability
* Benefits :
  + Accelerate static web content delivery by caching the data
  + Serve on demand or live streaming video as edge location sends the data as the first byte reaches it from the server
  + It encrypts the data and provides more security
  + Cutomization at the edge
  + Uses lambda edge to customise the data and provide it privately

**-> Monitoring and Management**

* **Amazon CloudWatch**
* Monitoring tool is required to observe how the application is working on the cloud and what improvements we need to make in the app so that it is in accordance to the current needs of the customers
* This tool will help users by identifying the errors in the application before hand and will rectify them so that business of the user is not affected
* Cloudwatch provides us with 2 levels of monitoring
  + Basic : app is monitored less frequently and we are provided with less number of metrics to chose from
  + Detailed : montoring is done frequently and provided with wide range of metrcis to chose from
* It allows to assign upto 10 dimensions to a metric
* Cloud watch metric math : it is a service that integrates a number of metrics and create a new time series and we can view this time series on cloudwatch dashboard
* CloudWatch Events : delivers a real time stream of system events that describe change in aws resources
* To use a lambda service we need to create a policy using iam dahsboard
* We need to create a role to assign lambda permissions to handle ec2 instances
* CloudWatch Logs

**-> Amazon Cloud Formation**

* for a complex application there is need of many aws resources and handling all the resources together is a mundane task
* So to manage all these resources we have a service called cloud formation
* Using it we can manage , create and provision all these resources together at a single place
* Its a tool using which we can create our applications quickly
* We can create templates in cloud formation where we putting all the resources needed and defining the dependencies of these resources and saving this design as a template
* This template can be used to create as many app we want
* We just need to create a template and a stack which we will do our work
* JSON doc has its format:
  + AWS template format version
  + Description
  + Metadata
  + Parameters : values that we will be passing to a template
  + Mappings
  + Conditions
  + Output
  + Resources

**-> AUTO SCALING AND LOAD BALANCER**

* Snapshots and AMI’s : snapshot is the copy of data in the hard drive of our instance but its not a bootable copy , whereas ami is a bootable copy of the instance and is used to replicate an instance so we don’t need to do all the config again
* Auto scaling determines the amount of load which is coming in and scales accordingly and distribute the load accordingly
* The traffic that is distributed equally in all the created instances is done by load balancer, determines which server is more free and can serve the query
* In the ami we do have the data that is to be present in the instance but we don’t know the machine that we want to launch
* This needs to be specified in launch configuration
* Http security group is to be added so that launched instances can communicate with each other
* Auto scaling group is used to specify the conditions in which we have to perform auto scaling
* Try to create auto scaling group on default vpc
* Load balancer performs the routing
* Types of load balancers:
  + Classic : distribute load equally among the servers
  + Application : depending on the task to be performed load is distributed accordingly

Here targets are auto scaling groups

* In load balancers we need to create target groups and we need to link the auto scaling group to the target group

**-> CLOUD SECURITY**

* we will opt for a private cloud when we want to store very confidential files on the cloud.
* Data that is not confidential and we don’t mind if public sees it
* We have both private and confidential files and also the files we want to seen by the public, we will be using hybrid cloud
* Cloud security is a mixture of both science and art, its a science as we need to come up with new techniques to ensure the security of the application , its an art as we need to create technologies for security with user experience in mind.
* Cloud security is done in 3 steps :
  + Monitoring data : ai algo which know what normal system behaviour is and any deviation from this normal behaviour cloud experts get an alert and start working on it
  + Gaining visibility : we should understand what caused that problem , cloud experts use tools that help them to know what caused that problem
  + Managing access : once we identified the problem and its root, and if some user did that then using the managing access we can wipe out that user from the application
* **Cloud Security in AWS:**
  + Monitoring : Service in aws Cloudwatch which is cloud monitoring tool
  + SNS is a protocol used to send notification
  + Gaining Visibility : we have to track whatever activity taking place in our account
  + CloudTrail is an aws service which logs all the API calls
  + It can let us know when the hacker got access into our system
  + It logs all the activities that is happening in aws will be logged in cloud trail
  + Managing access : aws iam provides this fascility , it is used to provide granular permission and helps to secure our system
  + Policy is the permissions given to the user

**-> IAM**

* To accomplish the access management we need this
* In a company instead of a number of accounts we have only 1 aws account by=ut we define the privileges a person has for the account
* **Components :**
  + Users : we need a user because we need to give permissions to a user
  + Any person who wants to access my aws account he has to be added as a user in aws
  + U should never work using root account and should always work using administrative access
  + We have two types of access :
    - Aws management console : using aws to actually deploy the resources or using other services
    - Programmatic : when we use aws in our program with help of apis etc
* Secret access key and access key is provided only once when the user is created so we need to store it
* I can do anything using administrative account as I can do with root account but if logged out require root access
* Groups : we need groups when we have some users where they have common accesses
* Only when we select the programmatic access we require the secret access key
* Roles : roles are assigned to applications
* Policies : permission which are given to a user , instance, roles
* If we have given permissions of allow as well as deny then it will give preferences to the least permissible policy that is deny

-> **MULTI FACTOR AUTHENTICATION**

* it provides 2 layer security as along with the password we need to enter a code as well
* With aws there is an app google authenticator which allows to implement virtual multi factor authentication

**-> DATABASE SERVICES**

**-> AWS REDSHIFT**

* Tradiditonal data warehouse : disadvantages - autoscaling is difficult and leads to poor performance, and high costs
* **Amazon Redshift** 
  + Amazon dataware is a collection of compute resources called nodes
  + When theses nodes are organized into a group they are called clusters
  + Each of the cluster runs an amazon redshift engine and contains 1 or more databases
  + Jdbc = java database connectivity is an api used to query from the database and uses language java
  + Odbc = other database connectivity uses sql for querying the databases
  + There are 2 types of nodes :
    - Dense storage : storage optimised and used to handle heavy work loads and basically they use hard drive storage
    - Dense compute : are compute optimised and used to handle high performance intensive work loads , and mainly use ssd kind storage
* Advantages of columnar storage : better data compression , query on similar kind of data is fast
* 5439 is the port to be handled by redshift

**-> DevOps Services**

* Something that works in development environment might work in production environment
* Developer : develops the code , updates the code for new changes and sends the code to operator for verification of its correctness
* Operator : they ensure that the application should work correctly in production environment as well , they deal with the customers get their feedback and if any changes are to be made, if they can implement themselves then they will else they will tell the developer to make the changes
* Conitnuous Integration and Delivery :
* CodePipeline : continuous delivery service which lets us automate steps required to release a software
* S3 should be in the region where our pipeline is so that data transfer between pipeline and S3 is easy

**-> QUESTIONS**

* Hybrid cloud is established by including ur public cloud severs in virtual private cloud and connecting this virtual cloud with ur on premise severs using VPN
* Eip is charged when we allocate it to an instance and not released when not used
* We are charged coz ip address is a rear commodity , eip (elastic ip add) is static that is wont change even when its stopped and run again, and if we allocate an eip and does not use it so it is blocked with us and people outside us cant use it , so aws charges us for it
* Aws emr(elastic map reduce) is used for deploying a hadoop cluster
* Aws snowball = used to transfer huge data from 1 place to otherm=, each file is encrypted before transfer
* Primary private address cant be changed
* Secondary private addresses can be changed
* Provisioned iops is used when we want fast input output and is used when we have continuous data , thus used when we have batch oriented workloads
* Standby instance is used for failover and cant be used with primary instance
* We should not used mysql on ec2 and used distributed database (rds)
* Read replica has small size compared to a snapshot
* DynamoDB = No SQL database
* elastiCache = fast storage but is not a database
* DynamoDB has the ability to scale more than RDS or any other relational database services, thus it is highly available
* If insatnce of db is deleted then there is no way getting its data back but when an instance is tried to be deleted we are asked whether we want to create a final snapshot of db and if we create that snapshot we can recover the data of the instance
* Use cases of dynamoDB = managing web sessions, storing json docs, storing metadata for S3 objects
* Provisioned io is very expensive so we can use elastic cache as it provides the same function of provisioned io but with reduced cost, so we are getting faster and at less cost the data
* For bigger work load we can use redshift as it is easy to scale and work would be done in parallel through the nodes