AWS Solution Architect and Sysops by Techincal Guftugu

Trainer : Bhupinder Rajput (B.Tech,M.Tech,MBA) AWS-CSA,GCP,CCNA,HCIG,AZ-103 Certified

Follow on Facebook : https://www.facebook.com/Technical-Guftgu-1964284537005718/

AWS Solution Architect & Sysops Youtube Playlist : https://www.youtube.com/-playlist?list=PLBGx66SQNZ8a_y_CMLHchyHz_R6-6i-i_

INDEX :

Session 1 : Introduction to Cloud Computing,Characteristics,Top Cloud Players,AWS Certifications

Cloud Computing :

• Cloud computing is the on-demand delivery of compute power,database,applications,storage and other IT resources through a cloud service platform via the Internet, with pay-as-you-go pricing model

Top Players in Cloud :

1.AWS
2.Microsoft Azure
3.Google Cloud Platform
4.Alibaba Cloud
5.Oracle,Vmware,IBM,Fujitsu

Characterstics of Cloud :

1.On demand self service
2.Broad Network Access

3.Scalability
4.Resource Pooling
5.Measured Services

AWS was launched in 2006.....AWS was moved to amazon.com in 2010.....Certifications was launched in 2013.....2015-2016 Amzon's Profit got doubled.....2017 Reinvent A.I

AWS Certifications :

1.Foundational : AWS Cloud Practitioner

2.Associate & Professional :

• AWS Solution Architect.(Associate and Professional)
• AWS Developer(Associate).....AWS Devops Engineer(Professional).....Development and Operations(DEVOPS)
• AWS Sysops Admin (Associate).....AWS Devops Engineer(Professional).....System Admin and Operations

3.Specality : AWS Advanced Networking,Security,Machine Learning,Alexa Skill Builder,Database,DataAnalytics

Session 2 : Service and Deployment Models in Cloud

Services in Cloud :

1.Infrastructure as a Service(IAAS) : Network to O.S
2.Platform as as Service(PAAS) : Network to Runtime
3.Software as as Service(SAAS) : Network to Application

Layer Model
Application
Data
Runtime
Middleware
O.S
Virtualization
Server
Storage
Network

Deployment Model in Cloud :

1.Public Cloud : Any one can create account and access services from service platforms like AWS,Google,Microsoft etc
2.Private Cloud : Also Known as enterprise cloud which are only available to the enterprise and its branches across the globe with the help of internet which is more secure than

Public Cloud
3.Hybrid Cloud : Merging of both Public and Private clouds to reap both their respective benefits

• Cloud Service Providers provide virtual private cloud also

Hypervisors used by Cloud Platforms :

• AWS uses Citrix as hypervisor for Virtualization
• Microsoft uses Hyper-V
• Vmware uses Vsphere Esxi

LABS :

Session 3 : Creating AWS Account

Session 1 : Introduction to Elastic Compute Cloud(EC2)

Elastic Compute Cloud : Amazon EC2 provides scalable computing capacity in the AWS Cloud

• We can use Amazon EC2 to launch as many or as few Virtual Servers as we need,configure security,Networking and manage Storage
• Amazon EC2 enables us to Scale Up or Scale Down the Instances Capacity
• Amazon EC2 has two Storage options i.e EBS & Instance Store
• Preconfigured templates are available known as Amazon Machine Images
• By default when we create an AWS account with amazon, our account is limited to a max of 20 instances per ec2 region with two default High I/O Instances

Types of EC2 Instances :

1.General Purpose : Balanced Memory and CPU
2.Compute Optimized : More CPU than Ram
3.Memory Optimized : More Ram
4.Accelerated Computing/GPU : Graphics Optimized
5.Storage Optimized : Low Latency
6.High Memory Optimized : High Ram,Nitro System
7.Previous Generations

Session 2 : General Purpose EC2 Instances

General Purpose Instances : These provide a balance of Compute,Memory and Networking Resources and can be used for a variety of Workloads

• 3Series in General Purpose Instances

1.A Series(Medium and Large) : A1
2.M Series(Large) : M4,M5,M5a,M5ad,M5d
3.T Series(Large,Medium,Small,Nano) : T2,T3,T3a.....T2 micro is eligible for Free Tier.....micro comes under nano type

• Available in Four Sizes : Nano,Small,Medium,Large

1.A Series :

a.A1 Instances : These are ideally suited for scale-out Workloads that are supported by ARM Ecosystem

• These instances are well suited for web server,Containerized microservices,Caching Fleets,Distributed Data Stores,Applications that require ARM Instruction Set

2.M Series :

a.M4 Instances : These Features a Custom Intel Xeon E5-2676 v3 Haswell processors optimized specifically for EC2
Capacities : VCPU : 2 to 40(MAX).....Ram : 8 to 160GB(MAX).....Instance Storage : EBS Only

b.M5,M5a,M5ad,M5d Instances : These Instances provide an ideal cloud infra,offering a balance of Compute,Memory,Networking Resources for a broad range of Applications
• Capacities : VCPU : 2 to 96(MAX).....Ram : 8 to 384GB(MAX).....Instance Storage : EBS and NVME SSD
• Used in Gaming Servers,Web Servers,Medium and Small Databases

3.T Series :

a.T2,T3,T3a Instances : Provides a Baseline level of CPU Performance with the ability to burst to a heigher level,when required by our Workload
• An unlimeted instances can sustain high cpu performance for any period of time and whenever required
• Capacities : VCPU : 2 to 8(MAX).....Ram : 0.5GB to 32GB
• Used for Websites and Web App,Code Repos,Developement,build,Test and for Microservices

Session 3 : Compute Optimized EC2 Instances

Compute Optimized Instances : Are ideal for compute-bound applications that benefit from high performance processors

• It only has C Series which has C4,C5,C5n

1.C Series

a.C4 Instances : These are optimized for compute intense workloads and deliver very cost effective and high performance at a low price per compute ratio
• Capacities : VCPU : 2 to 36....RAM : 3.75 to 65 GB.....Storage : EBS Only .....Network Bandwidth : 10GBPS
• Used for Web Server,Batch Processing,MMO Gaming,Video Encoding

b. C5 Instances : These are optimized for compute intense workloads and deliver very cost effective and high performance at a low price per compute ratio and are powered by NITRO SYSTEMS
• Capacities : VCPU : 2 to 72....RAM : 4 to 192 GB.....Storage : EBS & NVMe SSD .....Network Bandwidth : 25GBPS
• Used for High Performance Web Servers,Gaming,Video Encoding

• C5 support max 25 EBS Volumes and uses elastic network adapter and new EC2 Hypervisor(AWS Nitro System)

Session 4 : Memory Optimized EC2 Instances

Memory Optimized Instances : These are designed to deliver fast performance for workloads that process large data sets in memory

• These consists of 3 serieses....they are R,X,Z

1.R Series : R4,R5,R5a,R5ad,R5d

• High Performance,Relational(My Sql),Nosql(MangoDB,Cassandra) databases
• Distributed web scale cache stores that provide in-memory caching of key value type data
• Capacities : VCPU : 2 to 96.....RAM : 16 to 768GB....Instance Storage : EBS & NVMe SSD
• Used in Financial Services,Hadoop etc...

2.X Series : X1,X1e

• Well suited for High Performance Database,Memory intensive enterprise applications,Relational database Workload,SAP HANA, Electronic Design Automation
• Capacities : VCPU : 4 to 128.....RAM : 122 to 3904GB....Instance Storage : NVMe SSD

3.Z Series : Z1d

• High Frequency Z1d deliver a sustained all core frequency of upto 4.0GHZ,the faster of any cloud Instances
• AWS Nitro System,Xeon Processor upto 1.8TB of instance Storage
• Capacities : VCPU : 2 to 48.....RAM : 16 to 384GB....Instance Storage : NVMe SSD
• Use cases are Electronic design automation,certain database workloads with high per core licensing costs

Session 5 : Storage Optimized EC2 Instances

Storage Optimized Instances :These are designed for workloads that require high sequential read write access to very large data sets on local storage
• These are optimized to deliver tens of thousands of low latency,random I/O operations per second(IOPS) to application
• These consists of 3 Serieses...they are D,H,I

1.D Series : D2

• Well suited for massive parallel processing(MPP) data warehouse,MAP reduce and hadoop distributed computing,Log or data processing app
• Capacities : VCPU :4 to 36.....RAM :30.5 to 244GB.....Storage : SSD

2.H Series : H1

• This family features 16TB of HDD based local storage,high disk throughput and balance
• Well suited for app requiring sequential access to large amounts of data on direct

attached instance storage
• Applications that require high throughput access to large quantities of data
• Capacities : VCPU :8 to 64.....RAM :32 to 256GB.....Storage : HDD

3.I Series : I3 and I3en

• Well suited for high frequency online transaction processing system(OLTP),Relational Databases(No Sql,Distributed file system,data warehousing application)
• Capacities : VCPU :2 to 96.....RAM :16 to 768GB.....Storage : NVMe SSD.....Networking Performance : 25 to 100 GBPS.....Sequential Throughput : Read-16Gb/-s.....Write-6.4GB/s(I3).....Write-8GB/s(I3en)

Session 6 : Accelerated Computing EC2 Instances

Accelerated Computing Instance : These use hardware accelerators or co-processors to perform some functions such as floating point number calculations,graphics processing or data pattern, matching more efficiently than is possible in software running on CPU's

• These consists of 3 Serieses...they are P,G,F

1.F Series : F1

• These offer customizable hardware acceleration with field programmable gate arrays(FPGA)
• Each FPGA contains 2.5 million logic elements and 6800 DSP engines
• Designed to accelerate computationally intensive alogorithms,such as data flow or highly parallel operations
• F1 provides local NVMe SSD storage
• Capacities : VCPU : 8to 64.....RAM : 122 to 976GB.....FPGA : 1 to 8.....Storage : NVMe SSD
• Used in Financial Analytics,genomics research,real time video recording & big data research

2.P Series : P2 & P3

• It uses NVIDIA Tesla GPU's,provide high bandwidth networking,upto 32Gb of memory per GPU,which makes them Ideal for deep learning & computational fluid dynamics
• P2 Instances : VCPU : 4 to 64,GPU : 1 to 16, RAM : 61 to 732GB,GPU RAm : 12 to 192GB,Network B/W : 25GBPS
• P3 Instances : VCPU : 8 to 96,GPU : 1 to 8, RAM : 61 to 768GB,GPU RAm : 12 to 192GB,Network B/W : 25GBPS...Storage : SSD & EBS
• Used in Machine learning,databases,sesimic analysis,genomics,molecular modeling,AI,Deep Learning
• P3 supports CUDA9 & OpenCL API
• P2 supports CUDA9 and Open CL 1.2

3.G Series : G2 & G3

• Optimized for Graphics Intensive Applications
• Well suited for app like 3D Visualization
• G3 instances use NVIDIA Tesla m60 GPU and provide a cost effective high performance platform for graphics application
• Capacities : VCPU : 4 to 64,GPU : 1 to 4, RAM : 30.5 to 488GB,GPU RAM : 8 to 32GB,Network B/W : 25GBPS
• Used in Video creation services,3D visualization,Streaming Graphics intensive applications

Session 7 : High Memory EC2 Instances and Previous Generation EC2 Instances

High Memory Instance : These are purpose built to run large-in-memory databases & including production developments of SAP HANA in the cloud
• These instances are bare metal instances and do not run on a hypervisor
• Only available under dedicated host purchasing category(For Min 3 Years term)
• OS directly on the Hardware

Features :

• Latest Intel Gen Intel Xeon Pentium 8176M Processor
• 6,9,12TB of instance memory,the largest of any EC2 instances
• Powered by the AWS nitro system,a combination of dedicated hardware & Lightweight hypervisor
• Bare metal performance with direct access to host hardware
• EBS optimized by default at no additional cost

• This consists of U series,,,,,they are U-6tb.metal,U-8tb.metal,U-12tb.metal
• Network performance is 25Gb/s and dedicated EBS bandwidth-14GBPS
• Each instance offer 448 logical processors

Previous Gen Instances : T1,M1,C1,CC2,M2,CR1,CG1,i2,HS1,M3,C3 and R3
• These are not deleted instances and we can still purchase these instances

Session 8 : EC2 Purchasing Options

EC2 Instances Purchasing Options :

1.On-demand :
2.Dedicated Instances :
3.Schedule Instances :
4.Reserved Instances(RI) : Standard RI,Convertible RI,Scheduled RI
5.Dedicated Host :
6.Spot Instances :

• There are three ways to pay for EC2 Instance i.e On-Demand,Reserved Instance and Spot Instances
• Dedicated host and Dedicated instances costs are calculated as per On-Demand instance costs and Scheduled instances are billed as per reserved instance costs
• We can also pay for dedicated host which provides us with EC2 instance capacity on physical server dedicated for our use

On-Demand Instances :

• These are virtual servers that run in AWS or AWS Relational Database Server(RDS) and are purchased at a fixed rate per hour
• AWS recommends using these instances for applications with short term irregular workloads that cannot be uninterrupted
• They are also suitable for use during testing and development of apps on EC2

• With these we can only pay for EC2 Instances we use
• The use of these instances free from the cost and complexities of planning,purchasing and maintaining hardware and transforms what are commonly large fixed costs into much smaller variable cost
• Pricing is per instance-hour consumed for each instance,from the time an instance is launched until it is terminated or stopped
• Each partial instance consumed will be billed per second for linux instances and as a full hour for all other instance types

Dedicated Instances :

• Dedicated instances are run in a vpc on hardware that is dedicated to a single customer
• Our dedicated instances are physically isolated at the host hardware level from instances that belong to other AWS Account
• These instances may share hardware with other instances from the same aws account that are not dedicated instnaces
• Pay for dedicated instances is based on on-demand and save upto 70% by purchasing reserved instances and upto 90% by purchasing spot instances when compared to Dedicated instances

Dedicated Host :

• An amazon EC2 dedicated host is a physical server with ec2 instances capacity fully dedicated for our use
• Dedicated hosts can help us address compliance requirements and reduce costs by allowing us to use our existing server bound software requirements
• Pay for a physical host fully dedicated to running our instances and bring our existing per-socket,per-core,per-vm software license to reduce cost

Spot Instances :

• These let us take advantage of unused ec2 capacity in the aws cloud.These are available upto 90% discount compared to on-demand prices
• We can use these for various test & development workloads
• we also have option to hibernate,stop or terminate our spot instances when ec2 reclaims the capacity back with two minutes of notice
• These get interrupted when actually ec2 capacity requirement increases(On-demand and reserved instances) and amazon reclaims the space with 2 min notification.
• These can also get interrupted when the spot price raises above our choosen max spot price

Schedule Instances :

• These enables us to purchase capacity reservation that recur on a daily,weekly or monthly basis with a specified start time and duration,for a one-yaer term
• We reserve the capacity in advance so that we know it is available when we need it
• We pay for the time that the instances are scheduled,even if we do not use them
• Schedule instances are a good choice for workloads that do not run continuously but do run on a regular basis
• Purchase instances that are always available on the specified recurring schedule,for a one-year term
• Eg: We can use these instances for an application that runs during business hours or for batch processing that run at the end of the week

Reserved Instances :

• These provide a significant discount upto 70% comapred to on-demand pricing and provide capacity reservation when used in a specific availability zone
• Reserved instances give us the option to reserve a DB instances for a one or three years term an in turn receive a significant discount comapred on demand instances pricing
• 3 Types of RI....Standard,Convertible and Scheduled RI

• *Standard RI :* These provide the most significant discount upto 75% off on-demand and are best suited for steady state usage
• *Convertible RI :* These provide a discount upto 54% and the capability to change the attribute of RI as long as the exchange results in the creation of reserved instances of greater or equal values
• *Scheduled RI :* These are available to launch within in the time window we reserve....Same as Scheduled Instancs
• We cannot transfer a convertible or standard RI from one region to another region
• We can change the config of convertible RI from ec2 management console or get reserved instance management quota API
• There is no extra charge for converting from one config to another config but we need to pay the cost as per the changed config

Session 9 : EC2 Access,Status Check and EC2 Meta Data

EC2 Access Data :

• To access instances,we need a key and key pair name
• We can download the private key only once
• The public key is saved by aws to match it to the key pair name and private key when we try to login to the instance
• Without key pair we cannot access isntances via RDP or SSH(Linux)
• There is a 20 ec2 instances soft limit per region ,and we can submit requst to aws to increase limit

EC2 Status Check :

• By default aws ec2 instances performs automated status checks every 1 min
• This is done on every running ec2 instances to identify any H/W or software issues
• Status check is built into the aws ec2 instance
• They cannot be configured,deleted or disabled
• EC2 services can send its metric data to aws cloudwatch every 5 min (enabled by default)
• Enabled detailed monitoring is chargeable and sends metrics in every 1 min
• We can not charged for ec2 instances if they are stopped but attached ebs volumes get charged

When we stoppes an ebs backed ec2 instance :

• Instances perform a shutdown
• state changes from running to stopping
• ebs volumes remain attached to the instance
• any data cached in ram or instance store volume is gone
• instances retain its private ipv4 address and any ipv6 address
• instances releases its public ipv4 address back to aws pool
• Instances retain its elastic ip addresses

EC2 Terminate :

• When we terminate a running instance the instance state changes from running to shutting down and then to terminated
• During the shutting down and terminated states,we do not incur charges
• By default ebs root devices volumes are deleted automatically when the ec2 instances are terminated
• Any additional (non boot/boot) volumes attached to the instances by default,persist after the instances is terminated
• We can modify both behaviours by modifying the ‘delete on termination’ attribute of any ebs volumes during instances launch or while running
• Enable ec2 termination protection against accidental termination

Ec2 Metadata :

• This is instance data that we can use to configure or manage the instance
• Eg : ipv4 addr,ipv6 addr,dns hostname,AMI-Id,Instance id,instance type,local hostname,public keys,security groups
• Metadata can be only viewed from within the instance itself i.e we need to login to the instance
• Metadata is not protected by encryption,anyone that has access to the instance can view this data
• To view instance metadata use GET http://169.254.169.254/latest/metadata

Instances User Data :

• Data supplied by the user at instance launch in the form of a script to be executed during the instance boot
• User data is limited to 16kb
• We can change user dat by stopping ec2 first
• User data is not encrypted

EC2 Bare Metal Instances :

• Non virtualized environment
• Operating Systems runs directly on hardware
• Suitable for licensing restricted tier 1 business critical application
• i3 metal,i5 metal,r5metal,z1d metal,u-6tb1.metal

Elastic Block Storage : EBS backed instance

• We can easily replicate between availability zones with snapshots etc..
• EBS volumes attached at launch are deleted when instance terminate
• EBS volumes attached to a running instance are not deleted when instance is terminated but are deattached with data interact
• EBS is network attached storage

Instance Storage : Instance backed storage

• Physically attach to the host server
• Data not lost when os is rebooted
• Data is lost when underlying drive fails,instance is stopped or terminated
• We can not attach or detach to another instance
• Do not rely on for valuable long term data

LABS :

Session 10 : Creating Windows Server in AWS EC2

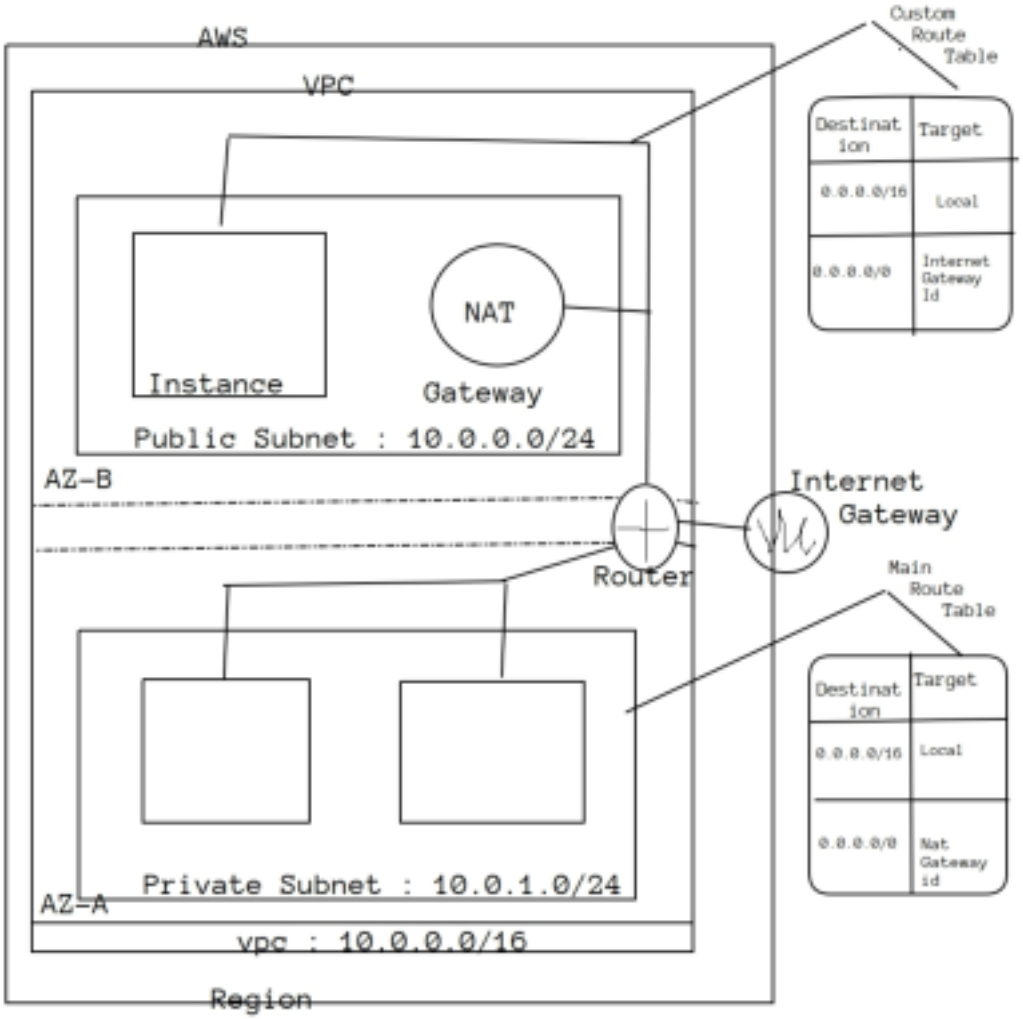
Session 11 : Install Webserver IIS and Creating Webpage in Win Server

Session 12 : Attaching extra volumes in existing Win Machine

Session 13 : Creating Linux Machine AWS EC2

Session 14 : Retriving Metadata of Linux Machine

Session 1 : Intoduction to Virtual Private Cloud



Virtual Private Cloud :

• A vpc is a virtual network that closely resembles a traditional networking that we operate in our own datacentre,with the benefits of using the scalable infrastructure of AWS
• To Simply say vpc is a virtual network or datacenter inside aws for one client
• It is logically isolated from other virtual n/w in the aws cloud
• Max 5 vpc can be created inside one region and 200 subnets in 1 vpc
• We can allocate max 5 elastic ip's
• Once we created a VPC, dhcp,nacl and security group will be automatically created

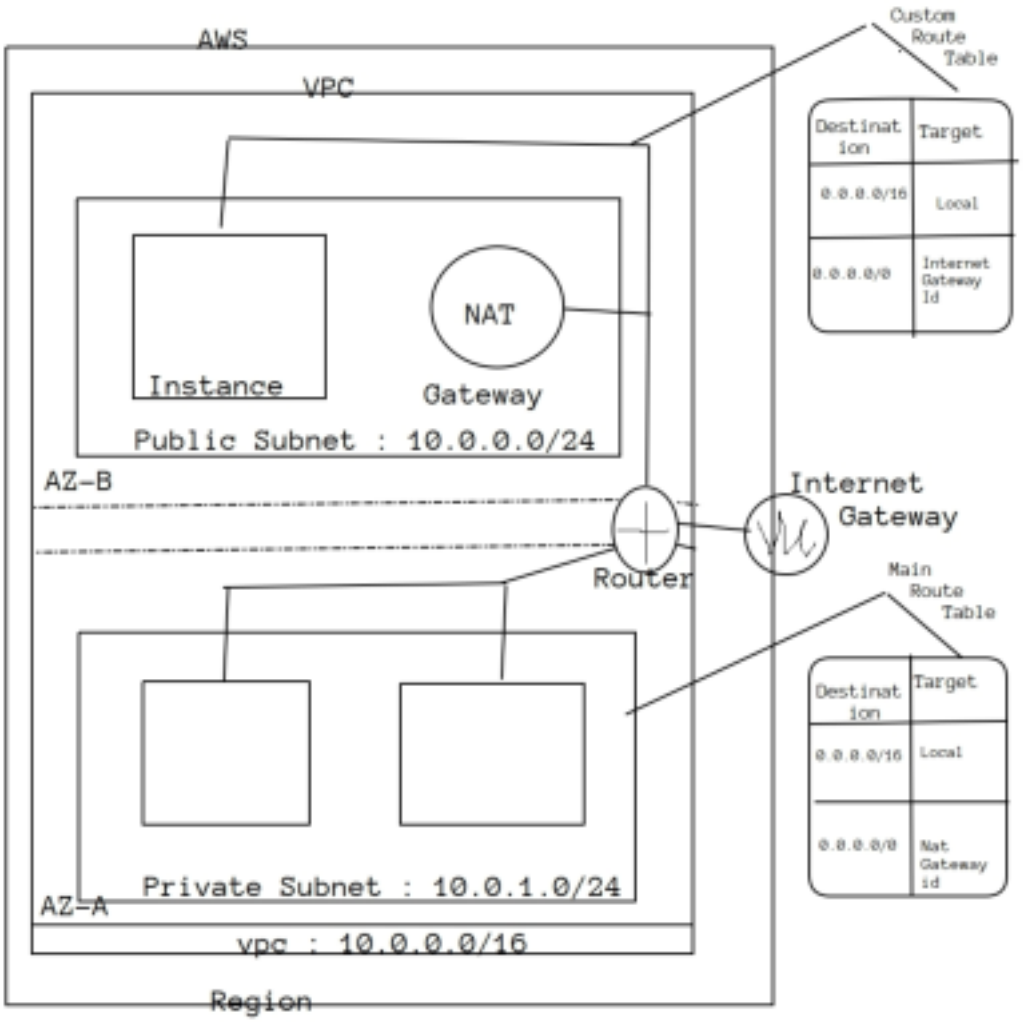
• A vpc is confined to an aws region and does not extend between regions

• Once the vpc is created,we cannot change its CIDR,block range
• If you need a diff cidr size,create a new vpc
• The diff subnets within a vpc cannot overlap
• We can however expand our vpc cidr by adding new /extra ip address ranges(except American gov cloud & AWS China)

Components of VPC :

CIDR & IP Address Subnets,
• Implied router & routing table
• Internet gateway
• Security groups
• Network ACL
• Virtual private gateway
• Peering connections
• Elastic ip

Session 2 : Types of VPC,Public & Private Subnets



VPC Types : Defaut VPC,Custom VPC

Default VPC :

• Created in each aws region when an aws account is created
• Has default cidr,security group,NACL and route table settings
• Has an internet gateway by default

Custom VPC :
• Is an aws account admin creates
• AWS user creating custom vpc can decide the CIDR,
• Has its own default security group,network acl, and route table
• Does not have an internet gateway by default,one needs to be created when needed

Steps to follow to create a VPC :
1.Create VPC
2.Subnet
3.Internet Gateway
4.Route Table

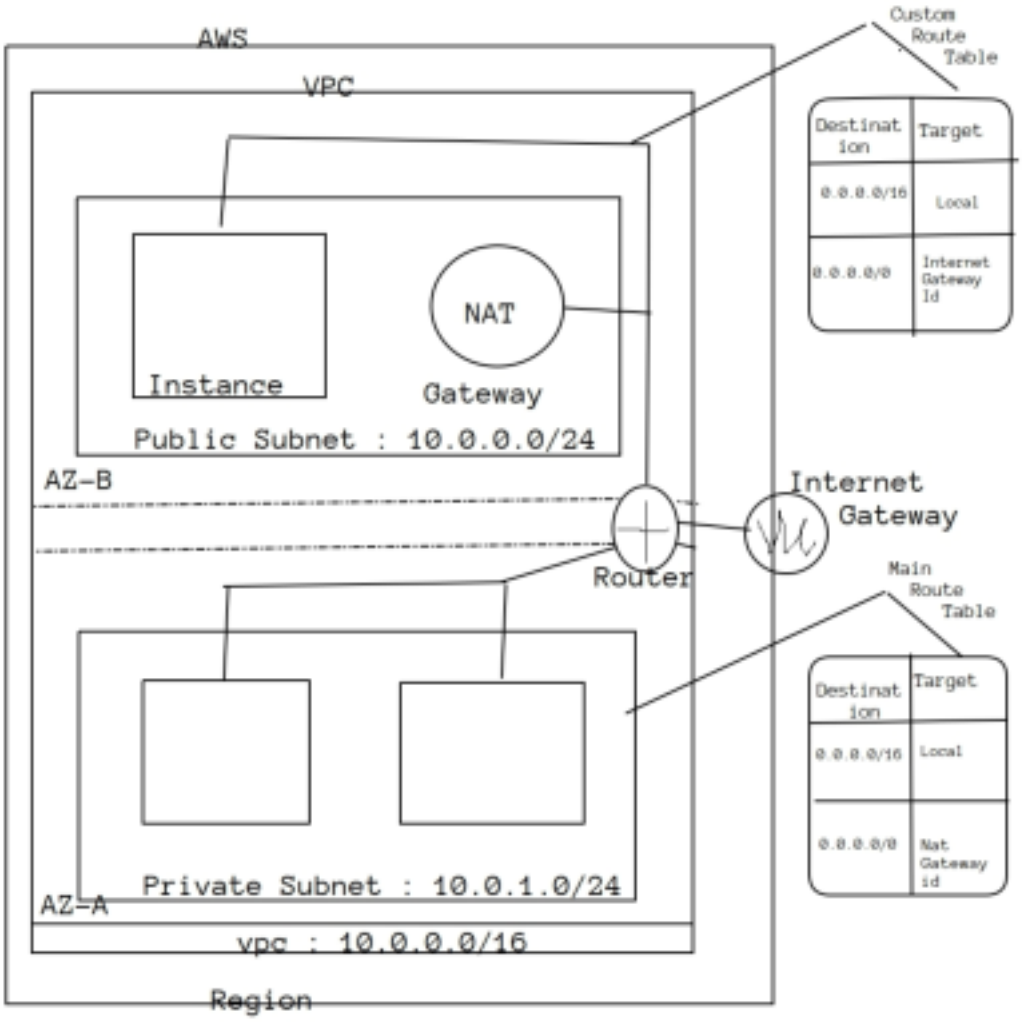
Public Subnet :
• If a subnets traffic is routed to an internet gateway,the subnet is known as public subnet
• If we want our instance in a public subnet to communicate with the internet over ipv4,it must have a public ipv4 addr or an elastic IP addr

Private Subnet :
• If a subnet does not have a route to the internet gateway,the subnet is Known as a private subnet

Note : When we create a vpc,we must specify an ipv4 cidr block for the vpc.The allowed block size is between /16 to /28 and the first four & last ip addr of a subnet cannot be assigned

Eg : 10.0.0.0/24 addrs following are reserved as follows:
10.0.0.0 ------>Network Addr
10.0.0.1-------->reserved by aws for the vpc router
10.0.0.2--------->reserved by aws,The ip addr of dns server
10.0.0.3-------->reserved for future use
10.0.0.25------->broadcast addr
• Aws does not support broadcast in a vpc,but reserves the address

Session 3 : Implied Router,Route Table,Internet Gateway



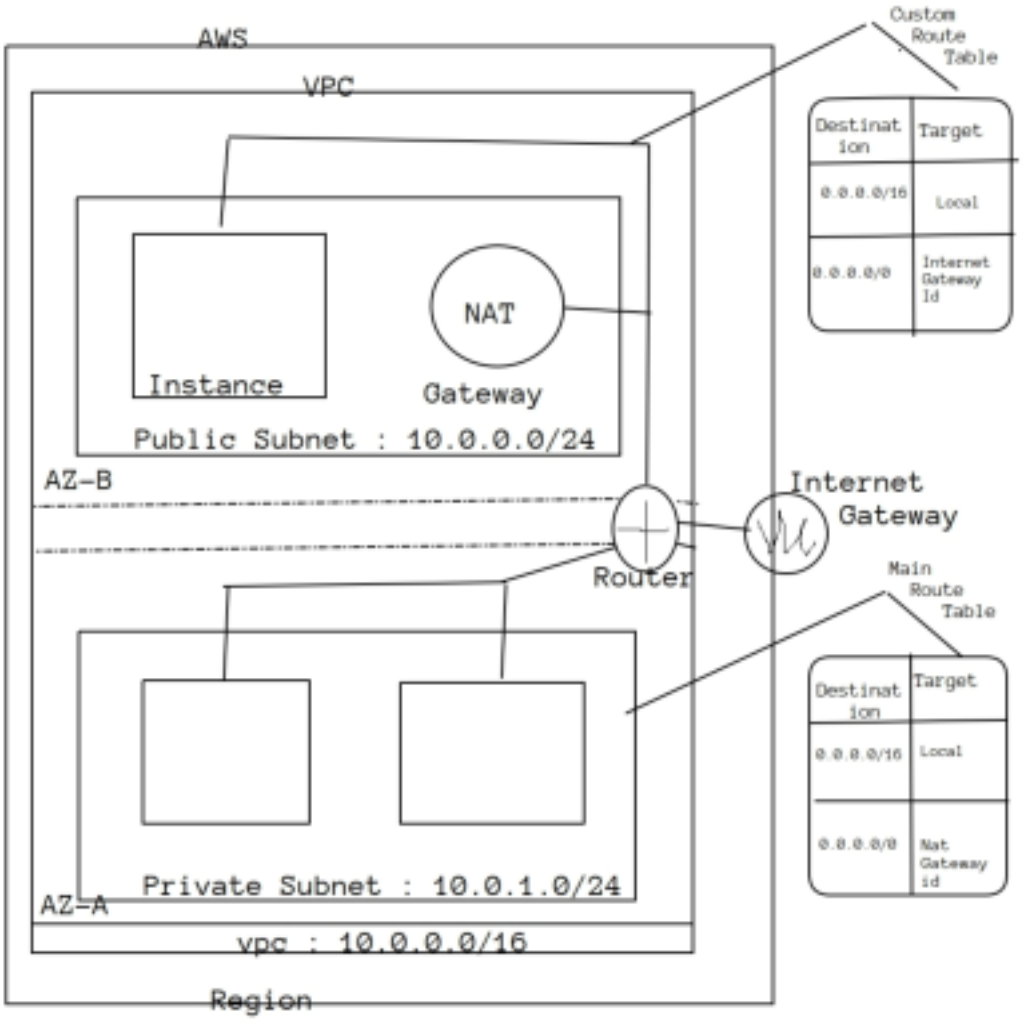
Implied Router & Route Table :

- It is the central routing function
- It connects the different AZ together and connects the vpc to the internet gateway
- We can have upto 200 route tables per vpc
- we can have upto 50 route entries per route table
- Each subnet must be associated with only one route table at any given time
- If we do not specify a subnet to route table association,the subnet will be associated with the default vpc route table
- We can also edit the main route table if we need,but we cannot delete main route table
- However we can make a custom route table manually, make it the main route table then we can delete the former main,as it is no longer a main route table
- We can associate multiple subnets with the same route table

InterNet Gateway : IGW

- An IGW is virtual router that connects a vpc to the internet
- Default vpc is already attached with an IGW
- If we create a new vpc then we must attach the igw in order to access the internet
- Ensure that our subnet's route table points to the internet gateway
- It performs nat between our private and public ipv4 addrs
- It supports both ipv4 and ipv6

Session 4 : NAT Gateway,Security Group,NACL,VPC Peering

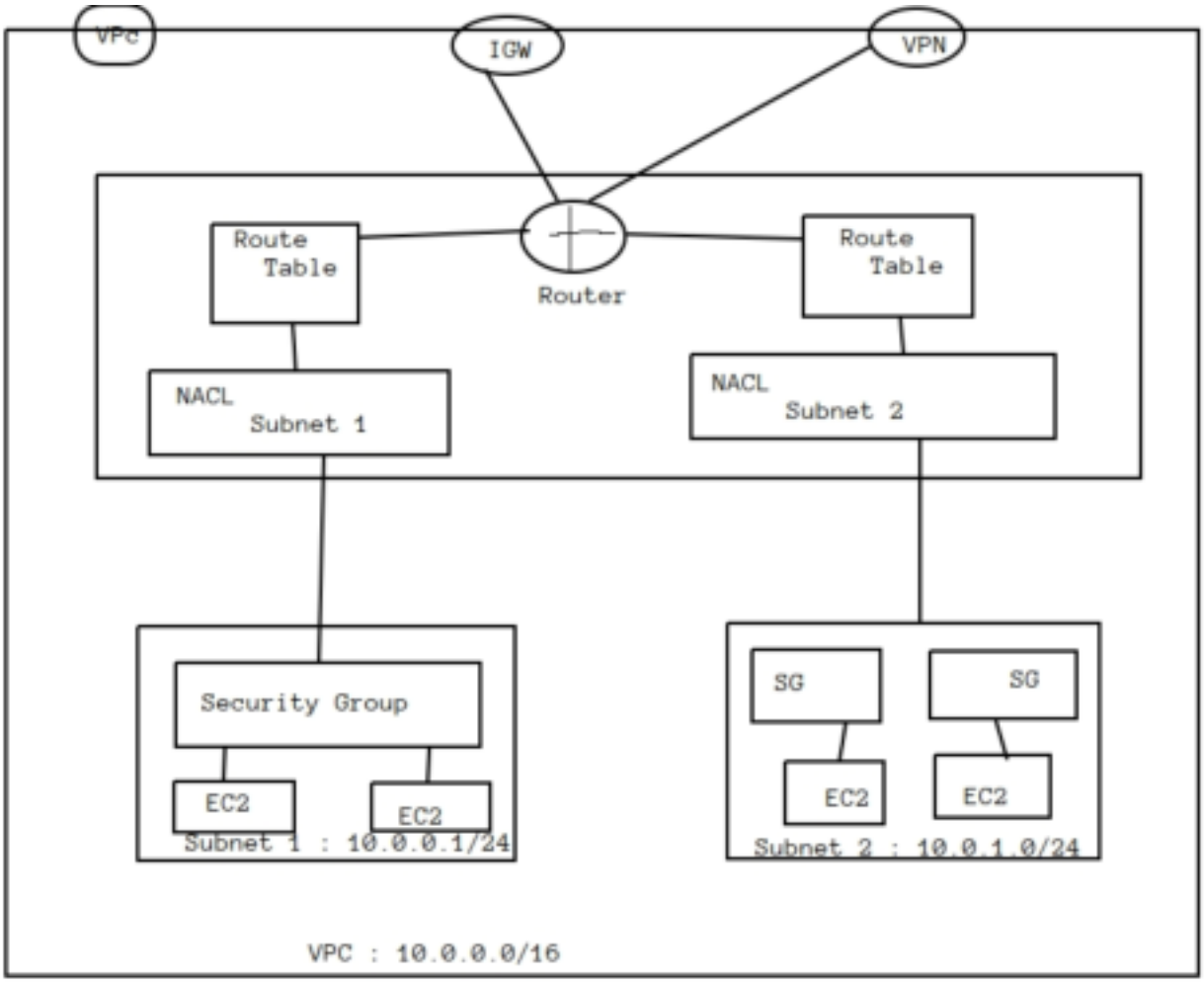


NAT Gateway : Also does PAT(Port Address translation)

• we can use a network address translation gateway to enable instances in a private subnet to connect to the internet or other aws services,but prevent the internet from initiating a connection with those instances
• We are charged for creating and using nat gateway in our account.NAT gateway hourly usage and data purchase rates apply.Amazon ec2 charges for data transfer also apply
• To create a nat gateway,we must specify the public subnet in which nat gateway reside
• We must also specify an elastic ip address to associate with nat gateway when we create it
• No need to assign public ip's to our private instances
• After we have created a nat gateway we must update the route table associated with one or more of our private subnets to point internet bound traffic to the nat gateway
• This enables instances in your private subnets to communicate with the internet
• Deleting a nat gateway,disassociates its elastic ip addr,but does not releases the address from your account

Security Groups :

• It is a virtual firewall works at ENI(Elastic Network Interface) level
• Upto 5 security gropus per ec2 instances interface can be applied
• Can only have permit rules,cannot have deny rule
• Stateful(if imbound allowed then automatically outbound is also allowed and vice versa) : return traffic is allowed then inbound traffic is also allowed,even if there are no rules to allow it



Network ACL :

• It is a function performed on the implied router
• NACL is an optional layer of security for our vpc that acts as a firewall for controlling traffic in and out of one or more subnets
• Our vpc automatically comes with a modifiable default network acl.By default, it allows all inbound and outbound ipv4 traffic and if applicable,ipv6 traffic
• We can create a custom network,acl and associate it with a subnet
• By default each custom network acl denies all inbound and outbound traffic untill we add rules
• Each subnet in your vpc must be associated with a network acl.if we dont explicitly associate a subnet with a network acl,the subnet is automatically associated with the default network acl
• we can associate a network acl with multiple subnet,however a subnet can be associated with only one network acl at a time.When we associate a network acl with a subnet,the previous association is removed
• A network acl contains a numbered list of rules that we evaluate in order,starting with the lowest numbered rule
• The highest number that we can use for a rule is 32766.Recommended that we start by creating rules with rule numbered that are multiples of 100,so that we can insert new rules where you need later
• It functions at the subnet level
• NACl are stateless,outbound traffic for an allowed inbound traffic,must be explicitly allowed too
• We can have permit and deny rules in a NACL

Diff between Security Groups & NACL :

• Security group Operate at instance level and NACL oprates at subnet level
• SG support allows rules only and NACl permits allow as well as deny rules
• SG is stateful,return traffic is automatically allowed and NACl is stateless,return traffic must be explicitly allowed by rules
• SG applies to an instance only and NACl applies to all instances in this subnet

VPC Peering :

• A vpc peering connection is a networking connection between two vpc that enables us to route traffic between them using private ipv4 addresses or ipv6 addresses
• Instances in either vpc can communicate with each other as if they are within the same network
• We can create a vpc peering connection between our own vpc, or with a vpc in another aws account.The vpc can be in diff region
• Transitive peering is not possible i.e if vpc-A peers with vpc-B and vpc-B peers with Vpc-C,but by default vpc-A is not peered with vpc-C

LABS :

Session 5 : Creating VPC,Subnets,Route Table,IGW

Session 6 : Access Internet Inside Private Subnet Using NAT Gateway

Session 7 : Establishing VPC to VPC Peering and Testing connection between two VPC's

Session 8 : Establishing connection between 2 VPC's in 2 Diff region

Session 9 : Network ACL Inside VPC

Session 10 : VPC Endpoint

VPC Endpoint : A vpc endpoint enables us to privately connect our vpc to supported aws services.Instances in our vpc do not require public ip address to communicate with resources in the services
• Endpoint is a virtual device

Session 11 : VPN Connection

Session 12 : Configuring NAT instance for Private Subnets & Internet Access

• We can use a NAT instance in a public subnet in our VPC to enable instances in the private subnet to initiate outbound IPv4 traffic to the internet or other AWS services,but prevent the instances from receiving inbound traffic initiated by someone on the internet

Note : NAT is not supported for ipv6 traffic -use an egress only internet gateway

Session 13 : Virtual Private Gateway,Customer Gateway & Site-to-Site VPN connection

• By default,instances that we launch into an Amazon VPC cant communicate with our own(our corporate or home network) Network.To enable the communication we have to establish site to site VPN connection

VPN Connection : A secure connection between our on-premises equipment and our VPC's

VPN Tunnel : An encrypted link where data can pass from the customer network to or from AWS.Each VPN connection include two VPN tunnels which we can simultaneously use for high availability

Customer Gateway : An AWS resource which provides information to AWS about our customer gateway device

Customer Gateway Device : A physical or software app on customer side

STEPS :

- Create two VPC's-One in Mumbai and another in Singapore(customer end)
- Create one linux machine in both the VPC,take RDP of it(Security Group-SSH,TCP,ICMP)
- Now go to mumbai region --Create Virtual Private Gateway
- Now Create customer gateway---Enter Public IP of Singapore EC2 instance
- Create Site to site VPN Connection,Add subnet of customer end
- Now go to route tables ----Route Propagation
- Site to site VPN----Download Configuration
- Now go to singapore region take access of ec2 using putty
- Then do the following :

Commands :

GO TO PUTTY,CONNECT TO OUR EC2 LOGIN AS-ec2-user

1. Commands for Installation of Openswan
   i. Change to root user: $ sudo su
   ii. Install openswan: $ yum install openswan -y
   iii. In /etc/ipsec.conf
   uncomment following line if not already uncommented:
   include /etc/ipsec.d/*.conf
   iv. Update /etc/sysctl.conf to have following
   net.ipv4.ip_forward = 1
   net.ipv4.conf.all.accept_redirects = 0
   net.ipv4.conf.all.send_redirects = 0
   v. Restart network service: $ service network restart

2. Command for /etc/ipsec.d/aws-vpn.conf
   conn Tunnel1
   authby=secret
   auto=start
   left=%defaultroute
   leftid=Customer end Gateway VPN public IP
   right=AWS Virtual private gateway ID- public IP
   type=tunnel
   ikelifetime=8h
   keylife=1h
   phase2alg=aes128-sha1;modp1024
   ike=aes128-sha1;modp1024
   keyingtries=%forever
   keyexchange=ike
   leftsubnet=Customer end VPN CIDR
   rightsubnet=AWS end VPN CIDR
   dpddelay=10

dpdtimeout=30
dpdaction=restart_by_peer

3. Contents for /etc/ipsec.d/aws-vpn.secrets
customer_public_ip aws_vgw_public_ip: PSK "shared secret"

4. Commands to enable/start ipsec service
$ chkconfig ipsec on
$ service ipsec start
$ service ipsec status

END .....

Session 1: AWS Storage-Types of Storage

Types of Storage :

Simple Storage Service(S3) : Object Level Storage and access it from anywhere
Elastic File System(EFS) : Is only for Linux
Elastic Block Storage(EBS) : Block level storage which can only be accessed through EC2 to it is attached
Glacier : Is recently called as S3 Glacier, which mostly used to store data which is not most imp but we want to store for future use
Snowball : Uses for data migration consists of huge data, which is a portable device sent usually in trucks to data centers of company which wants to migrate to cloud

• AWS offers a complete range of storage services to support both application and archival compliance requirements.Select from objects,file and block storage services as well as cloud data migration options to start designing the foundation of our cloud IT Environment

Session 2 : Block Storage vs Object Storage

Block Storage :

• It is suitable for transactional databases,random read/write loads and structured database storage
• Block storage divide the data to be stored in evenly sized blocks(data chunks) for instance ,it can split into evenly sized blocks before it is stored
• Data blocks stored in block storage would not contain metadata(data created,data size,data modified,content type etc)
• Block storage only keeps the address(index) where the data blocks are stored,it does not care what is in that block.Just now how to retrive it when required
• EBS Volume : Can only be accessed through EC2

Object Storage :

• It stores the files as a whole and does not divide them
• In this,an object is the file/data itself,its metadata,object global unique id
• The Object global unique id is a unique identifier for the object(can be the object name itself) and it must be unique such that it can be retrived disregarding where its physical

storage location is
• Object storage cannot be mounted as a drive
• It is accessed from the Internet
• Ex : AWS S3,Dropbox etc

Session 3 : Simple Storage Service(S3),Naming Rules,Sub Resources

Simple Storage Service (S3) :

• S3 is a storage for the internet.it has a simple webservices interface for simple storing & retrieving of any amount of data,anytime from anywhere on the internet
• S3 is object based storage
• We cannot install OS on S3
• S3 has a distributed data-store architecture where objects are redundantly stored in multiple locations(Min 3 Location in same region)
• Data is stored in Buckets
• A bucket is a flat container of objects
• Max capacity of a bucket is 5TB
•We can create folder in our bucket(Available through Console)
• We cannot create nested buckets
• Bucket ownership is non-transferrable
• S3 bucket is region specific
• We can have upto 100 Buckets per account (may expand on request)

S3 Bucket Naming Rules :

• S3 bucket names(keys) are globally unique across all AWS regions
• Bucket names cannot be changed after they are created
• If a bucket is deleted,its name becomes available again to us or other accounts to use
• Bucket names must be atleast 3 and no more than 63 characters
• Bucket names are part of the URL used to access a bucket
• Bucket name must be a series of one or more lables(xyz.Bucket)
• Bucket names can contain lowercase,numbers and hypen(-).We cannot use uppercase letters
• Bucket name should not be an IP Address
• Each label must start and end with a lowercase letter or a number
• By default buckets and its objects are private
• By default only owner can access the bucket

S3-Bucket Subresources :

• This Includes :
1.Lifecycle -To decide an objects lifecycle management
2.Website -To hold configurations related to static website hosted S3 Bucket
3.Versioning -Keep object versions as it changes (Gets Updated) ….Can be enabled or suspended but can not be disabled after enabling
4.Access Control List : Bucket Policies

• The name is simply two parts-Bucket Region's endpoint and bucket name
Eg : S3 bucket named mybucket in europe west region is https://s3-eu-west1.amazonaws.com/mybucket

S3 Objects :

• An object size stored in an S3 bucket can be 0Byte to 5TB
• Each object is stored and retrived by unique key(ID or Name)
• An object in AWS S3 is uniquely identified and addressed through -service endpoint,bucket name,object key(name),optionally object versions
• objects stored in a S3 bucket in a region will never leave that region unless we specifically move them to another region or CRR
• A bucket owner can grant cross account permissions to another aws account(or users in another account) to upload objects
• We can grant S3 Bucket/object permissions to Individual users,AWS Account,Make the resource public,to all authentic users

Session 4 : S3Bucket Versionning,MFA Delete,S3 Multipart Upload,Copying S3 Objects

S3 Bucket Versioning :

• Bucket versioning is a S3 bucket sub-resource used to protect against accidental object/-data deletion or overwriten
• Versioning can also be used for data retention or archive
• Once we enable versioning on a bucket,it cannot be disabled however it can be suspended
• When enabled,bucket versioning will protect existing object and new object and maintains their versions as they are updated
• Updating objects refers to PUT,POST,COPy,Delete actions on objects
• When versioning is enabled and we try to delete an object a delete marker is placed on the object
• We can still view the object and the delete marker
• If we reconsider deleting the objects,we can delete the “Delete marker” and the object will be available again
• We will be charged for all S3 storage cost for all object versions stored
• We can use versioning with S3 lifecycle policies to delete older versions or we can move them to a cheaper S3 storage(Or Glacier)
• Bucket Version States : Enabled,Suspended,Un-versioned
• Versioning applies to all objects in a bucket and not partially applied
• Object existing before enabling versioning will have a version ID as ‘Null’
• If we have a bucket that is already versioned,then when we suspend versioning existing objects and their versions remain as it is
• However they will not be updated/versioned further with future updates while the bucket versioning is suspended
• New objects(uploaded after suspension), they will have a version ID as ‘Null’
• If the same key(name) is used to store another objects,it will override the existing one
• An object deletion in a suspended versioning buckets will only delete the objects with ID ‘Null’

S3 Bucket Versioning - MFA Delete :

• MFA delete is a versioning capacity that adds another layer of security in case our account is compromised
• This adds another layer of security for changing our buckets versioing state and permanently deleting an object version
• MFA delete requires our security credentials and the code displayed on an approved physical or software based authentication device

S3 Multipart Upload :

• Is used to upload an object in parts
• parts are uploaded independently and in parallel in any order
• It is recommended for object sizes 100Mb or more as min size is 5MB to multiport upload
• We must use it for objects larger than 5GB
• This is done through S3 multipart upload API

Copying S3 Objects :

• The copy operation creates a copy of an object that is already stored in Amazon S3
• We can create a copy of our object upto 5Gb in size in a single atomic operations
• However to copy an object greater than 5GB,we must use the multipart upload API
• Incur charges if copy to another region
• Use the copy operation to generate additional copies of the subject,renaming object(copy to a new name),Changing the copy's storage class or encrypt it at rest
• Move object across AWS location/region
• Change object metadata

Session 5 : Storage Classes of Amazon S3

Storage Classes of Amazon S3 :

1.S3-Standard(for normal and very frequent access),
2.S3 Glacier Deep Archive(Cheapest),
3.Amazon Glacier(Long Term Storage),
4.S3 Standard Infrequent Access(Less cost but we pay to access it more frequently),-Standard IA
5.S3 One-Zone-IA(Only stores1 copy with less cost),
6.S3 Intelligent Tiering(Automatically shifts data between standard, standard IA,glacier etc based on usage)
7.S3 Reduced Redundancy Storage (was removed)

Amazon S3 Standard :

• It offers high durability availability and performance object storage for frequently accessed data
• Durability is 99.999999999% (11 time 9)
• Resilient against events that impact entire AZ
• Designed for 99.99% availability over a given year
• Support SSL for data in-transit and encryption of data at rest
• Storage costs for the object is fairly high but there is very less charge for accessing the objects
• Largest object that can be uploaded in a single put is 5GB
• Backed wih the Amazon S3 service level Agreement for availability

Amazon S3-IA :

• S3-IA is for data that is accessed less frequently but required rapid access when needed
• The storage cost is much cheaper than S3-Standard,almost half the price.But we are charged more heavily for accessing our objects
• Durability is 99.999999999%
• Resilient against events that impact entire AZ
• Availability is 99.9% in year
• Support SSL for data in transit and encryption of data at rest
• Data that is deleted from S3-IA within 3 days will be charged for a full 30 days

• Backed wih the Amazon S3 service level Agreement for availability

Amazon S3 Intelligent Tiering :

• This storage class is designed to optimize cost by automatically moving data to the most cost effective access-tier
• It works by storing objects in two access tiers
• If an object in the infrequent access tier is accessed,it is automaticaly moved back to the frequent access tier
• There are no retrieval fess when using the S3-Intelligent tiering storage class and no additional tiering fee when objects are moved between access tiers
• Resilient against events that impact entire AZ
• Same low latency and high performance of S3-Standard
• Object less than 128Kb cannot move to IA
• Durability is 99.999999999%
• Availability is 99.9%
• Backed wih the Amazon S3 service level Agreement for availability

Amazon S3 One-Zone IA :

• It is for data that is accessed less frequently but require rapid access when needed
• Data stored in single AZ
• Ideal for those who want lower cost option of IA-data
• It is good choice for storing secondary backup copies of on-premise data or easily recreatable data
• We can use S3 lifecycle policies
• Durability is 99.999999999%
• Availability is 99.5%
• Because S3 one zone-IA stores data in single Az,data stored in this storage class will be lost in the event of AZ destruction
• Backed wih the Amazon S3 service level Agreement for availability

Amazon S3 Glacier :

• S3 glacier is a secure,durable,low cost storage class for data archiving
• To keep cost low yet suitable for varyig needs,S3 glacier provides three retrieval options that range from a few minutes to hours
• We can upload object directly to glacier or use lifecycle policies
• Durability is 99.999999999%
• data is resilient in the event of one entire AZ destruction
• Support SSL for data in transit and encryption of data at rest
• We can retrieve 10GB of our amazon S3 glacier data per month for free with free tier account
• Backed wih the Amazon S3 service level Agreement for availability

Amazon S3 Glacier Deep Archieve :

• It is amazon S3's cheapest storage
• Design to retain data for long period Eg : 10Years
• All objects stored in S3-Glacier deep archieve are replicated and stored across atleast at three geographically dispersed AZ
• Durability is 99.999999999%
• Ideal alternative to magnetic tape libraries
• Retrieval time within 12hours
• Storage cost is upto 75% less than for the existing S3 glacier storage class
• Availability is 99.9%
• Backed wih the Amazon S3 service level Agreement for availability

LABS :

Session 6 : S3 Bucket Creation

Session 7 : Bucket Creation using CLI

Session 8 : Versioning

Session 9 : Enable Cross Region Replication/Data Replication

Cross Region Replication :

• CRR enables automatic,asynchronous copying of objects across buckets in diff aws regions.Buckets configured for cross regions replication can be owned by the same aws account or by diff accounts
• CRR is enabled with a bucket level configuration.We add the replication configuration to our source bucket.
• In the min configuration,we provide the destination bucket,where we want amazon S3 to replicate objects and an aws IAM role that amazon s3 can assume to replicate objects on our behalf

When to use CRR :

• Comply with compliance requirements
• Minimize latency
• Increase operational efficiency
• Maintain object copies under diff ownership

Session 10 : AWS S3 Object Lifecycle Management

Session 11 : Elastic File System

Elastic File System :

• It is fully managedd service that makes it easy to set up,scale and cost-optimize file storage in the Amazon Cloud.
• With few clicks in the aws mgt. console, we can create file systems that are accessible to amazon ec2 instances via a file system interface(using standard OS file I/O API's) and support full file system access semantics (such as strong consistency and file locking)
• These file systems can automatically scale from gigabytes to petabytes of data without needing to provision storage.
• Tens,hundereds or even thousands of amzon ec2 instances can access an amazon efs file system at the same time and amazon efs provides consistent performance to each amazon ec2 instance
• Amazon efs is designed to be highly durable and highly available
• There are no min fee or setup costs,we pay only for what we use
• This is only for Linux machines
• Designed to provide performance for a broad spectrum of workloads and applications,including big data and analytics,media processing workflows,content mgt.,web serving and home directories
• This is not available in all regions

Session 12 : Hosting Static Website on S3

Session 13 : Elastic Block Storage(Diff. between EBS & Instance Store)

- Two types of block store devices are avaiabe for ec2 :

1.Elastic Block Storage : Persistent and Network attached drive

- Ebs voume behave like raw,unformatted external block storage devices that we can attach to our ec2 instance
- Ebs volumes are block storage devices suitable for database style data that require frequent read & write
- Ebs volumes are attached to our ec2 instances through the aws network,like Virtual hard drives
- An EBS volume can attach to a single ec2 instance only at a time
- Both ebs volumes and ec2 instance must be in the same AZ
- An ebs volume data is replicated by aws accross multiple servers in the same az to prevent data loss resulting from any single aws component failure

2.Instance Store Backed EC2 : Basically the virtual hard drive on the host allocated to this EC2 instance is limited to 10GB per device,ephemeral storage(non-persistent storage) and the ec2 instance can't be stopped(can only be rebooted) or terminated and terminate and stop will delete data

Session 14 : EBS Types-GP2,PIOPS,SSD,ST1,SC1 and Magnetic

EBS Volume Types :

1.SSD Backed Volume : General Purpose SSD(GP2) which is default volume and Provisioned IOPS SSD(io1)...These are bootable
2.HDD Backed Volume : Throughput Optimized HHD(st1) and cold HDD(sc1) .....These are non-bootable
3.Magnetic Standard : This is Bootable

General Purpose SSD(gp2) :

- GP2 is the default EBS volume type for the amazon ec2 instances and are backed by ssd
- GP,balances both price and performance
- Ratio of 3IOPS/GB with upto 10,000 IOPS(Input/Output per second)
- Boot volume having low latency
- Volume size 1Gb to 16TB
- Price : $0.10 per GB/Month

Provisioned IOPS SSD (io1) :

- These volumes are ideal for both IOPS intensive and throughput intensive workloads that require extremely low latency or for mission critical application
- Designed for I/O intensive applications such as large relational or Nosql Databases
- Use if we need more than 10,000 IOPS
- Can provision upto 32000 IOPS per volume and 64,000 IOPS for nitro based instances
- Volume Size : 4Gb to 16Tb
- Price : $0.125 per GB/month

Throughput Optimized HDD(st1):

• ST1 is backed by hard disk drives and is ideal for frequently accessed,throughput intensive workloads with large datasets
• ST1 volumes deliver performance in term of throughput,measured in MB/s
• Big data,data warehouse,log processing
• It cannot be a boot volume
• Can provisioned upto 500 IOPS per volume
• Volume Size : 500GB to 16TB
• Price : $0.045 per GB/month

Cold HDD(SC1) :

• SC1 is also backed by HDD and provides the lowest cost per GB of all EBS volume Types
• Lowest cost storage for infrequent access workloads
• Used in file servers
• Cannot be a boot volume
• Can provisioned upto 250 IOPS per volume
• Voluem Size : 500GB to 16TB
• Price : $0.025 per GB/month

Magnetic Standard :

• Lowest cost per GB of all EBS volume type that is bootable
• Magnetic volumes are ideal for workloads where data is accessed infrequently and application where the lowest storage cost is important
• Price : $0.05 per GB/month
• Volume Size : 1GB to 1TB
• Max IOPS/Volume : 40 to 200

Session 15 : EBS Snapshots of root volume and non-root volume

EBS Snapshots :

• Ebs Snapshots are point-in-time images/copies of our ebs volume
• Any data written to the volume after the snapshot process is initiated,will be included in the resulting snapshot(but wiill be included in future,incremental update)
• Per aws account,upto 5000 ebs volumes can be created
• Per account,upto 10,000 EBS snapshots can be created
• EBS snapshots are stored in S3,however we cannot access them directly,we can only access them through ec2 API's
• While EBS volumes are AZ specific,snapshots are region specific
• Any AZ in region can use snapshot to create EBS volume
• To migrate on EBS from one AZ to another,create a snapshot(region specific) and create an EBS volume from the snapshot in the intended AZ
• We can create a snapshot to an EBS volume of the same or larger size than the original volume size from which the snapshot was initially created
• We can take a snapshot of a non-root ebs volume while the volume is in use of a running ec2 instance
• This means we can still access it while the snapshot is being processed
• However the snapshot will only include data that is already written to our volume
• The snapshot is created immediately but it may stay in pending status until the full snapshot is completed.This may take few hours to complete specifically for the first time snapshot of a volume

• During the period,when the snapshot status is pending,we can still access the volume(non-root),but I/O might be slower because of the snapshot activity
• While in pending state,an in-progress snapshot will not include data from ongoing reads and writes to the volume
• To take complete snapshot of our non-root EBS volume : stop or unmount the volume
• To create a snapshot for a root ebs volume, we must stop the instance first then take the snapshot

Session 16 : Incremental Snapshots

Incrementa Snapshot:

• EBS snapshots are stored incrementally
• For low cost storage on s3,and a guarantee to be able to fully restore data from the snapshots
• What we need is a single snapshot that further snapshot will only carry the changed blocks(incremental updates)
• Therefore we do not need to have multiple full/complete copies of the snapshot
• We are charged for : data transferred to S3 from our ebs volume we are taking snapshots
• Snapshots stored in s3
• First snapshot is a clone,subsequent snapshotsa are incremental
• Deleting snapshot will only remove data exclusive to that snapshot

Session 17 : Encryption of EBS Volume

EBS Encyption :

• EBS encryption is supported on all ebs volume types and all ec2 instance familes
• Snapshots of encrypted volumes are also encrypted
• Creating an ebs volumes from an encrypted snapshot will result in an encrypted volume
• Data encryption at rest means encrypting data while it is stored on the data storage device

• There are many ways we can encrypt data on an ebs volume at rest,while the volume is attached to an ec2 instance :

1. using 3rd party encryption technique on ebs volume,
2. encryption tools,
3.using encrypted ebs volumes
4.using encryption at the OS Level
5.Encrypt data at the application level before storing it to the volume
6.Using encrypted file system on the top of the ebs volume
• Encrypted volumes are accessed exactly like unencrypted ones,basically encryption is handled transparently
• We can attach an encrypted and unencrypted volumes to the same ec2 instance
• Remember that the ebs volumes are not physically attached to the ec2 instance,rather they are virtually attached through the ebs infrastructure
• This means when we encrypt data ona an ebs volume,data is actually encrypted on the ec2 instance then transferred,encrypted to be stored on the ebs volume
• This means data in transit between ec2 and encrypted ebs volume in also encrypted
• There is no direct way to change the encryption state of the volume

• To change the state(indirectly) we need to follw either of the followig two ways :

1.Attach a new,encrypted ebs volume to the ec2 instance that has the data to be encrypted then:

☐ Mount the new volume to the ec2 instance
☐ Copy the data from the unencrypted volume to the new volume
☐ Both volumes must be on the same ec2 instance

2.Create a snapshot of the encrypted volume

☐ Copy the snapshot and choose encryption for the new copy,this will create an encrypted copy of the snapshot
☐ Use this new copy to create on ebs volume which will be encrypted too
☐ Attach the new encrypted ebs volume to the ec2 instance

Root EBS volume Encryption :

• There is no direct way to change the encryption state of a volume
• There is an indirect workaround to this
☐ Launch the instance with ebs volume required
☐ Do whatever patching or install applications
☐ Create an AMI from the ec2 instance
☐ Copy the AMI from the EC2 instance
☐ Copy the AMI and choose encryption while copying
☐ This results it an encrypted AMI that is private
☐ Use the encrypted AMI to launch new ec2 instances which will have these ebs root volume encrypted

Session 18 : Sharing EBS Snapshots

EBS Encryption Key :

• To encrypt a volume or a snapshot we need an encryption key,these keys are called customer masters key(CMK) and are managed by aws key mgt. service(KMS)
• When encrypting the first ebs volume,aws kms creates a default cmk key
• This key is used for our first volume encryption.Encryption of snapshots created from this volumes and subsequent volumes created from these snapshots
• After that each newly encrypted volume is encrypted with a unique/seperate AES-256 bit encryption key.
• This key is used to encrypt the volume,its snapshots and any volumes created of the snapshots

Changing Encryption Key :

• We cannot change the encryption(CMK) key used to encrypt an existing encrypted snapshot or encrypt EBS volume
• If we want to change the key,create a copy of the snapshot and specify,during the copy process,that we want to re-encrypt the copy with a diff. key
• This comes in handy when we have a snapshot that was encrypted using our default CMK key and we wnat to change the key in order to be able to share the snapshot with other accounts

Sharing EBS Snapshot :

• By default,only the account owner can create volumes from the account snapshot
• We can share our unencrypted snapshots with the aws community by making them public
• Also we can share our unencrypted snapshot with a selected aws accounts by making them private then selecting aws accounts to share with
• We can not make our encrypted snapshot public
• We cannot make a snapshot of an encrypted ebs volume public on aws

• We can share our encrypted snapshot with specific aws accounts as follows:
☐ Make sure that we use a non default/custom cmk key to encrypt the snapshot not the default cmk key(aws will not allow the sharing if default cmk key is used)
☐ configure cross account permissions in order to give the accounts with which we want to share the snapshot,access to the custom cmk key used to encrypt the snapshot
☐ Without this the other accounts will not be able to copy the snapshot,nor will be able to create volumes of snapshots
• aws will not allow us to share snapshots encrypted using our default cmk key
• For the aws accounts with whom an encrypted snapshot is shared :
☐ They must first create their own copies of the snapshot
☐ Then they use that copy to restore/create ebs volume
• We can only make a copy of the snapshot when it has been fully saved to S3(its status show as complete) and not during the snapshots pending status(when data blocks are being moved to S3)
• Amazon S3 server side encryption(SSE) protect the snapshot data-in-transit while copying
• We can have upto 5 snapshots copy request running in a single destination per account

LABS :

Session 19 : Creating AMI and recreating EC2 in another region with that AMI

Session 20 : Copy AMI into another AWS Account and recreate EC2

Session 21 : Attach root volume with another EC2 Instance

Session 1 : Autoscaling(How it works & Components)

AutoScaling in AWS :

• Creating group of ec2 instances that can scale up or down depending on conditions we set
• Scale out means increasing and Scale in means reducing
• Enable elasticity by scaling horizontially(same type and size scaling) through adding or

terminating ec2 instances
• Autoscaling ensures that we have the right number of aws ec2 instances for our needs at all time
• Autoscaling helps us save cost by cutting down the number of ec2 instances when not needed and scaling out to add more instances only when it is required

Autoscaling Components :

1.Launch Configuration : Like instance type,AMI,keypair,Security Group
2.Autoscaling Group : Group name,Group Size,VPC,Subnet,Health Check Period
3.Scaling Policy : Metric type(Like Cpu Utilization),Target value

Session 2 : Autoscaling(Balancing,Attach & Detach)

Balancing :

• If autoscaling finds that the number of ec2 instances launched by ASG into subject AZ's is not balanced(EC2 instances are ot not evenly distributed).Autoscaling do rebalancing activity by itself
• Autoscaling(AS) tries to balance the instances distribution across AZ's
• While rebalancing,ASG launches new ec2 instances where there are less ec2 at present,and then terminates the instances from the AZ,that had more instances

What causes Imbalancing of EC2 :

☐ If we add or remove same subnet/AZ from ASG
☐ If we manually request for ec2 termination from our ASG
☐ An AZ that did not have enough ec2 capacity now has enough capacity and it is one of our ASG AZ

Attaching :

• We can attach a ruuning ec2 instances to an autoscaling group by using aws console or CLI,if the below conditions are met -
☐ Instances must be in running state(not stopped or terminated)
☐ AMI used to lauch the ec2 still exist
☐ Instance is not part of another ASG
☐ Instance is in the same AZ of the same group
☐ If the existing ec2 instances under the autoscaling group,plus the one to be needed,exceed the max capacity of the autoscaling group,the request will fail,ec2 instance would not be added

Detach :

• We can remove ec2 instances from an ASG using aws console or CLI
• We can then manage the detached instance independently or attach it to another ASG
• When we detach an instance,we have the option to decrement the ASG desired capacity
• If we do not,the autoscaling group will launch another instances to replace the one detached
• When we delete an ASG,its parameters like max,min and desired capacity are all set to 0.Hence,it terminate all its ec2 instances
• If we want to keep the ec2 istances and manage them independently,we can manually detach them first,then delete the ASG

How load balancer work with ASG:

• We can attach one or more elastic load balancer to our autoscaling group
• The elastic load balancer must be in the same region as the ASG
• Once we do this,any ec2 instance existing or added to the ASG will be automatically registered with the ASG defined ELB
• We do not need to register those instances manually on the ASG defined ELB
• Instances and The ELB must be in the same VPC

Health Check :

• ASG classifies its ec2 isntances health status as either healthy or unhealthy
• By default,AS uses ec2 status checks only to determine the health status of an instance
• When we have one or more ELB defined with the ASG,we can configure AS to use both the EC2 health check and the ELB health check to determine the instances health check
• Health check grace period is 300 sec by default
• If we set zero in grace period,the instance health is checked once it is in service
• Untill the grace period timer expires, any unhealthy status reported by ec2 status checks,or the ELB attached to the autoscalling group,will not be acted upon

• After grace period expires,ASG would consider an instances unhealthy in any of the following cases :
☐ EC2 status check report to ASG an innstance status other than running
☐ If ELB health check are configured to be used by the AS,the if the ELB report the instances as ‘out of service’
☐ Unlike AZ rebalancing,termination of unhealthy instances happen first,then AS attempt to launch new instance to replace the ones terminated
☐ Elastic IP and EBS volumes gets detached from the terminated instances,we need to manually attach them to the new instances

Session 3 : Types of AWS Autoscaling

On four situation,ASG sends a SNS email Notification :
1.An instance is launched
2.An instance is terminated
3.An instance fails to launch
4.An instance fails to terminate

Merging Autoscaling Groups :

• Can only be done from the CLI(not aws console)
• We can merge multiple,single AZ ASG into single,one multi-AZ ASG
• Scale out means launching more ec2 instances
• Scale in means terminating one or more instances by scaling policy
• It is always recommended to create a scale-in event for each scale-out event we create
• Launch config cant be edited,only copied or deleted

Monitoring :

• AWS ec2 services sends ec2 metrics to cloudwatch about the ASG instances :
☐ Basic monitoring(Every 300 secs enabled by default & free of cost)
☐ We can enable detailed(every 60 sec-chargeable)
• When the launch config is done by AWS cli,detailed monitoring for ec2 instances is enabled by default

StandBY state :

• We can manually move an instance from an ASG and put it in standby state
• Instances in standby state are still managed by autoscaling
• Instances in standby state are charged as normal in service instances
• They do not count towards available ec2 instances for workload/APP use
• AS does not perform health check instances in standby state

Auto-Scaling Policy : Manual or Dynamic

• Define how much we want to scale based on defined conditions
• ASg uses alarms and polices to determine scaling
• For simple or step scaling, a scaling adjustment can't change the capacity of the group above the max group size or below the min group size

Predictive/Scheduled Scaling : It looks at historic pattern and forecast them into the future to schedule change in the no of ec2 instances.It uses machine learning model to forecast daily and weekly pattern

Target Tracking Policies : Increase or decrease the current capacity of the group based on a target value for a specific metric.This is similar to the way that our thermostat maintain the temp of our home

Step Scaling : Increase or decrease the current capacity of the group based on a set of scaling adjustment,known as step adjustment,that vary based on the size of the alarm reach
• Does not support/wait for a cool-down times
• Support a warm-up timer time taken by newly launch instance to be ready and contribute to the watched metric

Simple Scaling : Single adjustment(up or down) in response to an alarm (cooldown timer)-300 sec

Scheduled Scaling :

• Use for predictable change
• We need to configure a schedule action for a scale out at a specific date/time and to a required capacity
• A schedule action must have a unique date/time
• We cannot configure two scheduled activities at the same time/date

LAB :

Session 4 : AWS Autoscaling Demo

Session 1 : Load Balancer --Application,Network,Classic Load Balancers

Load Balancer : LB distributes the web traffic to the available server (or) LB refers to efficient distributing of incoming traffic across a group of backend servers

LB Types : APP LB,Network LB,Classic LB

- An internet facing load balancer has a publically resolvable DNS name
- Domain names for content on the ec2 instances served by the elb,is resolved by the internet dns server to the elb dns name(and hence ip addr)
- This is how traffic from the internet is directed to the ELB front end
- Classic load balancer services support http,https,tcp,ssl
- Protocol ports supported are 1-65535
- It support ipv4,ipv6 and dual stack
- App LB distributes incoming app traffic across multiple targets such as ec2 instances in multiple AZ
- This increases the availability of our application
- Network LB has ability to handle volatile workloads and scale to millions of requests per second

Session 2 : ELB Listener

ELB Listener :

- An ELB listener is the process that checks for connection request
- We can configure the protocol/port number on which our ELB listener listen for connection request
- Frontend listeners check for traffic from client to the listener
- Backend listeners are configured with protocol/port to check for traffic from the ELB to the EC2 instances
- It may take sometime for the registration of the ec2 instances under the ELB to complete
- Registered ec2 instances are those that are defined under the ELB
- ELB has nothing to do with the outbound traffic that is initiated/generated from the registered ec2 instances destined to the internet or to any other instances within the VPC
- ELB only has to do with imbound traffic destined to the ec2 registered instances(as the destination) and the respective return traffic
- We start to be charged hourly(also for partial hours) once our ELB is active
- If we do not want to be charged as we do not need ELB anymore,we can delete it
- Before we delete the ELB,it is recommended that we point the route 53 to somewhere else other than the ELB
- Deleting the ELB does not effect or delete the ec2 instance registered with it
- ELB forwards traffic to eth0 of our registered insatnce
- In case the ec2 registered instances has multiple ip address on eth0,elb will route the traffic to its primary IP address

Session 3 : How LB finds Unhealthy Instances

- ELB supports IPv4 address only in VPC
- To ensure that the ELB service can scale elb nodes in each AZ,ensure that the subnet defined for the LB is at least /27 in a size, and has atleast 8 available IP addresses for the ELB nodes an use to scale
- For fault tolerance,it is recommended that we distribute our registered ec2 instances across multiple AZ, within the VPC region

• If possible,try to allocate same number of registered instances in each AZ
• The LB also monitors the health of its reg. instances and ensure that it routes traffic only to healthy instances
• A healthy instances show as ‘healthy’ under ELB
• When the ELB detects an unhealthy instance it stop routing traffic to the instance
• An unhealthy instance shows ‘unhealthy’ under the ELB
• By default aws console uses ping http(port80) for health check
• Registered instances must respond with a http 200 Ok message within the timeout period else it will be considered as Unhealthy
• AWS API uses ping TCP(port 80) for health check
• Response timeout is 5 sec (range is 2-60sec)

• *Health check interval :* Period of time between health checks -default 30 sec(range from 5 to 300 sec)
• *Unhealthy Threshold :* Number of consecutive failed health check that should occur before the instance is declared unhealthy(range is 2-10) -Default 2
• *Healthy Threshold :* Number of consecutive successful health check that must occur before the instance considered unhealthy(range 2-10)-Default -10

Session 4 : ELB Listener and Target Groups

• By default the elb distributes traffic evenly between the AZ it is defined in without consideration to the number of registered ec2 instances in each AZ

Cross Zone LB :

• Disabled by default
• When enabled,the elb will distribute traffic evenly between registered ec2 instances
• If we have 7 ec2 instances in One AZ and 3 in another AZ,and we enabled cross zone LB,each registered ec2 instances will get around the same amount of traffic load from the ELB
• ELB name we choose must be unique within the account
• ELB is region specific.So all registered ec2 instances must be in same region,but can be in diff AZ
• To define our elb in an AZ,we can select one subnet in that AZ.Subnet can be public or private
• Only one subnet can be defined for the ELB in an AZ
• If we try to select another one in the same AZ,it will replace the former one
• If we register instance in an AZ with elb,But do not define a subnet in that AZ for the ELB.These instances will not receive traffic from ELB
• ELB should always be accessed using DNS and not IP

• An ELB can be Internet Facing or Internal Facing :

Internet Facing : ELB nodes will have public IP address
• DNS will resolve the ELB DNS name to these IP addresses
• It routes traffic to the private IP address of our instances with private IP
• We need an public subnet in each AZ where the internet facing ELB will be defined,such that the ELB will be able to route Internet Traffic
• Format of the public ELB Dns name of internet facing ELB : name-1234567890region.elb. amazonaws.com

Internal Facing : internal-name 1234567890region.elb. amazonaws.com

ELB Listener :

• An elb listener is the process that checks for connection request
• Each Network LB needs at least one listener to accept traffic
• We must assign a security group to our ELB.This will control traffic that can reach our ELB front end listeners

Target Group : Logical grouping of targets behind the load balancer
• TG can exist independently from the load balancer
• TG can be associated with an autoscaling group
• TG can contain upto 200 Targets

LABS :

Session 5 : Application Load Balancer

Session 6 : Network Load Balancer

Session 7 : Establishing Load Balancer between 2 VPC's

Session 1 : Identity and Access Management Introduction

Identity & Access Management :

• IAM refers to a framework or policies and technologies for ensuring that the proper people in an organisation have the appropriate access to technology resources
(OR)
AWS IAM is a web service that helps us securely control access to AWS resources.We use IAM to conrtol who is authenticated(signed-in) and authorized(has permission) to use resources

• When we first create an AWS account,we begin with a single signin identity that has complete access to all AWS services and resources in the account
• This identity is called,the AWS account ‘Root User' and is accessed by signing in with the email address and password that we used to create the account
• AWS strongly recommends that we do not use the root user for everyday tasks,even the administrative ones
• Use other IAM user accounts to manage the administrative task of our account for securely lock away the root user credentials and use them to perform only a few account and service management tasks
• IAM user limit is 5000 for AWS account.We can add upto 10 users at one time
• We are also limited to 300 groups per AWS account
• We are limited to 1000 IAM roles under AWS account
• Default limits of managed policies attached to an IAM role and IAM user is 10
• IAM user can be a member of 10 groups
• We can assign two access keys(MAX) to an IAM user

Features of IAM :

1.Shared access to our AWS Account :
• We can grant other people permission to administer and use resources in our aws account without having to share our access credentials(password or access key)

2.Granular Permissions:
• We can grant diff permission to diff people for diff resources
• For instance,we can allow same users complete access to EC2,S3,Dynamodb,Redshift while for others,we can allow read only access to just some S3 buckets,or permission to administer just some ec2 instances or to access our billing information but nothing else

3.Secure access to AWS resources for applications that run on Amazon EC2 :
• We can use IAM features to securely give applications that run on ec2 instances the credentials that they need in order to access other aws resources.
• EX : Includes S3 buckets and RDS or DynamoDb databases

4.Multifactor Authentication(MFA) :
• We can add two factor authentication to our account and to individual users to extra security.We can use physical hardware or virtual MFA(Eg : Goog;le Authenticator)

5.Identity Federation :
• We can allow users who already have passwords elsewhere
• EX : In our corporate network or with an internet identity provider to get temporary access to our AWS account

6.Identity information for Assurance :
• If we use AWS cloudtrail,we receive log records that include information about those who made request for resources in our account.That information is based on IAM identities

7.PCI-DSS Compliance :
• IAM support the processing,storage and transmission of credit card by a merchant or service provider,and has been validated as being compliant with payment card Industries(PCI) and Data Security Standard(DSS)

8.Eventually Consistent:
• If a request to change some data is sucessful, the change is comitted and safely stored.However the change must be replicated across IAM,which can take sometime
• IAM achieves high availabilty by replicating data across multiple servers within AWS data centre around the world

• Free to use : AWS IAM is a feature of our AWS account offered at no additional cost
• We will be charged only for the use of other AWS products by our IAM users

Session 2 : IAM Terms-Principal,Request,Authentication,Authorization

Principal :

• A principal is a person or application that can make a request for an action or operation on an AWS resources
• Our administrative IAM user is our first principal
• We can allow users and services to assume a role
• IAM users,roles,federated users and applications are all AWS principals
• We can support federated users or programmetic access to allow an application to access our AWS account

Request :

• When a principal tries to use the AWS resources management console,that aws api,or aws cli,that principal sends a request to aws.

• The request include the following information : Actions,Resources,Principal,Environment data,Resources Data

1.Actions : That the principal want to perform
2. Resources : Upon which the actions are performed
3.Principal Information : including the environment from which the request was made
4.Request Context : Before AWS can evaluate and authorize a request ,aws gathers the request information and principal(the requester),which is determined based on the authorization data.This includes the aggregate permissions that is associated with that principal
5.Environment Data : Such as IP address,user agent,SSL enabled status or the time of day
6.Resource Data : Data related to the resource that is being requested

Authentication :

• A principal sending a request must be authenticated(signed in to aws) to send a request to AWS
• Some aws services like AWS S3 allow request from anonymous users,they are exception to the role
• To authenticate from the console as a root user,we must signin with our user name and password
• To authenticate from the API to CLI,we must provide our access key and secret key
• We might also be required to provide additional security information like MFA

Authorization :

• To authorize request,IAM allows values from the request context to check for matching policies and determie whether to allow or deny the request
• IAM policies are stored in IAM as JSON documents and specify the permissions that are allowed or denied
• *User(identity) based policies :* Specify permission allowed/denied for principal
• *Resource based Policies :* Specify permission allowed/denied for resources .Popular for granting cross account permissions
• IAM checks each policy that matches the context of our request
• If a single policy includes a denied action,IAM denies the entire request and stop evaluating.This is called Explicit Deny
• The evaluation logic follows these rules :
☐ By default,all requests are denied(implicit deny)
☐ An explict allow overrides this default
☐ An explicit deny overrides any allows

Session 3 : IAM Terms-Actions,Resources

• We can create a new IAM policy in the AWS management console using one of the following ways :
1.JSON : We can create our own JSON syntax
2.Visual Editor : We can constuct a new policy from scratch in the visual editor.If we use the Visuual editor,we do not have to understood JSON syntax
3.Import : We can import a managed policy within our aws account and then edit the policy to customize it to our specific requirements

Actions/Operations :

• Actions are defined by a service and are the things that we can do to a resource.Such as viewing,creating,editing and deleting that resources
• IAM supports approx 40 actions for a user resource including create user,del user etc
• Any actions or resources that are not explicitly allowed are denied by default
• After our request has been authenticated and authorized ,AWS approves the actions in our request

Resources :

• A resource is an entity that exist within a service
• Examples are ec2 instances,S3 bucket,IAM user,DynamoDB table
• Each AWS service defines a set of actions that can be performed on each resource
• After AWS approves the actions on our request,these actions can be performed on the related resources within our account
• If we create a request to perform an unrelated action on a resource,that request is denied
• When we provide permissions using an identity based policy in IAM,then we provide permissions to access resources only within the same account

Session 4 : Users,Roles & Groups

Identity Federation :

• If our account users already has a way to be authenticated such as authentication through our corporate network
• We can federate those user identities into AWS
• A user who has already logged to the corporate using their corporate identity
• The corporate can replace their existing identity with a temporary identity in our AWS account
• This user can work in the AWS management console
• Similarly,an application that the user is working with can make a programmatic request using permissions that we define

Federation is particularly useful in these cases :

1.Our users already have identities in a corporate directory
• If our corporate directory is compatible with security assertion markup language(2.0)
☐ We can configure our corporate directory to provide single sign on(SSO) access to the AWS mgt. console for our user
• If our corporate identity is not compatible with SAML 2.o
☐ We can create Identity broker app to provide single sign on access to the AWS mgt. console for our users
• If our Corporate directory is Microsoft Active directory,we can use AWS directory service to establish trust between our corporate directory and our AWS account

2.Our users already have internet identities :
• If we are creating a mobile app or web based app that can let users identity themselves through an Internet Identity Provider like login with facebook,google,amazon or any open ID connect(OIDC) compatible identity provider,the app can use WEB federation to access AWS
• AWS recommends to use AWS console for Identity Federation

IAM Users and SSO :

• IAM users in our account have access only to the AWS resources that we specify in the

policy that is attached to the user or to an IAM group that the user belongs to
• To work in the console,user must have permissions to perform the actions that the console performs,such as listing and creating AWS resources

IAM Identities :

• IAM identities is what we create under our AWS account to provide authentication for people,application and processes in our AWS account
• IAM Identities represent the user and can be authenticated and then authorized to perform actions in AWS
• Each of these can be associated with one or more policies to determine what actions a user,role or member of a group can do with which resources and under what conditions
• IAM group is a collection of IAM users
• IAM role is very similar to IAM user

IAM Users :

• An IAM user is an entity that we create in AWS.It represent the person or service which uses the IAM user to interact with AWS
• We can create 10 users at a time
• An IAM user can represent an actual person or an application that requires AWS access to perform actions on AWS resources
• A primary use for IAM users is to gives people the ability to signin to the AWS mgt. console for interactive task and to make programmatic request to AWS services using the API or CLI

For any User,we can assign them :

• A username and password to access the AWS console
• An access key id and secret key that they can use for programmatic access
• The newly created IAM users have no password and no access key.we need to create the User request
• Each IAM user is associated with one and only one AWS account
• Users are defied within our account,so users do not have to do payment.Bill would be pay by the parent account

IAM Groups :

• An IAM group is a collection of IAM users
• It is a way to assign permissions/policies to multiple users at once
• Use groups to specify permissions for a collection of users,which can make those permissions easier to manage for those users
• Ex : We could have a group called HR and give that group the types of permissions that HR department typically needs
☐ Any user in that group automatically has the permission that are assigned to the group
☐ If a new user joins our organisation and should have HR previliges,we can assign the appropriate permissions by adding the user to that group
☐ If a person changes job in our organization instead of editing that users permissions,we can remove him or her from the old groups and add him or her to the appropriate new groups

IAM Group Limitations :

• A group is not an identity in IAM because it cannot be identified as a principal in a permission policy
• Groups cant be nested
• We have a limit of 300 groups in an AWS account

• A user can be a member of upto 10 IAM groups

IAM Roles :

• An IAM role is very similar to a user in that it is an identity with permission policies that determine what the identity can and cannot do in AWS
• An IAM role does not have any credentials associated with it
• Instead of being uniquely associated with one person , a role is intended to be assumable by anyone who needs it
• An IAM user can assume a role to temporary take an diff permissions for a specific task
• An IAM role can be assigned to a federated user who signs in by using an external identity provider instead of IAM

IAM temporary Credentials :

• Temporary credentials are primarily used with IAM roles,but there are also other uses
• We can request temporary credentials that have a more restricted set of permissions than our standard IAM users
• This pervent us from accedentially performing task that are not permitted by the more restricted credentials
• A benefit of temporary credentials is that they expire automatically after a set period of time

Session 5 : Permissions & Policies,Federation users,Resource based policies,Root and IAM user

Permissions

• The access mgt portion of AWS identity and access mgt helps us to define what a user or other entity is allowed to do in an account,often referred to as authorization
• Permissions are granted through policies that are created and then attached to users,groups or roles

Policies and Users :

• By default,IAM users cant access anything in our account
• We grant permissions to a user by creating a policy which is a document that defines the effect,actions resource and optional conditions
• Any actions or rsources that are not explicitly allowed are denied by default

IAM Multiple Policies :

• Users or groups can have multiple policies attached to them that grant diff permissions
• in the case of multiple policies attached to a user(or a group),The users's permissions are calculated based on the combination of policies

Federation users and roles :

• Federation user dont have permanent identities in our AWS account the way that IAM users do
• To assign permissions to federated users we can create an entity referred to as a role and define permission for the role
• When a federated user sign in to AWS, the user is associated with the role and is granted the permission that are defined in the role

Resources Based Policies :

• In some cases(Like S3 Bucket),we can attach a policy to a resources in addition to attaching it to a group or user.This is called a resource based policy
• A resource based policy contains slightly different information than a user based policy
☐ In a resource based policy we specify what actions are permitted and what resources is affected
☐ We can explicitly list who is allowed access to the resource(a principal)
☐ Resource based policies include a principal element that specifies who is granted the permissions

IAM user-The root user

• When we first create an AWS account,we create an account(or root user) identity which we use to signin to AWS
• The account root user credentials are the email address used to create the account and a password,which can be used to signin to the AWS mgt console as the root user
• When we signin as the root user,we have complete,unrestricted access to all resources in our AWS account,including access to our billing information and the ability to change our password
• This level of access is necessary when we initially set up the account
• It is not possible to restrict the permissions that are granted to the AWS account

AWS recommends that :

• AWS recommends that we dont use root user credentials for everyday access
• Also AWS recommends that we do not share our root user credentials with anyone,because doing so gives them unrestricted access to our account
• Create an IAM user for ourself and then assign ourself administrative permission for our account
• We can then signin as that user to add more users as needed
• An IAM user with administrator permissions is not the same things as the AWS account root user

IAM Users :

• An IAM user is an entity that we create in AWS.It represents the person or service who uses the IAM user to interest with AWS
• An IAM user can represent an actual person or an application that requires AWS access to perform action on AWS resources
• IAM users are global entities,like an AWS account is today.No region is required to be specified when we define user permissions.Users can use AWS services in any geographic region

For any user we can assign them :

• A username and password to access the AWS console
• An access key(access key id and secret key) that they can use for programmatic access(issuing request) to our AWS services using API and CLI
☐ We assign either or both based on the user activities and needs
• We can view and download our secret access key only when we create the access key
• We cannot view or recover a secret access key later
• If we lose our secret access key.We can create a new access key
• Each IAM user is associated with one AWS account

By default a new IAM user :

• A new IAM user has no permission to do anything
• Has no password and no access key(neither an access key ID nor a secret access key).It means no credentials of any kind
• We must create the type of credentials for an IAM users based on what the user will be doing
• We can grant user permissions by attaching IAM policies to them directly or making them members of IAM group where they inherit the group policies/permissions
• We can have upto 5000 users per AWS account

Session 6 : IAM Role, Ways to use a role,IAM role vs Resource based policy,IAM role delegation,Cross Account Permission

IAM Role :

• An IAM role,is a set of permissions that grant access to actions and resources in AWS
• These permissions are attached to the role,not to an IAM user or Group.Instead of being uniquely associated with one person a role is intended to be assumable by anyone who needs it
• A role does not have standard long-term credentials associated with it
• If a user assumes role temporary security credentials are created dynamically and provided to the user

Following entities can use Role :

1. An IAM user in the same AWS account
2.An IAM user in a different AWS account
3.A Webservice offered by AWS such as Amazon Elastic Compute Cloud

There are two ways to use a role :

1.Interactively in the IAM console

• IAM users in our account,using the IAM console can switch to a role to temporarily use the permissions of the role in the console
• The user give up their original permission and take on the permission assigned to the role
• When the user exists the role,their original permissions are restored

2.Programmatically with the AWS CLI,tools for Windows powershell or API

• An application or a service offered by AWS(like amazon ec2) can assume a role by requesting temporary security credentials for a role with which to make programmatic request to AWS
• We use a role this way so that we dont have to share or maintain long-term security credentials for each entity that requires access to a resource

Difference between IAM role and Resource based Policy

• Unlike a user-based policy a resource based policy specifies who can access that resources
• Cross account access with a resource based policy has an advantage over a role with a resource that is accessed through a resource based policy,the user still works in the trusted account and does not have to give up his or her user permissions in place of the role permissions
• In other words,the user continues to have access to resources in the trusted account at

the same time as he or she has access to the resource in the trusting account
• This is useful for task such as copying information to or from the shared resource in the other account
• Note that not all services support resource based policy
• The following services support resource-based policy :
1.Amazon S3
2.Amazon SNS
3.Amazon SQS
4.Amazon Glacier Vault

IAM role delegation :

• Delegation is the granting of permissions to someone to allow access to resources that we control
• Delegation involves setting up a trust between the account that owns the resource(the trusting account) and the account that contains the users that need to access the resource(the trusted account)
• The trusted and trusting accounts can be any of the following :
1.The same account
2.Two accounts that are both under our organizations control
3.Two accounts owned by diff organization

• To delegate permission to access a resource,we create an IAM role that has two policies attached :
1.The trust policy
2.The permission policy
• The trusted entity is included in the policy as the principal element in the documennt
• When we create a trust policy,we cannot specify a wildcard(*) as a principal

Cross Account Permission :

• We might need to allow users from another aws account to access resources in our aws account.If so,dont share security credentials,such as access keys between accounts.Instead use IAM roles
• We can define a role in the trusting account that specifies what permissions the IAM user in the other account are allowed
• We can also designate which aws account have the IAM users that are allowed to assume the role.We do not define users here,rather AWS accounts

Role for Cross-Account Access :

• Granting access to resources in one account to a trusted principal in a diff account
• Roles are the primary way to grant cross-account access
• However with some of the webservices offered by AWS,we can attach a policy directly to a resource,these are called resource based policies.We can use them to grant principals in another AWS account access to the resource

LABS :

Session 7 : Creating IAM Users

Session 8 : Creating Groups & Inline Policy

Session 9: Cross Account Access using IAM role

• Login into 1st AWS account
• Create one group and then create two iam users in it
• Attach policy to group(EC2 Readonly access)
• Login to 2nd AWS account
• Create one S3 Bucket
• Create a role-Select Another AWS account-Insert Account ID of 1st AWS account-Attach policy -S3Readonlyaccess-Role Name-S3 read
• Now login back to 1st aws account click on Group-Permission-Inline Policy-Select-AWS security Token service-Assume Role
• Click on ‘ADD Statement’-Apply policy
• Now login as IAM user1
• Switch role-Now check,whether we are able to see the bucket of another account or not
• Repeat above 2 steps but with user 2
• Now login into 2nd AWS account-Role-Trusted Relationship-edit-paste arn of user1-update policy
• Now login again in user1-switch role-test s3 bucket
• Now login as user2-switch role-we will get error

Session 10 : Connecting Windows AD Server to AWS

• Create One Win server 2019 Base
• Change its password and then Install AD DS in ADD role & Features
• Now,create One forest i.e “tg.com”
• Now, go to server manager-Tools-DNS-Reverse Lookup Zone
• Now-DNS-Forward Lookup Zone-Enable Update Associated Pointer-Apply
• Now in Ethernet setting-Enter private IP of Server in Preferred DNS Server
• Now click on Server manager-Tools-AD USers and Computers-Create two users ‘A & B’ -Give Password-India@123
• Now go to AWS Mgt console-search Directory Services-AD Connector
• Directory DNS name-Darkrose-in-DNS IP address-Private Ip of AD server-usernmae-Administrator-Password-Same as AS server Password
• Go to IAM-Role-Directory Services-EC2Full access-Create Role
• Create One more role for Billing
• Add user A & B to above role
• Copy the URL and paste in Incognito Tab

Session 1 : What is Data,Databases and DBMS

Data : In simple words,data can be facts related to any object.Ex : Age,Job,House Num,Contact Num Name,etc related to us

Database : Database is a Systematic collection of Data.Databases support storage and manipulation of data.Ex : Facebook,Telecom Companies,Amazon etc

Database Management System(DBMS) : It is a collection of programms which enable its users to access database,manipulate data,reporting/representation of data

• *Types of DBMS :* Hierarchial,Network,Relational,Object Oriented

Session 2 : Relational Database (Attribute,Tuple,Horizontal and Vertical Scaling in RDBMS)

Relational Database :

• A relational database is a data structure that allows us to link information from different 'tables' or different types of data bucket
• Tables are related to each other
• All fields must be filled
• Best suites for OLTP(Online Transaction Processing)
• Relational DB : MySQL,Oracle DBMS,IBM DB2 etc..

Attribute	Attribute	Attribute	Attribute
Name	Age	Location	Phone
X	21	LA	+8100
Y	33	CA	+9300
z	18	IN	+9191

☐ A single column is called attribute or fields and a whole row is called Tuple
☐ A row of a table is also called record and it contains the specific information of each individual entry in the table
☐ Each table has its own primary key
☐ A schema(Design of Database) is used to strictly defines table,colum,indexes and relation between tables
• Relational DB are usually used in Enterprise applications/scenario but exception is MySQL which is used for web application
☐ Common application for MySQL include PHP and Java based web applications that require a database storage backend...EX : Joomla
• Cannot scale-out horizontally
• Virtually all Relational DB uses SQL

Session 3 : NoSQL Database(Types-Document,Key-value,Graph based and Columnar)

Non-Relational DB/No-SQL DB :

• Non-Relational databases store data without a structured mechanism to link data from different tables
• Require low cost hardware
• Much faster performance(read/write) compared to relational DB
• Horizontal scaling is possible
• Never provide table with flat fixed column records.It means schema-free
• Best suited for online analytical processing
• Ex : No-SQL Databases- MangoDB,Cassendra,DynamoDB,Postgresql,Raven,Redis

Types of No-SQL Databases :

1.Columnar DB : Cassendra,HBase
2.Document DB : MongoDB,CouchDB,RavenDB
3.Keyvalue DB : Redis,Riak,DynamoDB,Tokyo Cabinet

4.Graph DB : Neo4J,Flock DB

1.Columnar DB :

• It is a DBMS that stores data in colums instead of rows

ID	Name	Age	Bonus
1	Bob	30	3000
2	Alex	26	4000
3	Vijay	22	2000

• In a columnar DB,all the column1 values are physically together,followed by all the colum2 values
• In a row oriented DBMS,the data would be stored like (1.Bob,30,3000.....2.Alex,-26,4000.....ect)
• In a column based DBMS,the database would be stored like (1,2,3 ; Bob,Alex ; 30,26 ; 3000,4000....etc)
• Benefit is that because a colum-based DBMS is self indexing,it uses less disk space that a RDBMS containing the same data
• It easily perform operation like Min,Max & Avg

2.Document DB :

• Document DB make it easier for developers to store and querying data in a DB by using the same document model format they use in their application
• Document DB are efficient for storing catalogue
• Store semi-structured data as document typically in JSON or XML format
• In the following example,A JSON like document describe a book

```
[
  [ “year ”  :  2010,
    “title”    : “AWS Fundamentals”,
     “info”   : {
                    “writer” ; ['Rajput','Bhupinder'],
                    “release date”;"2010-01-07",
                    “rating”          ;    62,
                    “genre”           ;    ["it","science"]
    }
  }
]
```

• A document database is a great choice for content management application such as blog and video platforms

3.Key-Value-DB :

• A key-value-db is a simple DB that uses an associative array (think of a dictionary) as a Fundamental model where each key associated with one and only one value in a collection
• It allows horizontal scaling
• Use cases : shopping cart, and session store in app like FB & Twitter
• They improve application performance by storing critical pieces of data in memory for low latency access
• Amazon elastic cache as an in-memory key-value stores

4.Graph Based DB :

• A graph DB is basically a collection of nodes and edges.Each node represent an entity(like person) and each edge represent a connection or relationship between two nodes

Session 4 : AWS Relational Database Services - Template Types,Storage Types and DB Engine Available

• In an AWS fully managed relational DB engines service where AWS is responsible for :
☐ Security and patching
☐ Automated Backup
☐ Software updates for the DB engine
☐ If selected, multi-AZ with synchronous replication between the active and standby DB instance
☐ Automatic failover if multi-AZ option was selected
☐ By deafult,every DB has weekly maintenance window(max 35 days)

• Settings managed by the users :
☐ Managing DB settings
☐ Creating relational database schema
☐ Database performance tuning

Relational Database Engine Options :

1.MS SQl Server
2.My SQL → Support 64TB of DB
3.Oracle
4.AWS aurora -→ High Throughput
5.Postgre SQL -→ Highly Reliable & Stable
6.MariaDB --→ MySQL Compatible,64TB DB

There are two Licensing Options :

1.BYOL----Bring uour own license
2.License from AWS on hourly Basis

RDS Limits :

• Upto 40DB instances per account
• 10 of this 40 can be oracle or MS-SQL servers under License include model (or) under BYOL model,all 40 can be any DB engine you need

RDS Instance Storage :

• Amazon RDS use EBS volumes(not instance store) for DB and logs storage

1.General Purpose : Use for DB workloads with moderate I/O requirement .....Limits : Min 20GB....Max 16384GB
2.Provisional IOPS RDS Storage : Use for high performance OLTP workloads.....Limits : MIn 100GB....Max 16384GB.....This is recommend to use when we want to create multi-az environment for standby db

Templates Available in RDS :

1.Production : Use defaults for high availability and fast,consistent performance
2.Dev/Test : This instance is intended for development use outside of a production environment
3.Free-Tier : Use free-tier to develop new app,test existing app,or gain hands on experience with Amazon RDS

DB Instance Size :

1.Standard Class : Include M class
☐ MAx 96 VCPU....384GB Ram.....EBS 14000 MBPS

2.Memory Optimized Class : Include r & x class
☐ Max 96VCPU.....768GB ram....EBS 14000 MBPS

3.Burstable Class : Include t class
☐ Max 8VCPU....32GB Ram.....EBS 1500MBPS

Session 5 : AWS Relational Database Services - Concept of Multi-AZ in RDS

Multi-AZ in RDS :

• We can select Multi-AZ option during RDS DB instance launch
• RDS service creates a standby instances in a diff AZ in the same region,and configure “Synchronous Replication” between the primary and standby
• We can not read/write to the standby RDS DB instances directly
• We cannot select,which AZ in the region will be choosen to create the standby DB instance
• We can however view which AZ is selected after the standby is created
• Depending on the insatnce class,it may take 1 to few minutes to failover to the standby instance
• AWS recommends the use of provisioned IOPS instances for Muti-AZ RDS Instances

When Multi-AZ RDS Failover Triggers :

1.In case of failure of primary DB instance failure
2.In case of AZ failure
3.Loss of Network connectivity to primary DB
4.Loss of primary EC2 instance failure
5.EBS failure of primary DB instance
6.The primary DB instance or settings are changed
7.Patching the O.S of the DB instance
8.Manual failover(incase of rebooting)

Multi-AZ RDS Failover Consequences :

1.During failover,the CNAME of the RDS DB instance is updated to Map to the standby IP address
2.It is recommended to use the endpoints(URL) to reference our DB instances and not its IP address
3.The CNAME does not change,because RDS endpoint does not change region

• RDS endpoint does not change by selecting Multi-AZ option.However the primary and standby instances will have different Ip addresses,as they are in diff AZ
• It is always recommended that we do not use the IP address to point RDS

instances,always use endpoint.By using endpoint,there will be no change whenever a failover happens

When we do Manual Failover :

• In case of rebooting
• This is by selecting the “reboot with failover” reboot options on the primary RDS DB instances
• A DB instance reboot is required for changes to take effect when we change the DB parameter group or when we change a static DB parameter

Session 6 : AWS Relational Database Services - AWS RDS Backup and Retention Period

• Whenever failover occurs,AWS RDS sends SNS notification
• We can use API calls to findout the RDS events occured in the last 14 days
• Even,we can use CLI to view last 14 days events
• Using AWS console,we can view only last one day events
• In case of OS-patching,system upgrades,DB scaling,these things happens on standby first,then on primary to avoid outage
• In multi-Az,snapshots and automated backups are done on standby instance to avoid I/O suspension on primary

RDS Multi-AZ Deployment-Maintenance

• Firstly,perform maintenance on standby
• Now,convert standby into primary so that maintenance can be done on primary(currently)
• We can manually upgrade a DB instance to a supported DB engine version from AWS console as follows : RDS---->DB instance---->Modify DB----->Set DB Engine Version
• By default,change wiill take effect during the next maintenance window
• Or we can force an immediate upgrade if we want
• In multi-az,version upgrade will be conducted on both primary and standby at the same time,which will cause an outage
• So do it during maintenance window

Methods to Backup :

• There are 2 methods to backup and restore our RDS DB instances
1.AWS RDS automated backup
2.User Initiated manual backup

• Either we can take backup of entire DB instance or just the DB
• We can create a storage volume snapshots of our entire DB instances

Automated Backup

• Automated backups by AWS backup our DB data to multiple-AZ to provide for data durability
• Select -Automated Backup in AWS console
• Stored in Amazon S3
• Multi-AZ automated backups will be taken from the standby instance
• The DB instanc must be in “Active” state for automated backup

• RDS automatically backups the DB instance daily,by creating a storage volume snapshot of our DB instance(fully daily snapshot) including the DB transaction logs
• We can decide,when we would like to take backup(window)
• No additional charge for RDS backing up our RDS instances
• For multi-AZ deployment,backups are taken from the standby DB instance(True for MariaDB,Mysql,Oracle,PostgreSQL)
• Automated backup are deleted when we delete our RDS DB instance
• An outage occurs if we change the backup retention period from zero to non-zero value or the other way around
• Retention period of automated backup is 7 days(By default) via AWS console but AWS aurora is an exception its default is one day
• Via CLI or API,1 day by default
• We can increase it upto 35 days
• If we do not want backup,put zero in retention period

Session 7 : AWS Relational Database Services - RDS Encryption,Manual Backup and Billing

• In case of manual snapshot,point-in-time recovery is not possible
• Manual snapshot is also stored in S3
• They are not deleted automatically,if we delete RDS instance
• Take a final snapshot before deleting our RDS DB instance
• We can store manual snapshot directly with other AWS account
• When we restore a DB instance,only the default DB parameters and security groups are associated with the Restored instance
• We cannot restore a DB snapshot into an existing DB instance rather it has to create a new DB instance.It has new endpoint
• Restoring from a backup as a DB snapshot changes the RDS instance endpoint
• At the time of restoring,we can change the storage type(general purpose or provisioned)

RDS Encryption :

• We cannot encrypt an existing unencrypted DB instance
• To do that,we need to :
☐ create a new,encrypted instance and migrate our data to it(from unencrypted to encrypted) (OR)
☐ We can restore from a backup/snapshot into a new encrypted RDS instance
• RDS supports encryption at rest for all DB engines using KMS

What actually encrypted when data at rest :

1.All its snapshots
2.Backups of DB(S3 Storage)
3.Data on EBS volume
4.Read replica created from the Snapshot

Points related to RDS Billing :

• No upfront cost
• We have to pay only for :
☐ DB instances hours(partial hours chargeable)
☐ Storage GB/Month
☐ Internet data transfer
☐ Backup storage(i.e S3)...This increases by increasing DB backups retention period

Aws also Charge for :

•Multi-AZ DB hours
• Provisioned storage(Multi-AZ)
• Double write I/O
• We are not charged for DB data transfer doing replication from primary to standby

LABS :

Session 8 : Accessing MySQL DB instance from Linux

Session 9 : Accessing MySQL DB instance from Windows Server

Session 1 : Basics of AWS Route-53 and its Functions .....DNS and how it Works

• We can use Route-53 to register new domains,transfer existing domains,route traffic for our domains to our AWS and external resources and monitor the health of our resources
• DNS TCP port number is 53.....This is the reason why amazon used the name Route 53

Route-53 Functions :

1.DNS Management
2.Traffic Management
3.Availability Monitoring
4.Domain Registration

Session 2 : Route-53 Functions and Steps

Route-53 performs three main functions :

1.Register a domain
2.As a DNS,it routes internet traffic to the resources for our domain
3.Check the health of our resources
☐ Route-53 sends automated requests over the internet to a resource(can be a webserver) to verify that the server is,reachable functional or available
☐ Also we can choose to receive notifications when a resource becomes unavailable and choose to route internet traffic away from unhealthy resources

• We can use Route-53 for any combination of these functions :
☐ EX : We can use Route-53 both to register our domain name and to route internet traffic for the domain
☐ Or we can use Route-53 to route internet traffic for a domain that we registered with another domain register

• When we register a domain with Route-53,the service automatically makes itself the DNS service for the domain by doing the following :

1. It creates a hosted zone that has the same name as our domain
2.It assigns a set of four name servers to the hosted zone,unique to the account
☐ When someone uses a browser to access our website,these name servers inform the browser where to find our resources,such as a web server or an Amazon S3 bucket
☐ It gets the name servers from the hosted zone and adds them to the domain

AWS Supports :

1.Generic Top level Domains.....com,.org,.net etc.....
2.Geographic Top Level Domains.....in,.us,.me etc

Registering a domain with Route-53 :

• We can register a domain with route-53 if the TLD is included on the supported TLD list
• If the TLD is not included,we can't register the domain with route-53

Using Route-53 as our Service :

• We can use Route-53 as the DNS service for any domain,even if the TLD for the domain is not included on the supported TLD list

Note : Each amazon Route-53 account is limited to a maximun of 500 hosted zones and 10,000 resource record sets per hosted zone. We can increase this limit by requesting to AWS

Steps to configure Route-53 :

1.We need to register a domain,this can be route-53,or another DNS registrar,but then we connect our domain name in that registrar to route-53
2.Create hosted zone on route-53(if we purchased domain from registrar other than route-53),this is done automatically if we registered our domain using route-53
☐ Inside the hosted zone,we need to create record sets

Delegate to Route-53 :

• This step connects everything and make it works
• Connect the domain name to the route-53 hosted zone.This is called delegation
• Update our domain registrar with the correct name servers for our route-53 hosted zone
• No other customer hosted zone will share this delegation set with us
• Doing this means route-53 DNS service will be serving DNS traffic for the domain of the hosted zone
• If we registered our domain with a diff registrar,we need to configure the route-53 NS's list in our registrar DNS database for our domain

When we are using another domain provider and we did all the changes :

• When we migrate from one DNS provider to another,For an existing domain this change can take upto 48 hours to be effective
• This is because name server DNS records are typically cached across the DNS system globally on the internet for upto 48 hours(TTL) periods

Transferring a Domain to Route-53 :

• We can transfer a domain to route-53 if the TLD is included on the Amazon supported TLD list
• If the TLD is not included,we cant transfer the domain to Route-53
• For most TLD,we need to get authorization code from the current registrar to transfer a

domain

Session 3 : Name Server and SOA

Route-53 Hosted Zone :

• A route 53 hosted zone is a collection of records for a specified domain
• We create a hosted zone for a domain and then we create records to tell the domain name system how we want traffic to be routed for that domain
• Basically a hosted zone is a container that holds information about how we want to route traffic for domain and its sub domains
• We can create public(internet) hosted zone or private(internal) hosted zone
• For each public hosted zone that we create amazon route 53 automatically creates name server(ns) record and a start of authority(SOA) record.Dont change these records
• Route 53 automatically creates a name server(ns) record with the same name as our hosted zone
• It list the four name servers that are authoritative name servers for our hosted zone
• Do not add,change or delete name servers in this record
• When we create a hosted zone,amazon route-53 automatically creates a name server(ns) records and a start of authority record(SOA) for the zone
• The ns record identifies the four name servers that we give to our registrar or our DNS service so that DNS queries are routed to route53 name servers
• By default route-53 assigns a unique set of four name servers(known collectively as a delegation set)
• Ex : ns-1337 awsdns-39.com
ns-895 awsdns-47.net
ns-428 awsdns-53.org
ns-1597 awsdns-07.co.uk

Route 53 as our Authoritative DNS :

• Once we update the route 53 NS settings with our domain registrar to include the route 53 name server,route 53 will be responsible to respond to DNS queries for the hosted Zone
• This is true whether we do have a functioning website or not
• Route53 will respond with information about the hosted zone whenever someone types the associated domain name in a web browser
• We can create more than one hosted zone with the same name and add diff records to each hosted zone
• Route 53 assigns four name servers to every hosted zone
• The name servers are diff for each of them
• When we update our registrars name server records, be careful to use the route 53 name servers for the correct hosted zone the one that contains the records that we want route 53 to use when responding to queries for our domain
• Route 53 never returns values for records in other hosted zone that have the same name

Route 53 Hosted zone default entries :

• Inside the hosted zone by default we have two entries :
☐ *NS Entry :* Contains the unique sets of name servers for this hosted zone
☐ *SOA entry :* Contains information about the hosted zone

Session 4 : DNS Record Types

• If we are currently using another DNS service and we want to migrate to Amazon Route 53 :
☐ Start by creating Hosted Zone
☐ Route 53 automatically assigns the delegation sets,the four name servers to our hosted zone
☐ To ensure that the DNS routes queries for our domain to the route 53 name servers
☐ Update our registrar's or our DNS servers NS records for the domain to replace the current Name servers with the names of the four Route 53 name servers for our hosted zone
☐ The method that we use to update the NS rcords depends on which registrar or DNS service we are using
☐ Some registrar only allow us to specify name servers using IP addresses they dont allow us to specify fully qualified domain names
☐ If our registrar requires using Ip addresses,we can get the IP addresses for our name servers using the dig utility (for mac and Linux) and nslookup(for windows)

Transferring a domain between accounts within AWS :

• Transferring a domain to a diff AWS account :
☐ If we registered a domain with one aws account and we want to transfer the domain to another AWS account,we can do so by contacting the AWS support center and Requesting the transfer

Migrating a hosted zone to a diff AWS account :

• If we are using Route53 as the DNS service for the domain,route53 does not transfer the hosted zone when we transfer a domain to a diff AWS account
• If domain registration is associated with one account and the corresponding hosted zone is associated with another account,neither domain registration nor DNS functionality is affected
• The only effect is that we will need to sign into the route 53 console using one account to see the domain and sigin using the other account to see the hosted zone

Supported DNS Record Types by Route 53 :

1.A record : Address Record maps domain names to IP Address(32 bit IPv4 Address)....www.techguftugu.com in A 5.5.5.5

2.AAAA Record : IPv6 address record maps domain name to an IPv6 address(128 bit IPv6 address so the name AAAA[32bits*4])

3.CNAME Record : Canonical name record maps an alias to a hostname....web in CNAME techguftugu.com

4.NS record : Name server Record used for delagating zone to a nameserver.....techguftugu.com in NS ns1.techguftugu.com

5.SOA Record : Start of Authority Record

6.MX record : Mail echange defines where to deliver mail for user@domain name....techguftugu.com in MX 10 mail.techguftugu.com

NS record : It defines which name server is a authoritative to a particular zone or domain name and point us to other DNS servers

• A/AAAA are called host records,like business cards
• CNAME is an alternative records,or an alias for another record
☐ Helpful in redirection or if we want to hide details about our actual servers from the user

SOA Record :

• Every single zone has one and only SOA resource record at the beginning of the zone
• It is not an actual record,it includes the following info :
☐ Who the owner is(email for the domain)
☐ The authoritative server
☐ The serial number which is incremental with changes to the zone data
☐ The refreshing time/cycle info and the time to live(TTL)

CNAME Record :

• A CNAME value element is the same format as a domain name
• The DNS protocol does not allow us to create a CNAME record for the top node of a DNS namespace,also known as the zone apex(or root domain)
• For EX : if we register the DNS name techguftugu.com,the zone apex is techguftugu.com and we cannot create a CNAME for techguftugu.com
• However we can create CNAME records for WWW.techguftugu.com,support.techguftugu.com and so on
• In addition,if we create a CNAME record for a subdomain we cannot create any other record for that Subdomain
• EX: If we create a CNAME for www.techguftugu.com we cannot create any other records for which the value of the name field is www.techguftugu.com

Session 5 : Route53 Routing Policies-Simple,Failover,Geo Location,Latency,Geo Proximity,Weighted and Multi-Value Policy

• When we create a record,we choose a routing policy,which determines how amazon route 53 respond to Queires
☐ Simple,Failover,Geo Location,Latency,Geo Proximity,Weighted and Multi-Level Routing Policies

Failover Routing Policy :

• Failover routing lets us route trafiic to a resource when the resource is healthy if the main resource is not healthy,then route traffic to diff resource
• The primary and secondary records can route traffic to anything from an amazon S3 buccket that is configured as a website to a complex tree of records
• Failover routing policy is applicable for public hosted zone oly

Geolocation ROuting :

• It lets us choose the resources that serves our traffic based on the geographic location of our users i.e the location that DNS queries originate from
• Ex : We may have presence in europe and asia and we want users in the asia to be served in the asia and those in europe to be served by servers in europe

Benefits :

• We can localize our content and present some or all of our website in the language of our

users
• We can also use geolocation routing to restrict distribution of content to only the locations in which we have distribution rights
• We can specify geographic locations by continent,by country or by state in the US
• If we create seperate records for overlapping geographic regions for ex : one record for North America and one for Canada-priority goes to the smallest geographic region(canada)
• Geolocation works by mapping IP address to location.However some IP address are not mapped to geographic location

Latency based Routing :

• If our application is hosted in multiple amazon ec2 regions,we can improve performance for our users by serving their request from the amazon ec2 region that provides the lowest latency
• To use this routing we need to create latency records for our resources in multiple ec2 regions
• When amazon route 53 receives a DNS query for our domain or subdomain
☐ It determines which amazon ec2 region we have created latency record for
☐ Determine which regions gives lowest latency to users
☐ Then select a latency record for that region
• Ex : Suppose we have ELB in US-East and in asia pacific(mumbai) region
• We created a latency record for each load balancer
• Here's what happened when a user in londen enters the name of our domain in browser
• DNS routes the request to a route 53 name server
• Route 53 refers to its data on latency between london and the mumbai region and between london and N.virginia
• If latency is lower between london and N.verginia,route 53 respond to the query with the IP address for the N.verginia LB

Weighted Routing Policy :

• It lets us associate multiple resources with a single domain name or subdomain name and choose how much traffic is routed to each resource
• This can be useful for a variety of purposes,including load balancer and testing new versions of software
• Weights can assign any number from 1 to 255
• Weighted routing policy can be applied when there are multiple resource that performs the same function...Ex : Webserver serving the same website
• To configure weighted routing,we create records that have the same name and type for each of our resources
• Amazon route 53 sends traffic to a resource based on weight that we assign to the record as a proportion of the total weight for all records in the group
• Ex : Suppose for www.tg.com has three resource record sets with weights of 1(20%) , 1(20%) and 3(60%).....Sum =5
• On average,route 53 select each of the first two resources record set one-fifth of the time and returns the third resource record set three-fifth of the time

Geo Proximity Routing Policy :

• Use when we want to route traffic based on the location of our resources and optionally,shift traffic from resources in one location to resources in another
• We can also optionally choose to route more traffic or less to a given resource by specifying a value,known as a ‘bias’.A bias expand or shrinks the size of the geographic region from which traffic is routed to a resource

Multivalue Answer Routing Policy :

• Use this,when we want route 53 to respond to DNS queries with upto eight healthy record selected at random
• This lets us configure amazon route 53 to return multiple values,such as ip addresses for our webservers in response to DNS queries.We can specify multiple values for almost any record,but multivalue answer routing also lets us check,the health of each resource,so route 53 returns only values for healthy resources.Its not a substitute for our load balancer but the ability to return multiple health checkable IP addresses is a way to use DNS to improve availability and load balancing

Name	Type	Value	TTL	Set ID	H.C(He Check)
www.tg.com	A record	192.0.2.2	60	Web1	A
www.tg.com	A record	198.50.100.1	60	web2	B
www.tg.com	A record	200.1.1.1	60	web3	C
www.tg.com	A record	192.0.3.3	60	web4	D

LAB :

Session 6 : AWS Route 53 DEMO

Session 1 : Aws Cloudfront and How it Works-Edge Location,Origin and Regional Cache(Explanation)

• Amazon Cloudfront is a web service that gives businesses and web application developers an easy and cost effective way to distribute content with low latency and high data transfer speed

Session 2 : Edge Location and Regional Edge Cache

• Cloudfront is a global service
• Amazon cloudfront is a webservice that speeds up distribution of our static and dynamic web content,such as html,css,js,and image files to users
• Cloudfront delivers our content through a worldwide network of data centers called edge locations
• When a user request content that we are serving with with cloudfront,the user is routed(via dns resolution) to the edge location that provides the lowest latency,So that content is delivered with the best possible performance
• If the content is already in th edge location with the lowest latency,cloudfront delivers it immediately
• This dramatically reduces the number of networks that our users request must pass through, which improves performance
• If not,cloudfront retrives it from an amazon s3 bucket or an HTTP/webserver that we have identified as the source for the definitive version of our content(origin server)
• Cloudfront also keeps persistent connection with origin server so files are fetched from the origin as quickly as possible

We can Access Amazon Cloudfront in the following ways :

• AWS management console
• AWS SDK
• Cloudfront API
• AWS CLI

Cloudfront Edge Locations

• Edge locations are not tied to available zones or regions
• Amazon Cloudfront has 216 point of presence(205 edge locations and 11 regional edge caches) 84 cities across 42 countries

Cloudfront - Regional Edge Caches :

• Amazon cloudfront has added several regional edge cache locations globally at close proximity to our viewers
• They are located between our origin webserver and the global edge locations that serve content directly to our viewers
• As objects become less popular individual edge locations may remove those objects to make room for more popular content
• Regional edge cache working as a alternative of origin to reduce the burden of origin
• Regional edge cache have a large cache width than any individual edge location,so object remian in the cache longer at the nearest regional edge caches

Cloudfront Regional Edge Cache -Working :

• When a viewer makes a request on our website or through our application,DNS routes the request to the cloudfront edge location that can best serve the users request
• This location is typically the nearest cloudfront edge location in terms of latency
• In the edge location,cloudfront checks its cache for the requested files
• If the files are in the cache,cloudfront returns them to the user
• If the files are not n the cache,the edge servers go to the nearest regional edge cache to fetch the object
• Regional edge cache have feature parity with edge locations .EX : a cache invalidation request removes an object from both edge caches and regional edge caches before it expries
• The next time a viewer request the object,cloudfront returns to the origin to fetch the latest version of the object
• Proxy method PUT/POST/PATCH/OPTIONS/DELETE go directly to the origin from the edge locations and do not proxy through the regional edge caches
• Dynamic content as determined at request time,does not flow through regional edge cache,but goes directly to the origin

LAB :

Session 3 : AWS Cloudfront DEMO

Session 1 : How Simple Queue Service Works

Session2 : Important points of SQS and Billing

• SQS is a pull service and SNS is push service
• SQS is a fast,reliable,fully managed message queue service
• it is a webservice that give us access to message queues that store messages waiting to

be processed
• It offers a reliable,highly scalable hosted queue for storing messages between servers
• It allows the decoupling of application components such that a failure in one components does not cause a bigger problem to application functionality(like a coupled app)
• Using SQS,we no longer need a highly available message cluster or the burden of running it
• We can delete all the messages in an SQS queue without deleting the SQS queue itself
• We can use application on ec2 instances to read and process the SQS queue messages
• We can use autocaling to scale ec2 fleet processing the SQS messages,as the queue size increases
• These applications on ec2 instances can process SQS messages/jobs then post the results to other Queues to other AWS services

AWS Queue Types :

1.Standard Queue

• High(unlimited) throughput
• Atleast one delivery
• Duplicacy is possible
• Best effort ordering

2.FIFO Queue

• Limited throughput(300 TPS)
• Exactly one processing
• Duplicacy not possible
• Strict ordering -First in First out
• FIFO queues are limited to 300 transactions per second(TPS),but have all the capabilities of standard queue

SQS Pricing :

• The first 1 million monthly requests are free.After that pricing is according to regions EX : In mumbai region Standard Queue-$0.40/million request.....FIFO Queue : $0.50/million request

How Amazon SQS Charges :

• API action : Every Amazon SQS actions count as a request
• FIFO Request : API actions for sending,receiving,deleting and changing visibility of messages from FIFO Queues are charged at FIFO rates
• Contents of Request : A single request can have from 1 to 10 messages,upto a max total payload of 256KB
• Size of Payload : Each 64KB chunk of a payload is billed as 1 request(EX : API action with a 256KB payload is billed as 1 request)
• Interaction with Amazon S3
• Interaction with AWS KMS

Session 3 : Receive Message wait Time,Delivery Delay,Visibility timeout and Dead letter Queue

Short Polling :

• A request is returned immediately even if the queue is empty
☐ It does not wait for messages to appear in the queue
☐ It queries only a subset of the available servers for messages(based on weighted random distribution)
☐ Default by SQS
☐ Receive message time is set to 0
• More requets are used which implies higher cost

Long Polling :

• Is preferred to regular/short polling.It uses fewer requests and reduce cost by :
☐ Eliminating false empty responses by querying all the servers
☐ Reduce the number of empty respones by allowing amazon SQS to wait untill a message is available in the queue before sending a response unless the connection timeout (20 sec)
☐ Receive message time is set to a non-zero value(max 20 sec)
☐ Polling is same for both pollings

SQS-Retention Period :

• SQS messages can remain in the queue for upto 14 days(SQS retention period)
• Range is 1 min to 14 days(default is 4 days)
• Once the max retention period of a message is reached,it will be deleted automatically from the queue
• Messages can be sent to the queue and read from the queue simultaneously
• SQS can be used with DynamoDB,EC2,ECS,Redshift,RDS,Lambda,S3 to make distributed/-decoupled applications
• We can have multiple queues with diff priorities

SQS-Visibility Timeout :

• Is the duration of time a message is locked for read by other servers
• Max is 12 hours and default is 30 sec
• A server that read a message to process, it can change the message visibility timeout if it needs more time to process the message
• After a message is read,there are the following possibilities :
☐ An ACK is received that a message is processed,so it must be deleted from the queue to avoid duplicates
☐ If a fail is received or the visibility timeout expires,the message will then be unlocked for read,such that it can be read and processed by another servers

Delivery Delay :

• AWS SQS provides delivery delay options to postpone the delivery of new messages to a queue.If delivery delay is defined for a queue,any new messages will not be visible to the server for the duration of delay.The default(min) delay for a queue is 0 sec.The max is 15 Min

Receive Message Wait Time :

• The default time is 0 Sec.This is max amount of time that a long polling receive call will wait for a message to become available before returning an empty reponse(Max value is 20 Sec)

Dead Letter Queue :

• The main task of a dead letter queue is handling message failure.A dead letter queue lets us set aside and isolate messages that can't be processed correctly to determine why their

processing didn't succeed
• Dont use a dead letter queue with a FIFO queue,if we don't want to break the exact order of messages or operations
• DLQ must be of the same type as the source queue(standard or FIFO)

LAB :

Session 4 : SQS Queue Triggers on Lambda Function

Session 1 : SNS Theory

• SNS is a fast,flexible,fully managed push notification service
• It is a web service that co-ordinates and manages the delivery or sending of messages to subscribing endpoints or clients
• It allows for sending individual messages or fan-out messages to a large number of recepients or to other distributed AWS service
• Message published to an SNS topics will be delivered to the subscriber immediately
• Inexpensive,pay as you go model with no upfront cost
• Reliable: At least three copies of the data are stored across multiple AZ in same region
• It is a way of sending messages.When we are using autoscaling,it triggers an SNS service which will email us that ‘our EC2 instance is growing’

Publisher-----SNS Topic-------->1.LAmbda.....2.SQS.....3.HTTP/S.....4.Email.....5.SMS

Publisher : Publishers are also known as producers that produce and send the message to the SNS which is a logical access point

Subscriber : Subscribers such as webserver,email addresses,amazon SQS queues,AWS Lambda,Receive the message or notification from the SNS over one of the supported protocols(Amazon SQS,email, lambda,https,sms)

SNS Topic :

•Is a logical access point and communication channel
• Each topic has a unique name
• A topic name is limited to 256 alphanumeric charcters
• The topic name need to be unique within the AWS account
• Each topic is assigned an AWS ARN once it gets created
• A topic can support subscribers and notification delivers over multiple protocols
• Messages/request published to a single topic can be delivered over multiple protocols as configured when creating each subscriber
• Delivery formats/transport protocols(endpoints)
☐ SMS
☐ Email
☐ Email-JSON-For Applications
☐ HTTP/HTTPS
☐ SQS
☐ AWS Lambda
• When using Amazon SNS,we(as the owner) create a topic and control access to it by defining access policies that detremine which publishers and subscribers can communicate with the topic

• Instead of including a specific destination address in each message,a publisher sends messages to topic that they have created or to topics they have permission to publish to
• Amazon SNS matches the topic to a list of subscribers who have subscribed to that topic,and delivers the message to each of these subscriber
• Each topic has a unique name that identifies the Amazon SNS endpoint for publisher to past messages and subscribers to register for notifications
• Subscribers receive all messages published to the topics to which they subscribe,and all subscribes to a topic receive the same messages
• By default,only the topic owner(who created it) can publish to the SNS topic
• The owner set/change permissions to one or more users(with valid AWS ID) to publish to his topic
• Only the owner of the topic can grant/change permission for the topic
• Subscribers can be those with/without AWS ID.Only subscriber with AWS ID can request subscription
• Both publishers and subscribers can use SSL to help secure the channel to send and receive messages

Supported Push Notification Platforms :

☐ Amazon Device Messaging
☐ Apple push notifaction service
☐ Google cloud messaging
☐ Windows piush notification service
☐ Baidu cloud push for Android

• SNS topic can have subscribers from any supported push notification platform,as well as any other endpoint type such as SMS or email
• When we publish a notification to a topic,SNS will send identical copies of that message to each endpoint subscribed to the topic

Amazon SNS Alternatives

☐ Amazon Kinesis Data Stream
☐ Amazon Managed Queue Service(AWS MQ)
☐ Apache Kafka
☐ Twilio
☐ Pusher

Amazon SNS Pricing :

1.Publish Action : Each 64kb of request payload count as one request.So,256kb payload will charged as four payloads
2.Mobile Push Notification : Ex : $0.50/million request
3.SMS : Price depends on country
4.Email : $ 2/1,00,000
5.HTTP/HTTPS Notification : $0.60/miilion requests
6.SQS and Lambda calls are free.These are charged at SQS and lambda roles
7.Data Transfer

LAB :

Session 2 : Sending Email and SMS from SNS

Session 1 : DynamoDB Theory

Database Types :

1. Unstructured Data : It has information that either does not have a pre-defined data model or is not organised in a pre-defined manner
• Unstructred information is typically text heavy,but may contain data such as dates,numbers and facts as well.Examples include email messages,word processing,documents,videos,photos,audio files,presentations,webpages

2.Semi-Structured Data : It is information that does not reside in a relational database but that does have some organisational proerties that make it easy to analyse...Eg : XML & JSON

3.Structured Data : It refers to information with high degree of organization,such that inclusion in a relational database is seamless and readily searchable by simple,straight forward search engine algorithims or other search operations
• All data which can be stored in database SQL in table with rows and colums.They have relational key and can be easily mapped into pre-defined fields

Dynamo DB Table :

• A table is a collection of data items
• Like all other DB,dynamodb stores data in tables

Items :

• Each table contains multiple items
• An Item is a group of attributes that is uniquely identifiable among all of the other items
• An item consist of a primary or composite key and a flexible number of attributes
• Items in Dynamodb are similar into Rows,Records in other db

Attributes :

• Each itme is composed of one or more attributes
• An attributes consist of the attribute name and a value or a set of values
• An attribute is a fundamental data elemant,something that does not need to be broken down any further
• Aggregate size of an item cannot exceed 400KB including key and all attributes
• If the item size is more than 400KB, then we can upload the item in S3 bucket and then we can add the URL from S3 bucket in the table

•Dynamodb allows low latency read/write access to items ranging from 1 byte to 400KB
• Dynamodb can be used to store pointers to S3 stored objects,or items of sizes larger than 400KB too if needed
• Dynamodb stores data indexed by a primary key-We can specify the primary key when we create the table
• Each item in the table has a unique identifier or primary key that distinguishes the item from all the others in the table
• The primary key is the only required attributes for items in a table
• Dynamodb tables are Schemaless
☐ Which means that neither the attributes nor their data types need to be defined beforehand
☐ Each item can have its own distinct attribute

DynamoDB - Read Capacity Unit :

• One read capacity unit represents one strongly consistent read per second or two eventually.Consistent reads per seconds for an item upto 4KB in size
• If we need to read an item that is larger than 4KB,DynamoDB will need to consume additional read capacity units
• The total number of read capacity units required depends on the item size,and whether we want an eventually consistent or strongly consistent read

DynamoDB - Write Capacity Unit :

• One write capacity unit represents one write per second for an item upto 1KB in size
• If we need to write an item that is larger than 1KB,DynamoDb will need to consume additional write capacity units
• The total number of write capacity units required depends on the item size

DynamboDb - Pricing :

• Reads are cheaper than writes when using DynamoDB
• We pay for :
☐ Each tables provisioned read/write throughput(hourly rate)
☐ We are charged for provisioned throughput regardless whether we use it or not
☐ Indexed data storage
☐ Internet data transfer(if crosses a region)
☐ Free tier per account(accross all tables) of 25 read capacity units and 25 write capacity units per month

• DynamoDB can do 10000 write capacity units/sec or 10000 read capacity units per second per table

DynamoDB Limits :

• 256 tables per account per region
• No limits on the size of any table

LAB :

Session 2 : DynamoDB Demo

Session1 : Basic Introduction,Lambda Function,Trigger,Downstream Resources and Runtime

• AWS Lambda is a compute service that lets us run code without provisioning or managing servers
• With AWS Lambda,we can run code for virtually any type of application or backend-service all with zero administration
• AWS Lambda manages all the administration,It manages :
☐ Provisioning and capacity of the compute fleet that offers a balance of memory,CPU,Network and other resources
☐ Server and O.S maintenance
☐ High availability and automatic scaling

☐ Monitoring fleet health
☐ Applying security patches
☐ Deploying security patches
☐ Deploying our Code
☐ Monitoring and logging our Lambda functions
☐ AWS Lambda runs our code on a high availability compute infrastructure

• AWS Lambda runs our code on a high-availability compute infrastructure
• AWS Lambda executes our code only when needed and scales automatically,from a few requests per day to thousands per second
• We pay only for the compute time we consume.No charges when our code is not running
• All we need to do is supply our code in the form of one or moe lambda functions to AWS Lambda,in one of the languages that AWS Lambda supported(Currently Node JS,JAVA,Powershell,C#,Ruby,Python and GO) and the service can run the code on our behalf

• Typically the lifecycle for an AWS Lambda based application includes authoring code,deploying code to AWS Lambda and then monitoring and troubleshooting
• This is in exchange for flexibility means we cannot log into Compute instances or customize the O.S or language runtime
• If we do want to manage our own compute,we can use EC2 or Elastic Beanstalk

How Lambda Works :

• First we upload our code to lambda in one or more lambda functions
• AWS Lambda will then execute the code in our behalf
• After the code is invoked lambda automatically take care of provisioning and managing the required servers

AWS Lambda vs AWS EC2

AWS Lambda :

• AWS lambda is a Platform as a Service
• It supports only limited languages like NodeJs,Java,Python,Ruby,GO,C#,Powershell
• We write our code and push the code into AWS Lambda
• We cannot log into compute instances,choose customized O.S or language platform

AWS EC2 :

• It is a IAAS
• No environment restrictions,we can run any code or language
• For the first time in EC2,we have to choose the O.S and install all the software required and then push our code in EC2
• We can select variety of O.S,instance types,Network & security patches,RAM & CPU etc

Important Terms used in AWS Lambda :

1.Functions : A function is a resource that we can invoke to run our code in AWS Lambda.A function has code that processes events and a runtime that passes request and responses between lambda and the function code

2.Runtime : Lambda Runtime allows functions in different languages to run in the same base execution environment.The runtime sits in between the lambda server and our

function code,relaying invocation events,context information and responses between the two

3.Event : It is a JSON formtted document that contain data for a function to process

4.Event Source/Trigger : An AWS service such as Amazon SNS,or a custom service that triggers our function and executes its logic

5.Downstream Resources : An AWS service,such as DynamoDB tables or S3 buckets,that our lambda function calls once it is triggered

6.Concurrency : No.of request that our function is serving in any given time..Default is 1,000 times

Session 2 : AWS Lambda Invocation Types

When Lambda Triggers :

• We can use AWS Lambda to run our code in response to :
☐ Events such as changes to data in amazon s3 ucket or an amazon dynamodb table
☐ To run our code in response to HTTP request using amazon AOI gateway
☐ With these capabilities,we can use lambda to easily build data processing triggers for AWS services like amazon s3 and amazon dynamodb,process streaming data stored in kinesis or create our own backend that opeartes at AWS scale,performance and security

Example of S3 :

• The user create an object in a bucket
• Amazon S3 detects the object created event
• Amazon S3 invokes our lambda functions using the permission provided by the execution role
• Amazon s3 knows which lambda function to invoke based on the event source mapping that is stored in the bucket notification configuration

AWS Lambda Function Configuration :

• A Lambda function consists of code and any associated dependencies
• In Addition,a lambda function also has configuration inforamtion associated with it
• Initially,we specify the configuration information when we create a lambda function
• Lambda provides an API for us to update some of the configuration data

Lambda function configuration information includes the following key elements :

• Compute resource that we need.We only specify the amount of memory we want to allocate from our lambda function
• Aws lambda allocates,CPU power proportional to the memory by using the same ratio as a general purpose amazon ec2 instance types,sucha s an M3 type
• We can update the configuration and request additional memory in 64mb increments(64 multiples) from 128 to 3008 MB
• Functions larget than 1536MB are allocated multiple CPU threads

Maximum Execution Timeout :

• We pay for the AWS resources that are used to run our lambda function
• To prevent our lambda function from running indefinitely,we specify a timeout
• When the specified timeout is reached,AWS lambda terminates our lambda function
• Default is 3 seconds and maximum is 900 Seconds (15 Min)

IAM Role : This is the role that AWS lambda assume when it executes the lambda function on our behalf

AWS Lambda Function - Service it can Access :

• Aws Services or non-aws services
• Aws services running in AWS VPC(eg-redshift,elasticcache,rds instance)
• Non-aws services running on ec2 instance in an aws vpc
• Aws lambda run our function code securely within a vpc by deafult
☐ However,to enable our lambda function to access resources inside our private vpc,we must provide additional vpc specific configuration information that includes vpc subnet ID and security group ID's

Different ways to invoke Lambda Function : Synchronous invoke(push),Asynchronous invoke(event),Poll-based invoke(pull based)

Synchronous Invocation :

• Synchronous invoke are the most straight forward way to invoke our lambda function
• In this model our function executes immediately when we perform the lambda invoke API call
• Invokation flag specifies a value of ‘request response’
• We want for the function to process the event and return a response

Here is the list of services that invoke lambda function synchronously

☐ Elastic lad balancer
☐ Cloudfront
☐ Amazon cognito
☐ Api gateway
☐ Amazon Lex
☐ Kinesis data firehose

Asynchronous invocation :

• In this incokation lambda places the event in a queue and returns a success response without additional information
• Lambda queues the event for processing and returns a response immediately
• we can configure lambda to send an invocation record to another service like SQS,SNS,Lambda and eventbridge

Here is list of functions that invoke lambda function asynchronously
☐ Amazon S3
☐ SNS
☐ SES
☐ Cloudformation
☐ Cloudwatch logs
☐ Cloudwatch events

☐ AWS codecommit
☐ Aws config

Poll-based Invocation :

• This invocation model is designed to allow us to integrate with aws stream and queue based service with no code or server management lambda will poll the following service on our behalf,retrive records and invoke our function
• The following are supported services
☐ Amazon Kinesis
☐ SQS
☐ DynamaDB streams

Session 3 : Setup S3 Trigger with Lambda(Lab)