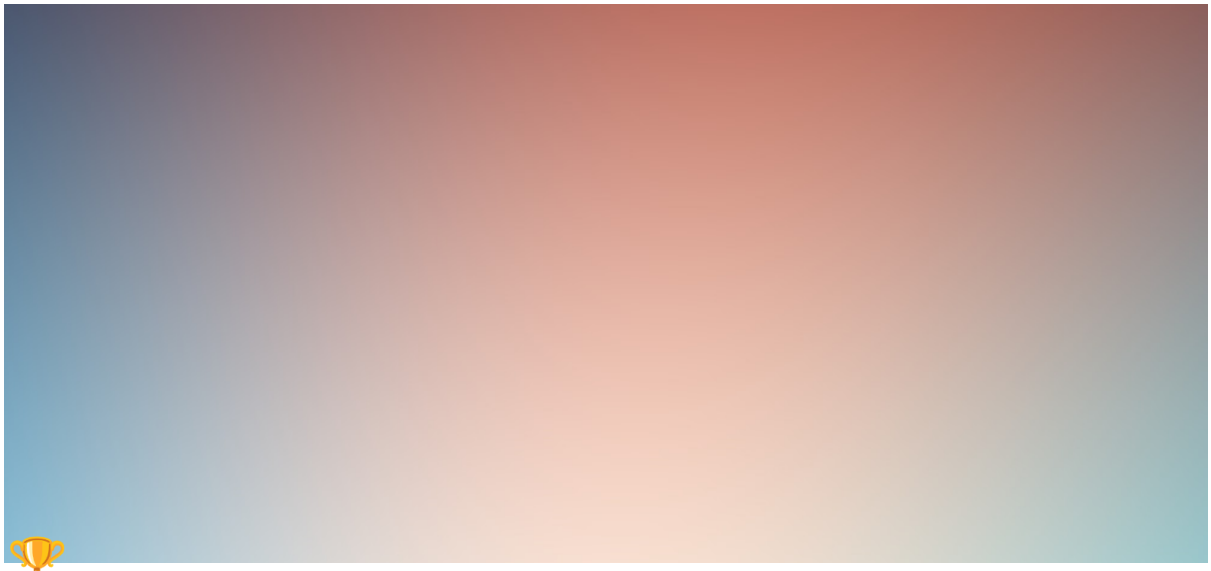


AWS Essentials

▼ **Singh Rounak.**

Foundational Services:

1.Compute

2.Storage

3.Network

Notes:

EC2, VPC, S3 and EBS(Elastic Block Store)

DynamoDB, RDS

IAM

Elastic Cloud Compute:

Virtualized servers running on AWS Data centers

Virtual Computing - Scale Capacity/Pay only for usage/Linux or Win/Deploy across Availability zones for reliability.

S3	EBS	EC2 instance store
Object level storage medium.	Network attached hard drives that are accessible at the block level	Local storage
Pay what you use	Pay for what you provision, (use or not)	

S3 Bucket:

Data gets stored in multiple locations (Accessible at anytime from anywhere.)

You can control access to buckets and objects with Access Control Lists. (ACLs)

Bucket Policies (add or deny permissions) - enabling centralized management of permissions.

Identity and access management (IAMs)

Upload and download data to S3 vis SSL encrypted endpoints (accessible from internet and within Amazon).

Customers can also use their own tools for encryption (Client encryption libraries)

You can encrypt data using ASW SDKs

Data Transfer

Versioning -

- protects data from accidental damage and deletes with no performance penalty.
- Generates new version with every upload.
- Easy retrieval of deleted objects or rollback to previous versions.
- Data once versioned, cannot go back to un-versioned.

States of S3:

1.Un-versioned

2.Versioning-enabled

3.Versioning-Suspended

Storage Classes:

1.S3 Standard (Durability 99%, Availability 99%)

2.S3 Standard IA (Durability 99%, Availability 99%) Retrieval fee associated with objects

3.Glacier (Durability 99%, Availability 99%) No realtime access, Must store objects before you can access them.

S3 Object lifecycle :

Log files, archive docs, Digital media archives, Financial and healthcare records, raw genomics sequence Data, Long Term data backups, Data retained for security compliance.

Delete certain files after use continuously.

→ Glacier provides long term, low-cost archiving service which is optimal for infrequently accessed data. (3-5 hours retrieval time). as low as @0.01$/GB per month.

Eg - Soundcloud Application allows users to upload 12 hrs of recording everyday. It uses a mix of Glacier(infrequent data) and S3(active user data).

Amazon Block Storage - AWS EBS (DISK DRIVE with file system ,cannot be accessed via Internet)

→ Provides block level storage volumes for offering low -latency performance

→ Automatically replicated in Availability zone. (Highly available and durable, independent from live instance)

→ Snapshots stored in S3.

Life cycle :

→**Create**(Vol 1GB to 16 TB) - Gen Purpose(SSD and provisioned IOPS-SSD) and Magnetic option

→ **Attach** (attach volume to particular EC2 instance)

→ **Attached and in-use** (Format from EC2 instance OS and mount formatted drive)

→ **Create Snapshot**(Snapshot to S3)

→ **Detach**(Call Detach Volume)

→ Delete Volume

You can use encrypted volumes and also create point in time snapshots.

EBS decouples the data persistence from the lifecycle of an EC2 instance.

Use Cases - OS(boot/root volumes)/ Databases(scales with performance needs)/ Enterprise Applications(block storage for mission critical applications)/Business Continuity and Applications.

AWS LAMBDA:

1. It is one of the AWS Services that falls under the ‘Compute’ domian along with EC2, EBS, ELB.
2. Allows us to execute code virtually for any type of service or backend application. Just supply the code to Lambda in anyone of the languages that AWS Lambda supports.(Node.js, Java, Python, C# etc)
3. Also runs code in response to certain events from other services and relating to those events, Lambda can also run certain function written in supporting language.
4. Use cases - Process Images (Loading objects in S3 bucket), Analyzing social media data, etc

Working -

Clients send data to Lambda as requests and Lambda defines the number of container to process data based on the number of incoming requests.

The containers contain the code that the user has provided to process the data or to satisfy the query (Container has image of the full-fledge application required to process data that is isolated from its environment)

When requests increase, the volumes and size of data processing increases. So more number of containers are created with provisioned services and when the requests reduce, the number of containers decrease as well. (This helps in saving resources.) We are only charged for the amount of time that our application is running in the containers within the lambda function (usually in milliseconds).

USE CASE 1:

Near realtime backups are not possible as manual backup is not possible

Lambda can help in realtime backup by creating 2 S3 buckets (source(upload data) and destination(backup data)) for these buckets to talk to themselves IAM roles and policies are required and AWS Lambda function copies files between buckets.

Process - whenever there is a change in the Metadata in the source bucket (i.e when data is uploaded in source), the Lambda function gets triggered and starts copying data to the destination bucket. After this we can test it buy actually putting some data in source bucket

The same can be done done for buckets belonging to 2 different accounts. This is done by configuring the IAM roles and policies

Differences between Lambda and EC2 -

Elastic Cloud Compute	**Lambda**
Servers provisioned to run services .Started 2006	A compute service that lets you run code without provisioning or managing servers.
Used to deploy resources to host web apps and services. (Renting). High-availability must be established by the user for various AZs.	Inherently Highly Available. (Services are deployed into multiple A.Zs automatically.)
Supports variety of OS's. Backbone of multiple services. Pay for entire EC2, even if it is idle for half the time.	Pay for what you use, never pay for idle
User handles scale according to needs. However, autoscaling can be set.	Autoscales with usage automatically.
Deployed for backend service	Deployed for backend service
infrastructure managed by user - VM size, OS, AMI etc. requires management and orchestration.	Underlying infrastructure managed by AWS.
More maintenance overhead	Less maintenance overhead
Can Install almost any Software	Can't Install Software. Only Libraries can be used.
More work to configure VM and OS.	Easy selection of compute and memory power
Hard disk attached to VM.	No Hard disk. Deployment packet size is limited.
Benefits - Complete control of compute environment. More focus on maintenance	Benefits - Easy to run services. More focus on solving the business problem as maintenance is not required.
Faster migration to cloud with other services.	Native Integration with other services (triggered by S3)
Best when usage is predictable	Best when usage is unpredictable

AWS RDS - Relational Database Service (https://www.youtube.com/watch?v=knrNBkr5iTM**)**

Databases are heart of many functioning business applications and become difficult to manage and operate with high availability as you scale your app - Installs,Patching,Monitoring,Performance tuning,Backups,Security)

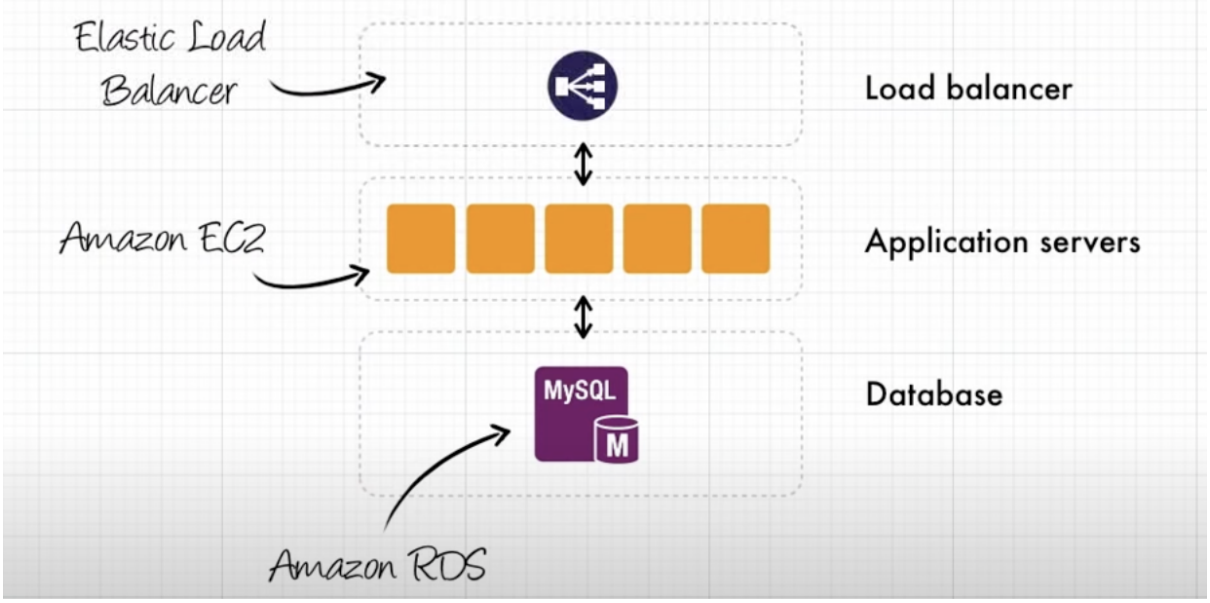
Database administration is resource intensive. **AWS RDS is a web service** simplifies this process by automating time-consuming administrative tasks creating less operational overheads.

Gives users the choice to use any database (SQL, NoSQL, PostGres, etc)

It provides the familiar capability of MySQL or Oracle databases without the mundane time consuming administrative tasks of RDBMS.

Provides Advanced features - with just few clicks and API calls

1. It supports rapid provisioning of relational databases in a utility computing environment.
2. RDS database instances are available on demand as, pay as you use service
3. Provide scalable storage capacities under the hood. Scale as you use.
4. Provide High Availability. So that Databases are up and running all the time.



→ Cost Efficient

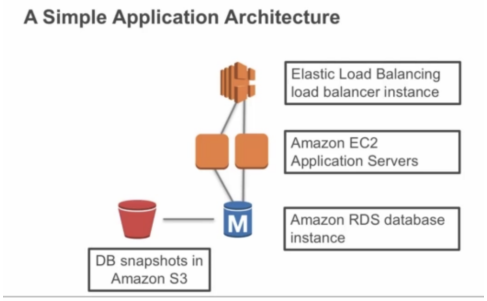
→ Access to full capabilities of Amazon Aurora, MySQL, MariaDB, Microsoft SQL Server, Oracle and PostgreSQL Databases.

→ Fast predictable performance

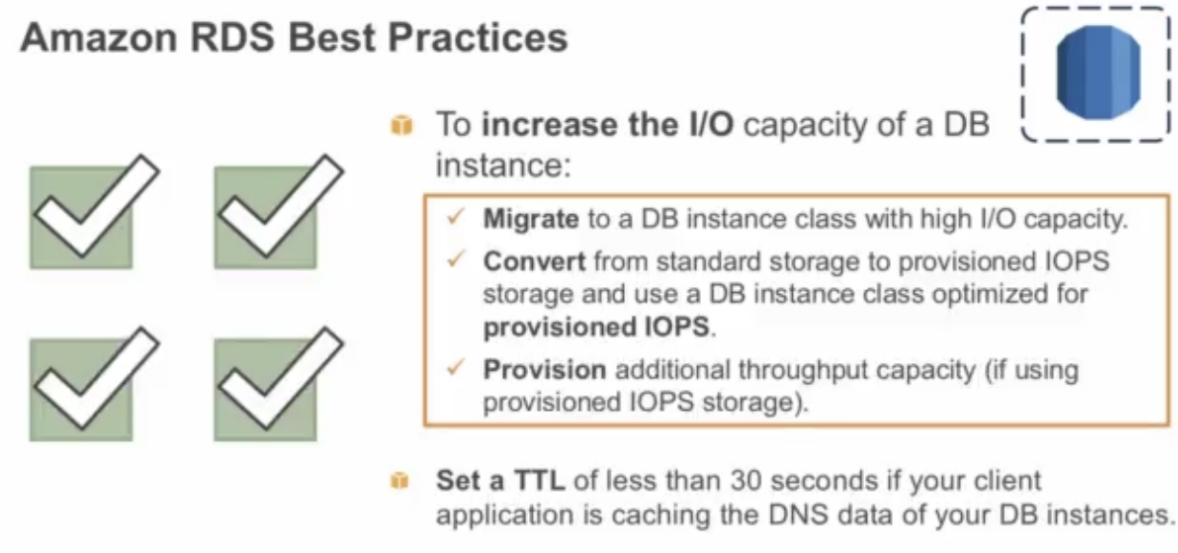
→ Creates Backups

1. **DB Instances -** Basic building blocks of RDS, isolated from db environment on the cloud. These contain multiple user-created databases. (created using - AWS management console, CLI, APIs)

2. **Automatic Backup -** Restores DB at a point in time, enabled by default. (Choose retention period upto 35 days).
    a. Manual Snapshots - let's a user build a new db instance from a snapshot. Initiated by user and persists until the user deletes it, Stored in S3.
3. **RDS Security -** Run your DB on VPC, utilize security groups and Policies to control users
4. **Architecture -**



5. **Best Practices -**
    a. Monitor memory, CPU and storage usage
    b. Multi- AZ deployments to automatically provision and maintain synchronous standby in a different Av zone.
    c. Enable automatic backups
    d. set the backup window to occur during daily low in WriteIOPS.



Deploying RDS + MySQL Workbench:

1. Go to AWS Management Console
2. Select RDS and click on ‘Create Database’
3. Select Standard create
4. ENGINE OPTIONS - Select ‘MySQL’ (default)
5. TYPE - Free Tier
6. IDENTIFIER- Name the instance identifier as ‘aws-simplified’.
7. CREDENTIALS - Create a Master Username and Password.
8. INSTANCE -Select instance size as Burstable classes (db.t2.micro)
9. STORAGE - select General purpose [SSD], allocated space = 20GiB, Autoscaling - enabled[max = 1000 GiB]
10. Availability and Durability - (Not Applicable for free tier)
11. Additional Connectivity Configuration - Subnet group[select*], Publicly accessible[Yes], VPC group[Choose existing]
12. Authentication - [Password Authentication] and Finally Click on ‘Create Database’

Your database instance will be created and you can find that on the Database section under Amazon RDS.

You can start using it when the status changes to Available.

With VPC Multi AZ Deployment - https://hexaware.udemy.com/course/aws-essentials-z/learn/lecture/6766302#overview

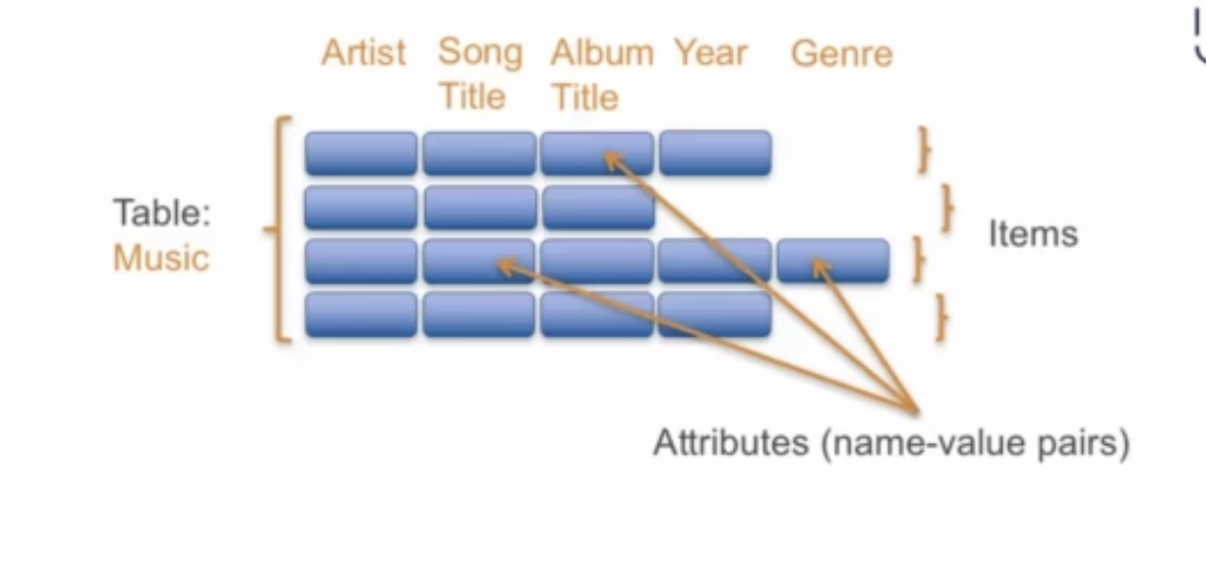
DynamoDB - NoSQL Database Service

→ No limit

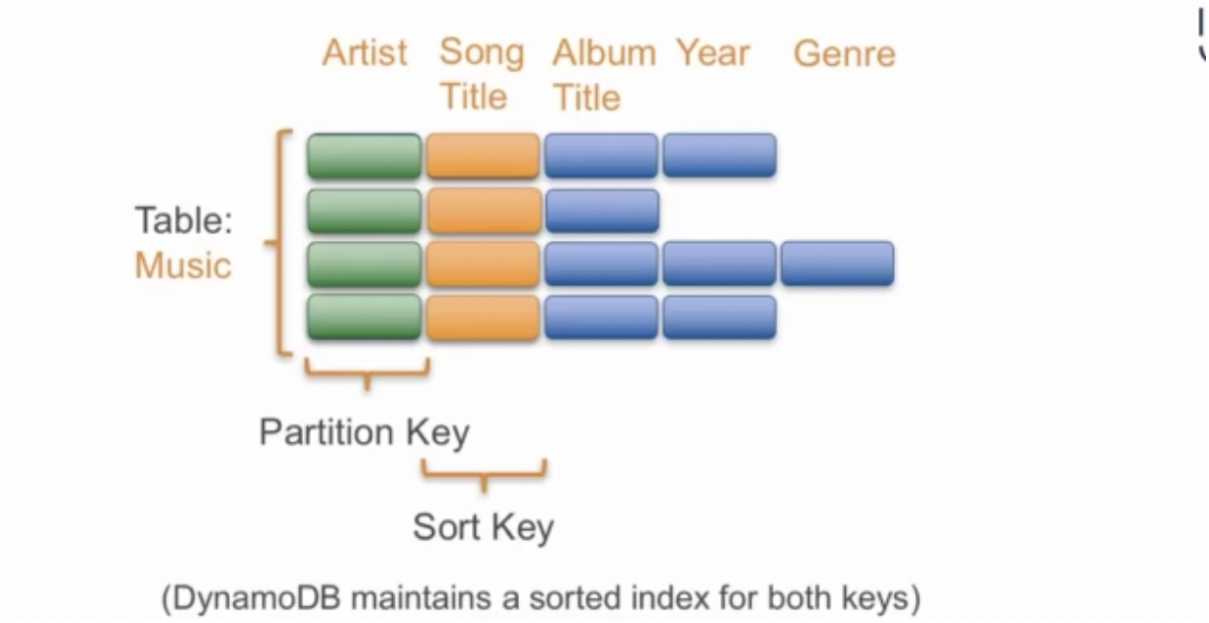
→ Fast, Predictable performance using SSDs

→ Easy Provision and change of request capacity

data example - JSON document



Primary Keys - 1. Partition key 2.Composite key



Provisioned Throughput -

→ Users specify how much provisioned throughput capacity is needed for reads and writes

→ Amazon DynamoDB allocates necessary machine resources to meet our needs.

Supported Operations - 1. Query, 2. Scan

Amazon RDS and Amazon DynamoDB

Factors	Relational (Amazon RDS)	NoSQL (Amazon DynamoDB)
Application Type	• Existing database apps • Business process–centric apps	• New web-scale applications • Large number of small writes and reads
Application Characteristics	• **Relational** data models, transactions • **Complex** queries, joins, and updates	• Simple data models, transactions • Range queries, simple updates
Scaling	Application or **DBA–architected** (clustering, partitions, sharding)	**Seamless, on-demand scaling** based on application requirements
QoS	• Performance–depends on data model, indexing, query, and storage optimization • Reliability and availability • Durability	• Performance–**Automatically optimized** by the system • Reliability and availability • Durability

---

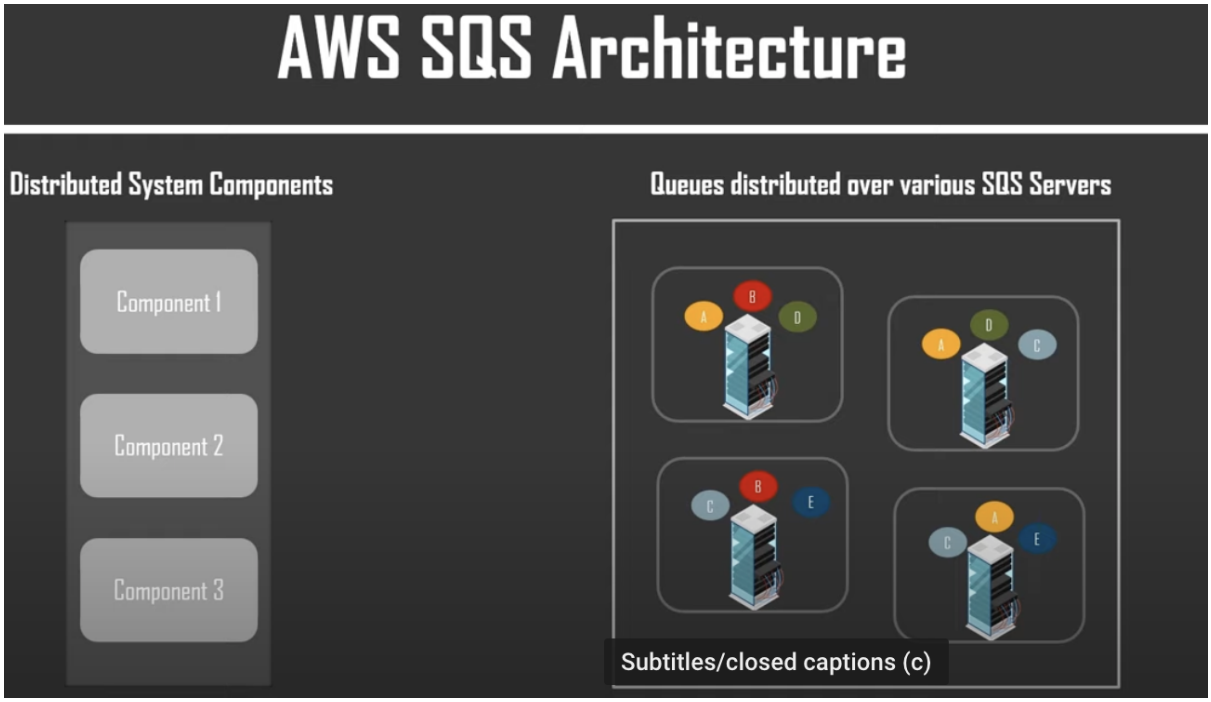
AWS SQS - Simple Queue Service

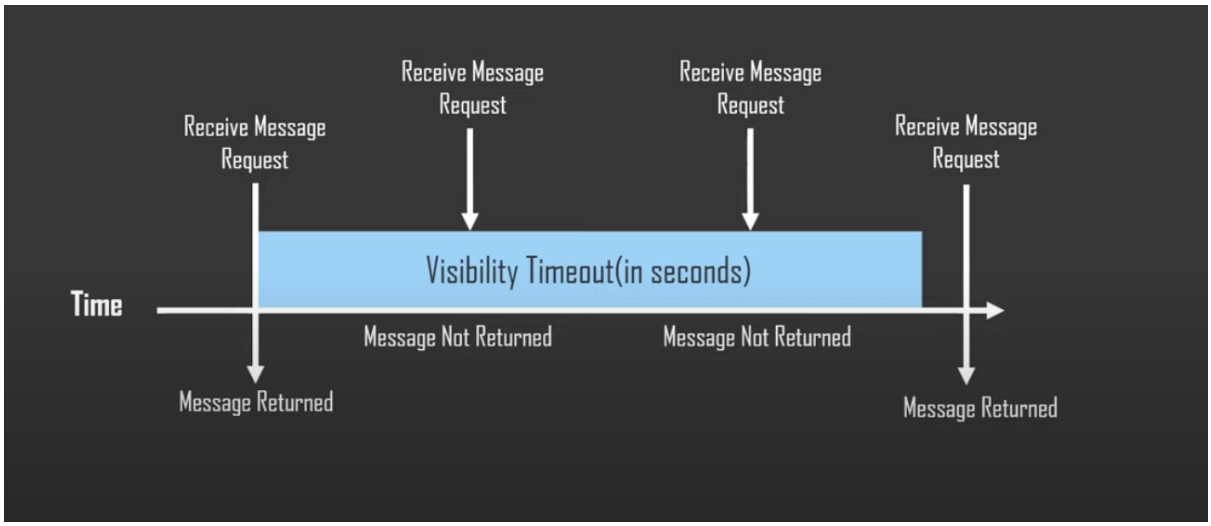
1. Fully managed message queueing service that allows users to separate and scale the microservices, distributed systems, and serverless applications.

2. Queuing service for message processing. A system must poll the queue to discover new events. Messages are processed by a single consumer.
3. Usage depends on weather your system care about an event?

Sends and receives messages between software components at any volume

Makes sure messages are not lost during the process and reach the consumer on time.





AWS SNS - Simple Notification Service

1. Uses Publisher/Subscriber system - A user may own a topic and publish to that topic and the subscribers get notified for the events that are delivered.
2. Fan-Out approach -Publishing messages can deliver to many subscribers of many types(SQS, Lambda, Email)
3. Usage depends on whether other system care about an event?

USE CASE for SQS and SNS:

1. Credit Card Transactions - purchase request for some REST API.

AWS SECURITY SERVICES:

You can control and access to your buckets and objects using

1. ACLs (Access Control Lists) - To let other AWS accounts and not specific users access your account
2. Bucket Policies - Add/Deny permissions across some or all objects contained in an S3 bucket.
3. Identify and Access Management Policies (IAM) policies - Only users within your AWS account

Upload/Download data to S3 vis SSL encrypted endpoints.

→ Encrypt Data using AWS SDK.

→ Policies can be attached to users, groups or S3 buckets `enabling centralized management of permissions.

→ Encryption and Decryption can be enabled by both Client and Severs when using S3. (Adds additional Security).

→ Some buckets with encryption can only upload and store encrypted data

VPC and SUBNETS:

Subnet - A range of IP addresses in your VPC.

Subnets cannot span multiple availability zones.

AWS VPC

Monday, February 14, 2022

6:09 PM

VPC - Virtual private cloud.

Each AWS account is shipped with a default VPC and a subnet. VPCs are isolated network in a region in the cloud. When you create a VPC, you need to define all the configuration details.

It is your own network in the cloud that have various availability zones which can be utilized to run EC2 instances for hosting applications or databases.

The Availability zones are the different regions in the cloud (physical locations of AWS servers like Virginia-east-001, etc)

Subnets - these are sub-networks inside the VPC and each subnet can be one availability cell. VPC can have multiple subnets.

You can create your own VPC and create various EC2 machines on different subnets to perform certain functions as per your requirements.

We set up our own IP range in the VPC (internal IPs). You can assign public Ips to the instances that you are running in the subnets, This way, your instances receive internal Ips (reaching within the network) and the public IPs (reaching from outside the network)

Each subnet takes the subset of the internal IP range. For eg 111.000 to 11.10 10 ….. 111.120.120 etc

All the internet access happens from the internet gateway. This gateway is important for all the traffic leaving your VPC and managing your connection of your Private cloud network with the outside internet.

Access Control:

You can assign

On Lower Level

1. Security groups to your VPCs - virtual Firewall applied on per instance level.

On Higher Level

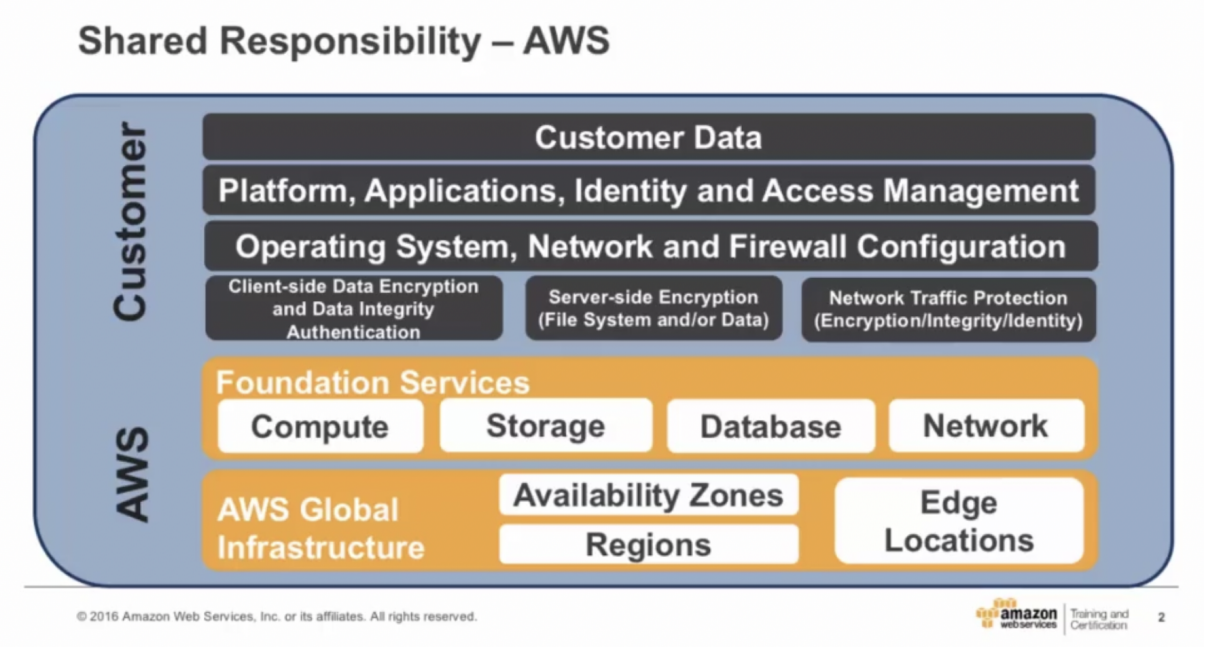
1. Network Access Control Lists (NACL) - these apply on subnet level. Which traffic can enter subnets

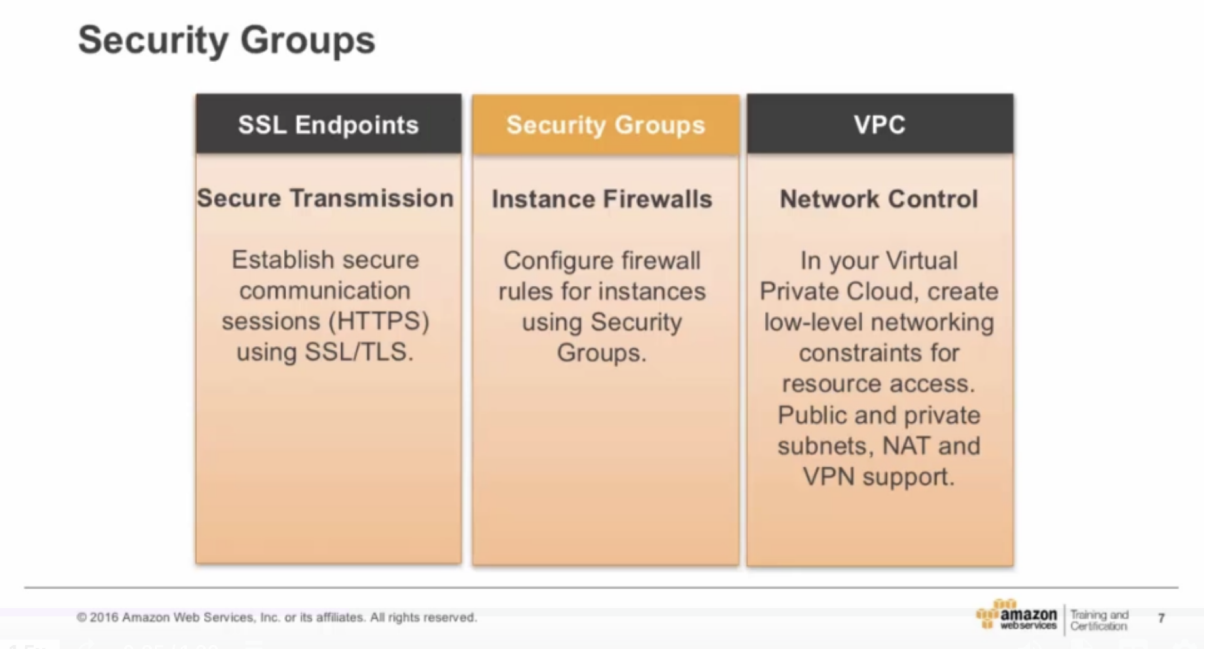
You can combine both SGs and NACLs to control access to your VPC

1. You can also control routing - In case your firewall is breached or broken , you can route the traffic to a proxy using route tables which you get by default when you create a VPC.

The traffic can be sent to a different instance, allowed to leave, sent to a proxy - however you set up your route table.

1. You can also control the assignment of Public IP's

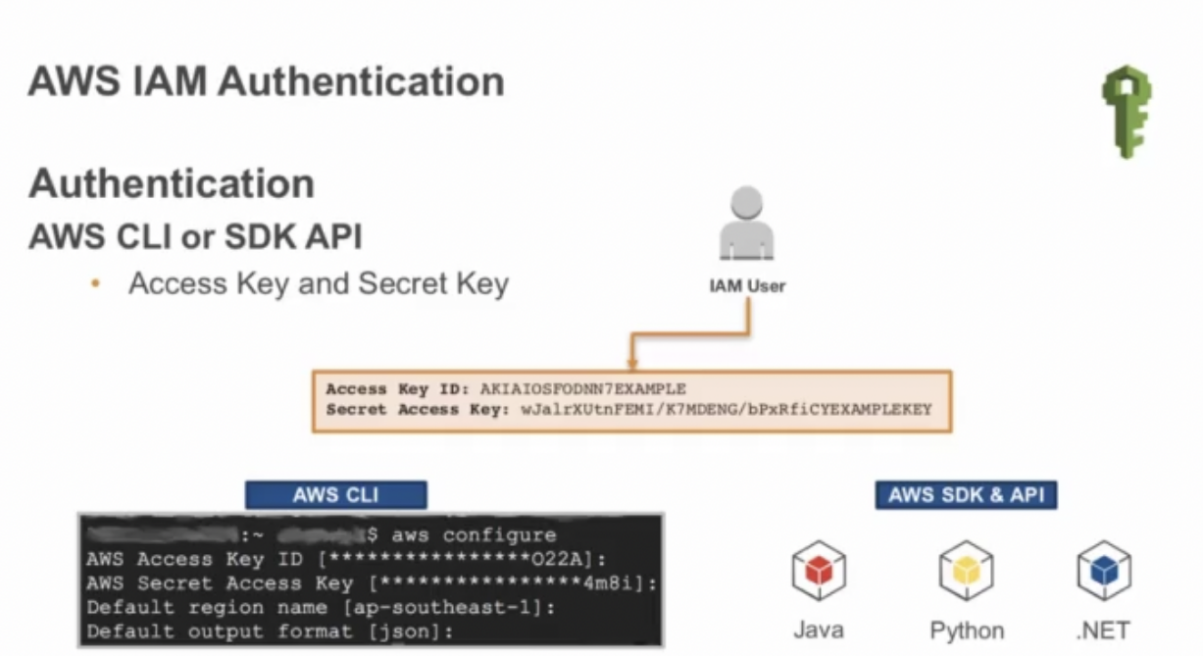




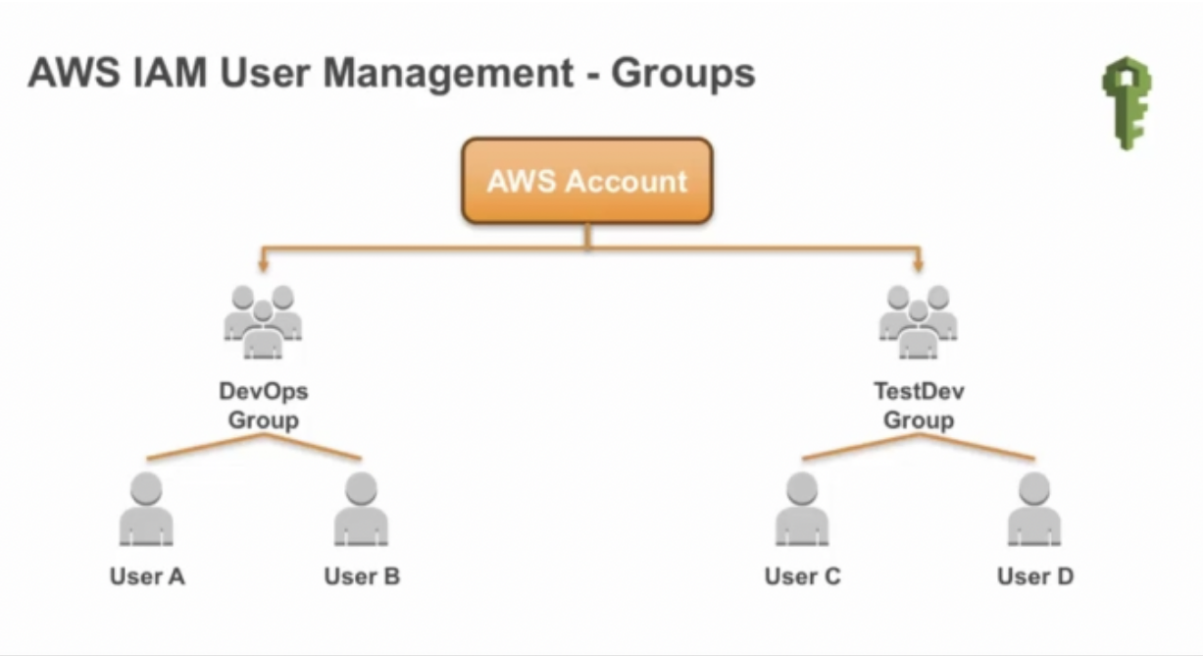
IAM is a free complimentary service.

AWS IAM Authentication - Done using AWS CLI or SDK API.

Has an Access key and a Secret Key



We can also assign permissions to groups.



After users and groups are authenticated, they then have to be authorized to use the underlying AWS services.

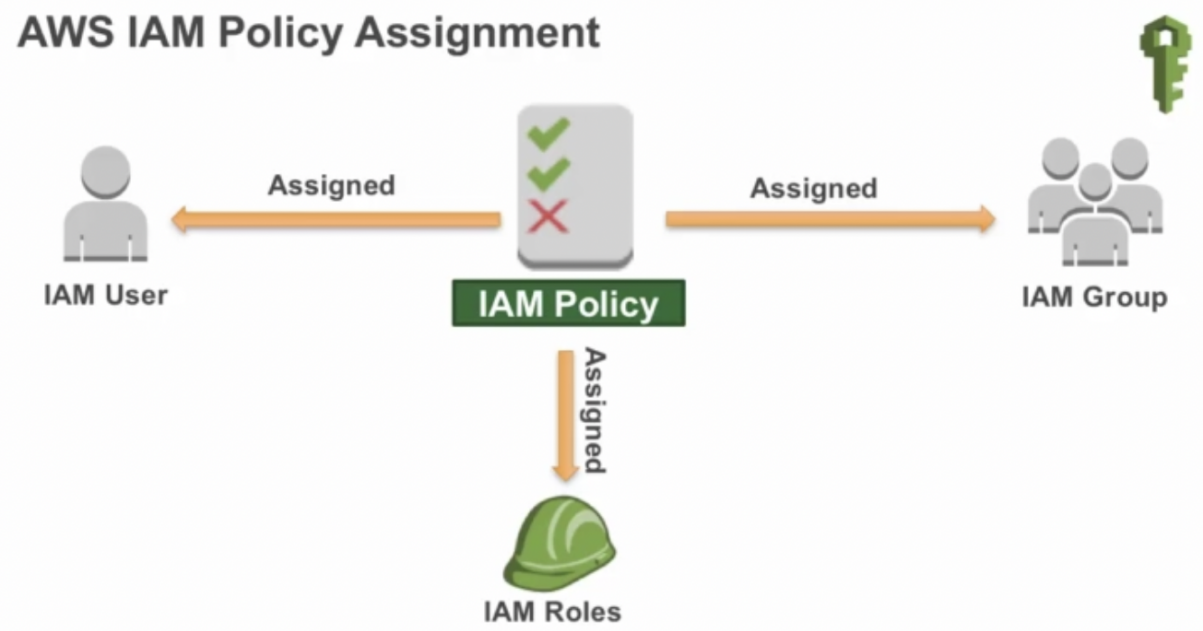
→ You assign permissions by creating policies [JSON format docs]

→ An IAM role is AWS identity with permission policies that determine what what a user can or cannot use on AWS

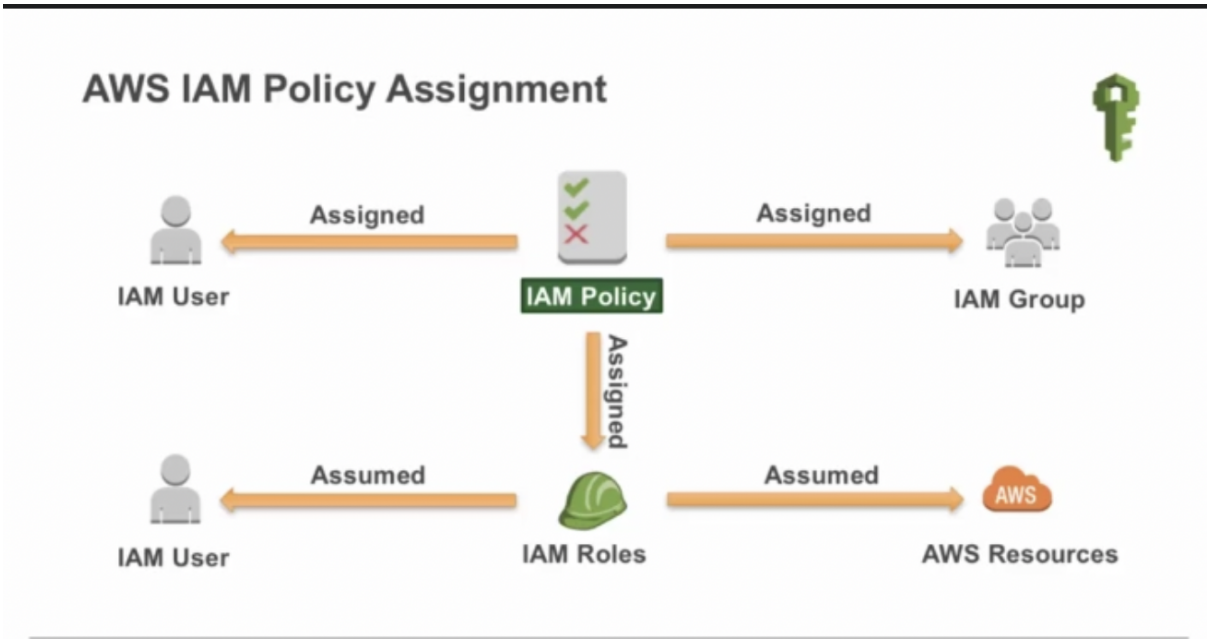
→ A role is assumable by anyone who needs it and not directly correlated with a person.

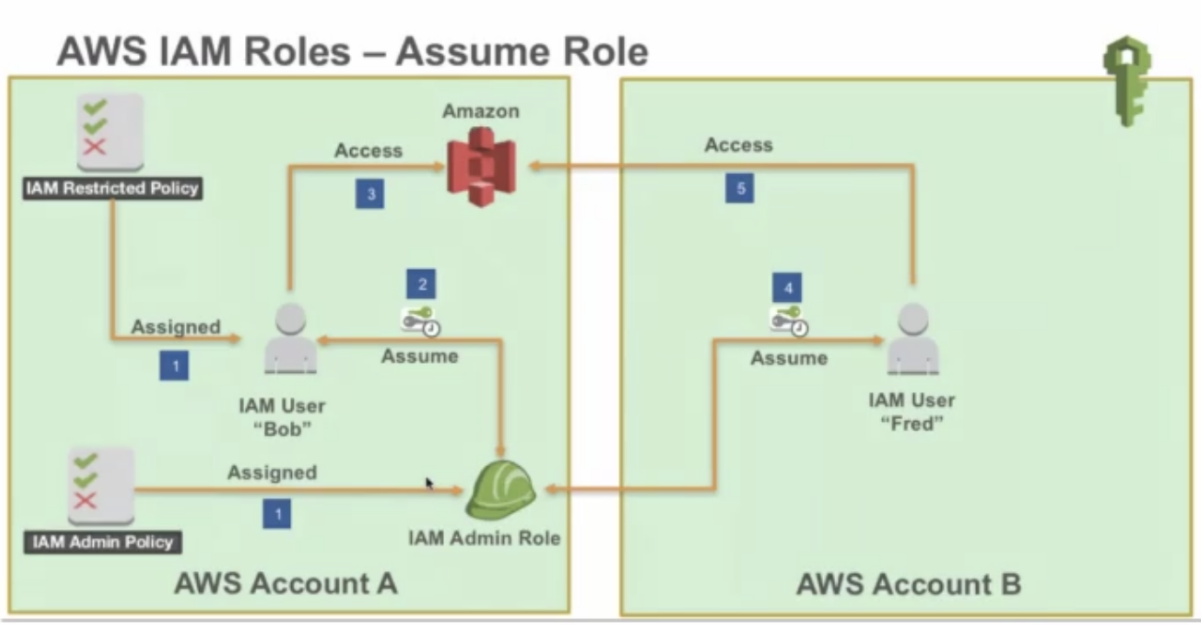
Access keys are created dynamically and the Role does not inherently associated with it.

*** A single IAM policy can be assigned in multiple different places.

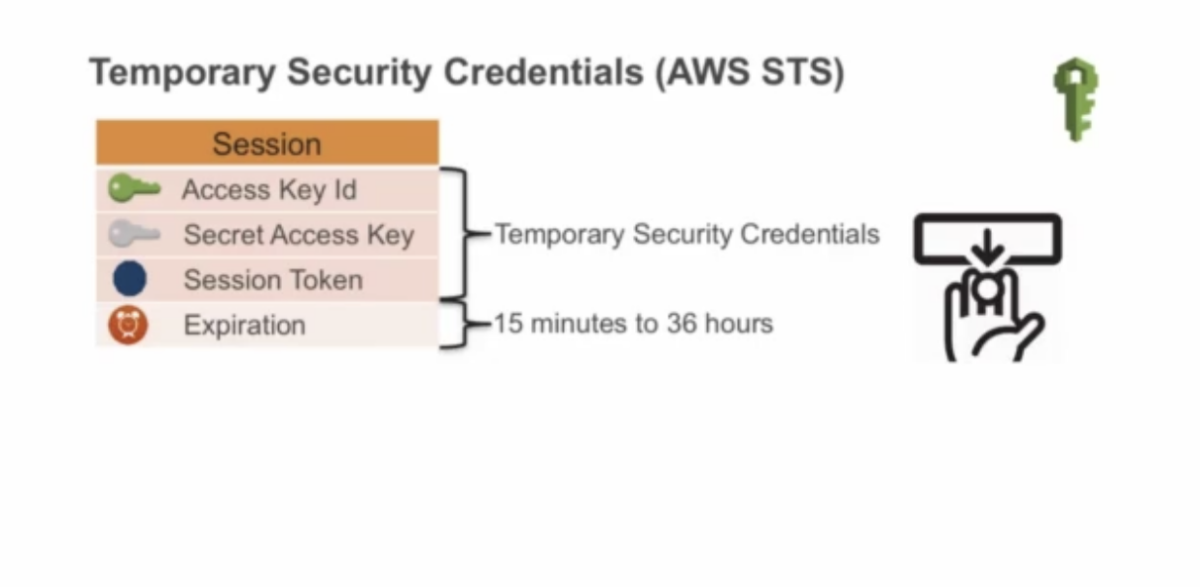


Policies are assigned, Roles are assumed





Temporary Security credentials are generated using AWS STS - Security Token Services.



BEST PRACTICES - AWS IAM

→ Delete AWS account's root access keys.

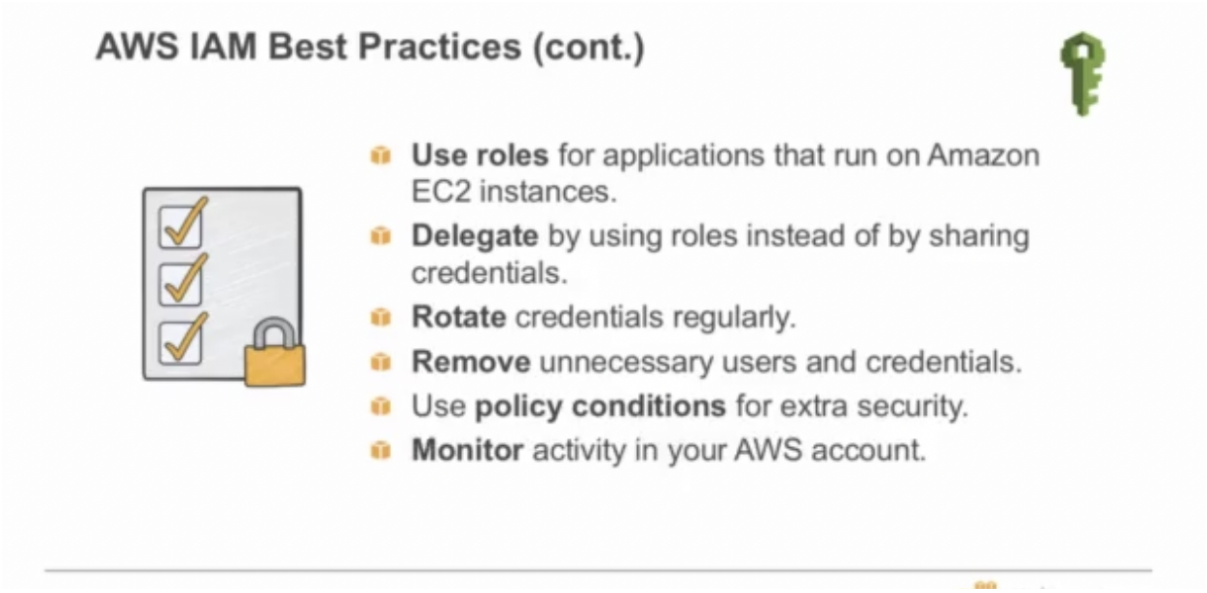
→ Create Individual IAM users.

→ User groups to assign permissions to IAM users.

→ Grant least privilege

→ Configure a strong password policy

→ Enable MFA for privileged users



Module 4

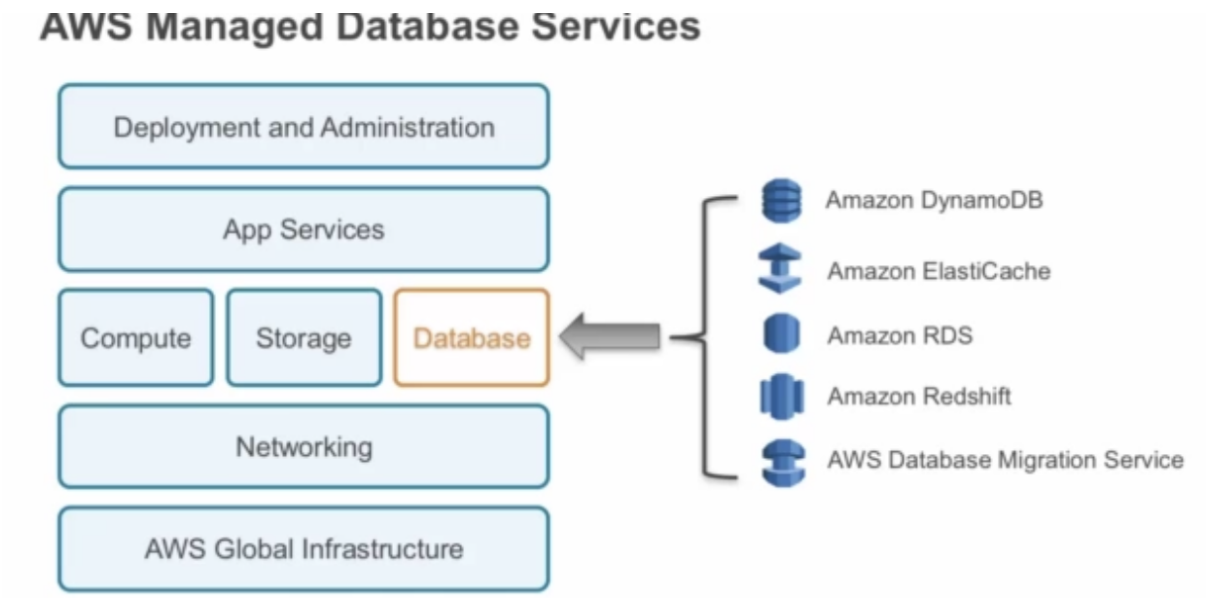
AWS DATABASES -

NoSQL vs SQL

RDS

DynamoDB

	SQL	NoSQL
Data Storage	Rows, Columns	KV pairs
Schemas	Fixed	Dynamic (schema less)
Querying	Using S.Q.L	Focused on collection of documents
Scalability	Vertical	Horizontal



ELASTICITY and MANAGEMENT of AWS Servcies:

1. AUTO-SCALING
2. ELASTIC LOAD BALANCING
3. CLOUDWATCH
4. TRUSTED ADVISOR

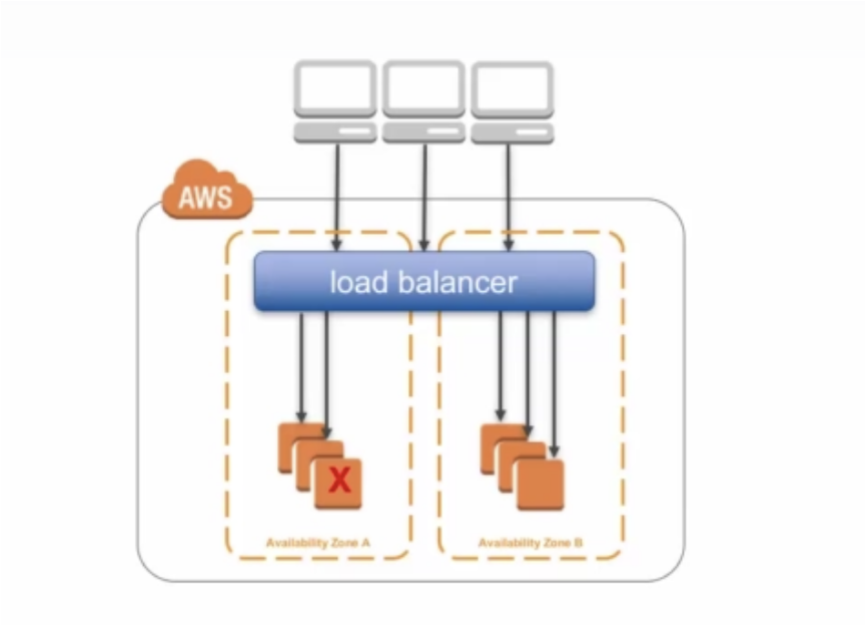
EG. Two Backend services are registered with the load balancer in EC2 instances will be sending data in the form of metrics (latency, CPU performance) to amazon CloudWatch. We can set alarms if any of the metrics exceed a set value.

This alarm triggers the autoscaling policy. Autoscaling creates new instances to balance the load.

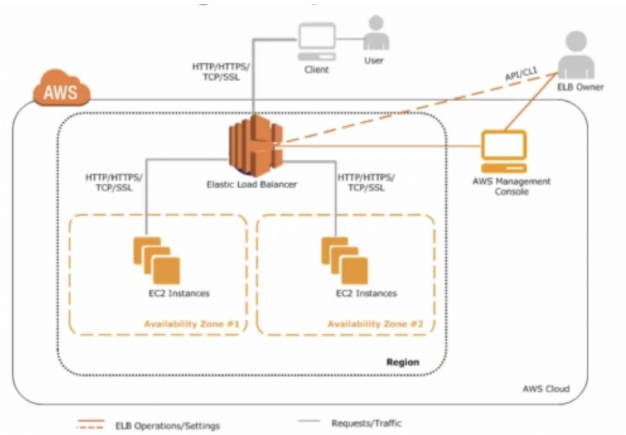
1. AUTO SCALING. (What, When , Where)
    a. **What** - Launch Configurations - template used to launch EC2 instances that includes configuration details (AMI iD, Instance type, Keypair, Security Groups, Block Device Mapping, User Data)
    b. **When** - Autoscaling group -
    c. **Where** - Autoscaling lifecycle

    - Works best for applications that experience variability in usage on a daily basis.

    It provides better fault tolerance. Utilize multiple av-zones., ensures better availability, better cost management as it



2. Load Balancers - Perform Health checks on the instances continuously and reports its state. The health status is used in conjunction with autoscaling which replaces the instance if the health is poor. This forms a auto healing architecture where services are monitored continuously and repairs are made regularly in terms of replacement.



Selecting the Load Balance Type-

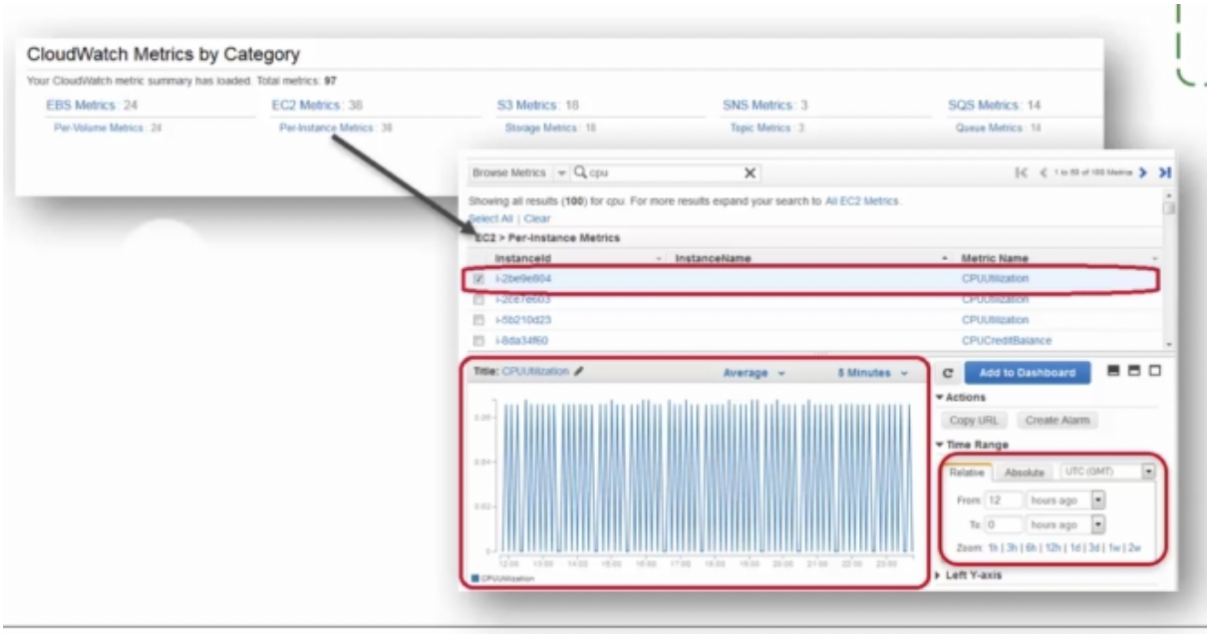
1. Internet Facing - the load balancer accepts incoming reqs directly from clients from internet and routes them to backend instances
2. Internal - the load balancer is configured to be internal. All incoming reqs are generated within VPC and LB does not connect to the internet.

No matter what kind of LB we choose, we have the option of creating an HTTP Load Balancing. It uses an SSl-TLS protocol, LB has the option of accepting incoming reqs directly from clients from internet and routes them to backend instances (or) re-encrypting and sending it over HTTPS.

AMAZON CLOUDWATCH

Enables us to monitor our AWS resources in near real-time.

Collects and reports data once every 5 mins (detailed metrics at 1 min frequency). The right hand corner shows time based settings for monitoring.



We can set alarms. for initiating Autoscaling and Load balancing based on thresholds we set.

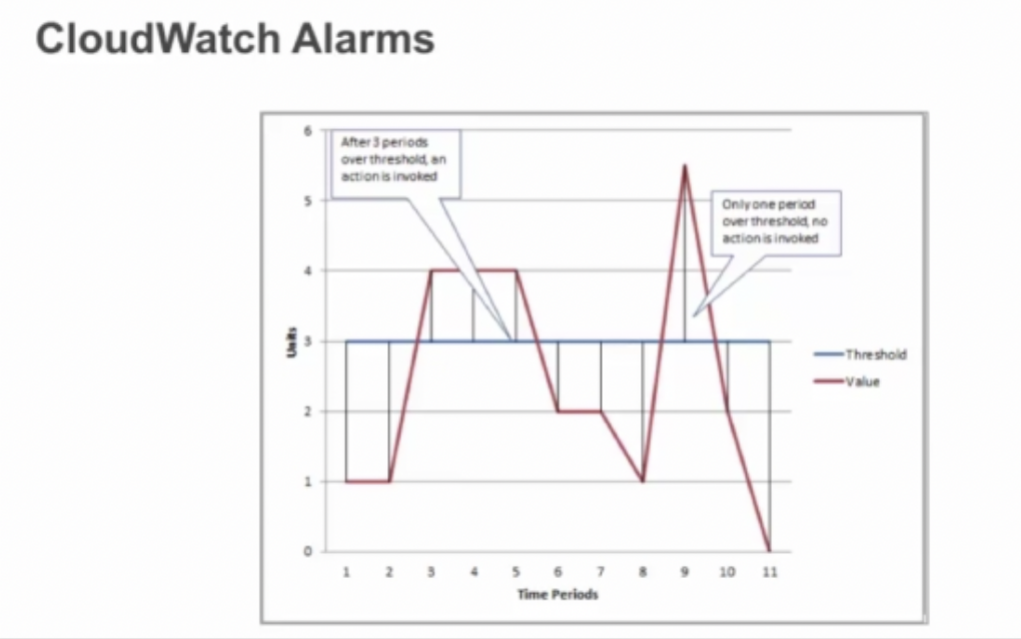
It is a metrics repository.

Monitoring Service for resources and applications run on AWS.

Gives visibility into resource utilization, operational performance and demand patterns.

Cannot pull metrics on its own.

Alarms watch continuously - action is a notification sent to the SNS topic our autoscaling policy. sustained changes only trigger alarms. short fluctuations and odd values are ignored.



The supported services that send metric data to AWS Cloudwatch are as follows:

AWS TRUSTED ADVISOR - Recommendation Engine for Best Practices

Performs checks in 4 categories:

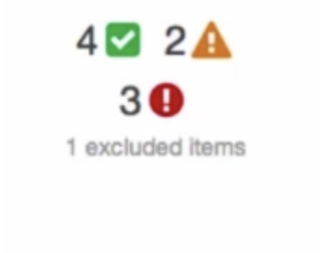
1. Cost Optimization
2. Security
3. Fault Tolerance and
4. Performance Improvements

Checks are reported on a dashboard in a 3 color scheme

RED - Action is recommended

YELLOW - Investigate a finding

GREEN - Okay, nothing needs to be done



Helps in saving money

examples -

a. AWS EC2 Reserved Instance Optimization
b. Low Utilization AWS EC2 instances
c. Idle Load Balancers
d. Underutilized EBS Volumes
e. Unassociated Elastic IP Addresses. (IP address usage charges more money if not checked)
f. RDS Idle DB instances.

Checks security groups - unrestricted access-reduces opportunities for attacks

Checks -

1. IAM use, Password Policy
2. cS3 bucket permissions ,
3. MFA on root account
4. RDS permissions

Fault Tolerance- following checks are made by the trusted advisor:

1. Makes sure we are using Recent EBS snapshots
2. Making sure our load balancers are set to balance across cross availability zones
3. Autoscaling resources are also set to move across cross aval. zones
4. Making Sure RDS is set to Multi AZ.
5. Correct Route 53 Name Server Designations.

6. ELB connection Draining.

Performance improvements

1. High utilization EC2 instances, throughput optimization
2. Service Limits
3. Larger no. of rules in Security Groups
4. Over utilized EBS Magnetic volumes
5. CloudFront Alternate Domain names.

Scaling and Load Balancing a Web Application-
https://hexaware.udemy.com/course/aws-essentials-z/learn/lecture/6768964#overview