**AWS**

**Amazon web services**

**CONTENTS**

|  |  |
| --- | --- |
| **Chapter** | **Topics** |
| Global Infrastructure | A Conceptual Introduction to Amazon Web Services (AWS)  Regions  Availability Zones  End points  Creating an AWS Account Usage Tracking Billing |
| IAM | IAM Essentials  IAM Policies  IAM Users  IAM Groups  IAM Roles |
| Cloud Watch | Alarms  Events  Monitoring  Logging |
| AWS Simple Notification Service SNS | Introduction to SNS |
| CloudTrail | Understanding the concepts CloudTrail |
| VPC | Basic understanding of VPC  Subnets  Route Table  Internet Gateway  NAT Gateway  Security in VPC: NACL  VPC Peering  VPC Endpoints  Limitation of VPC |
| Server Based Compute (EC2) Fundamentals | EC2 Architecture, Instance, Types and Sizes  EC2 instance purchasing types  Instance Roles  AMI’s  Ec2 Storage Architecture  EBS, Types  EFS  Overview of Different types of storage classes  Snapshots  Life cycle manager  Security Groups  Elastic IPs  Key pairs  Load Balancing  Autoscaling  Route 53 |

|  |  |
| --- | --- |
| **Chapter** | **Topics** |
| S3 | Understanding S3  S3 Naming Convention  Transferring Data to S3  Storage classes in S3  Lifecycle policy in S3  Cost optimization for S3  Versioning in S3  Encryption in S3  Static website and CORS |
|  |  |

**Q&A:**

**Virtualization:**

act of creating a virtual (rather than actual) version of something, including virtual computer hardware platforms, storage devices, and computer network resources

**Hypervisor:**

is a software, hardware, or firmware that generally make virtualization possible. It simply creates virtualization layers that separates CPU/processors, RAM and all other physical resources from virtual machines that we create.

**SSL certificate**

* An SSL certificate is a digital certificate that authenticates a website's identity and enables an encrypted connection. SSL stands for Secure Sockets Layer, a security protocol that creates an encrypted link between a web server and a web browser.
* The key difference between SSH vs SSL is that SSH is used for creating a secure tunnel to another computer from which you can issue commands, transfer data, etc. On the other end, SSL is used for securely transferring data between two parties – it does not let you issue commands as you can with SSH.

**Cloud services**

* **IaaS**: cloud-based services, pay-as-you-go for services such as storage, networking, and virtualization. AWS EC2, Rackspace, Google Compute Engine (GCE), Digital Ocean, Magento 1 Enterprise Edition
* **PaaS**: hardware and software tools available over the internet. AWS Elastic Beanstalk, Heroku, Windows Azure (mostly used as PaaS), Force.com, OpenShift, Apache Stratos, Magento Commerce Cloud.
* **SaaS**: software that’s available via a third-party over the internet. BigCommerce, Google Apps, Salesforce, Dropbox, MailChimp, ZenDesk, DocuSign, Slack, Hubspot

**LAN**

A local area network is a computer network that interconnects computers within a limited area such as a residence, school, laboratory, university campus or office building. By contrast, a wide area network not only covers a larger geographic distance, but also generally involves leased telecommunication circuits.

**(1)**

**GLOBAL INFRASTRUCTURE**

1. **Aws Free tier account and monitoring**

Billing dashboard

go to preferences and enable the receive free tier usage alert

1. **Global infrastructure**

We can access AWS i) AWS console ii) AWS cli iii) SDKs

1. **AWS Region:**

* *AWS Region* is a separate geographic area where we cluster data centers.
* Each AWS Region is completely independent.
* Choosing AWS region
  + Compliance
  + Proximity
  + Available services
  + Pricing

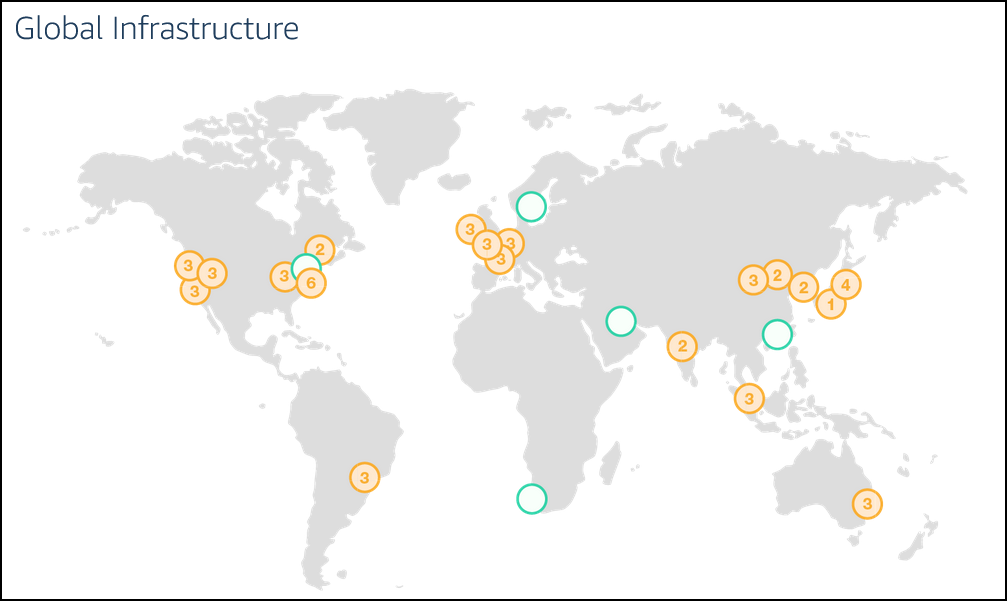
**Availability zones:**

Each **AWS Region** consists of multiple, isolated, and physically separate AZ's within a geographic area.

* Min. 2, max. 6
* Avoid disasters
* Redundant
* AZs of particular region are connected to each other with high bandwidth, ultra-low latency networking

**Edge Locations:**

* Data centers where end users can access services (frequently accessed) located at AWS regions to reduce latency...



1. **Compute section:**

EC2 Elastic compute cloud : Virtual machine

ECS Elastic Container service : Container as a service.

EBS Elastic beanstalk : PAAS solution manages infrastructure for you.

(Amazon ECS is a fully managed container orchestration service that makes it easy for you to deploy, manage, and scale containerized applications.)

AWS Lambda : Serverless computing platform.

(easy-to-use service for deploying and scaling web applications and services developed with Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker on familiar servers such as Apache, Nginx, Passenger, and IIS.You can simply upload your code and Elastic Beanstalk automatically handles the deployment, from capacity provisioning, load balancing, auto-scaling to application health monitoring)

1. **Storage overview:**

**Databases:**

SQL Database - RDS,

NOSQL Database - DynamoDB, Elastic cache, Redshift.

**Storage:**

AWS S3 object storage service file used in application storing file and retrieving files.

**(2)**

**IAM (Identity and Access Management)**

* IAM provides access to accounts services where we can manage User, Roles, Groups & Policy password policy.
* It applies globally to all AWS regions.
* In AWS you apply least privilege principle.
* Create an administrator account which has almost all permissions as root user for safety

**Users**: we create users and assign necessary permissions to them in the form of policies.

**Groups:** We can create groups for ex. Dev QA etc. and attach policies at the group level.

**Policy:**

A policy is a set of permissions

Always explicit deny overrides explicit allow

JSON document

**{**

**"Version": "2012-10-17",**

**"Statement": [**

**{**

**"Effect": "Allow", (allow or deny)**

**"Principal":{“acc/user/role to which policy is applied”},**

**"Action": \*, (list of actions this policy allows or denies)**

**"Resource": "\*" (list of resources to which the actions applied to)**

**},**

**Policy types:**

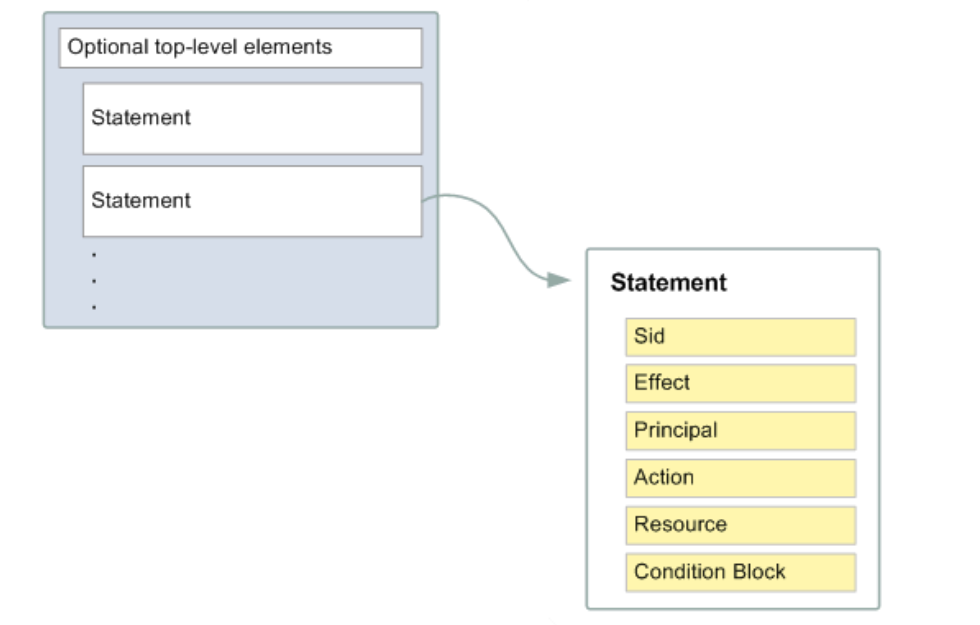
* **Identity Based Policy:** Applicable on users, groups of users, and roles
* AWS Managed policy: predefined restrictions
* Custom Managed Policy:
* Inline Policy: An inline policy is a policy that's embedded in an IAM identity (a user, group, or role). That is, the policy is an inherent part of the identity. You can create a policy and embed it in an identity, either when you create the identity or later.
* **Resource Based policy:** Attach to a resource such as an Amazon S3 bucket
* **Session based Policy:** create a temporary session for a role or federated user

Imp Notes:

More than one policy can be attached to a user or a group at the same time.

Policies can’t be attached directly to resources like EC2 instance, S3 bucket etc.,

**Basic Policy structure:**



**Effect**: Can take only two values allow or deny

**Principal:** who is assuming the policy

**Resource:** on whom you are assuming the policy

Ex:

{

"Version": "2012-10-17",

"Statement": [

{

"Sid": "FirstStatement",

"Effect": "Allow",

"Action": ["iam:ChangePassword"],

"Resource": "\*"

},

{

"Sid": "SecondStatement",

"Effect": "Allow",

"Action": "s3:ListAllMyBuckets",

"Resource": "\*"

},

]

}

Q: Can we attach multiple policies to user group or a role.

Yes, we can.

**Roles:**

A role is a set of permissions that grant access to actions and resources in AWS.

* Roles comes between services, like ec2 wants to access S3 or non-AWS user (hybrid account) should access AWS Resources.
* Policies can’t be attached to aws resources hence roles come into picture.
* EC2 can be attached one role at a time.

**Q:** Can we assign multiple roles to a EC2 instance?

No, we can’t. we can assign only single role to EC2 instance.

**Q:** If an ec2 instance is not able to access s3 bucket what could be the reason

A Role needs to be attached with proper policy defined.

**Assume Role:**

Returns a set of temporary security credentials that you can use to access AWS resources that you might not normally have access to. These temporary credentials consist of an access key ID, a secret access key, and a security token.

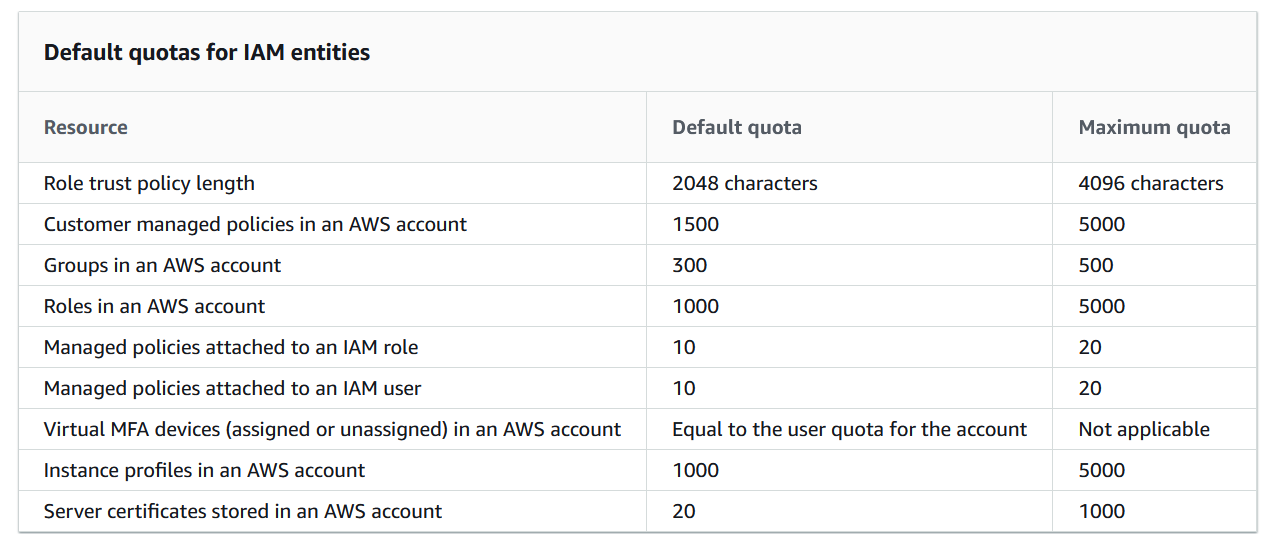
**Access AWS:**

* AWS Management console**:** p/w and MFA
* AWS CLI**:** protected by access keys
* AWS SDK**:** for code. protected by access keys

The **AWS Security Token Service (STS)** is a web service that enables you to request temporary, limited-privilege credentials for AWS Identity and Access Management (IAM) users or for users that you authenticate (federated users).

**Info:**

* IAM Credentials report lists all your AWS Account's IAM Users and the status of their various credentials.
* A statement in an IAM Policy consists of Sid, Effect, Principal, Action, Resource, and Condition.
* Version is part of the IAM Policy itself, not the statement.
* You receive AWS CloudTrail log records that include information about IAM identities who made requests for resources in your account.
* IAM is eventually consistent. IAM achieves high availability by replicating data across multiple servers within Amazon’s data centers around the world.
* By default, a brand new IAM user has NO permissions to do anything.

****

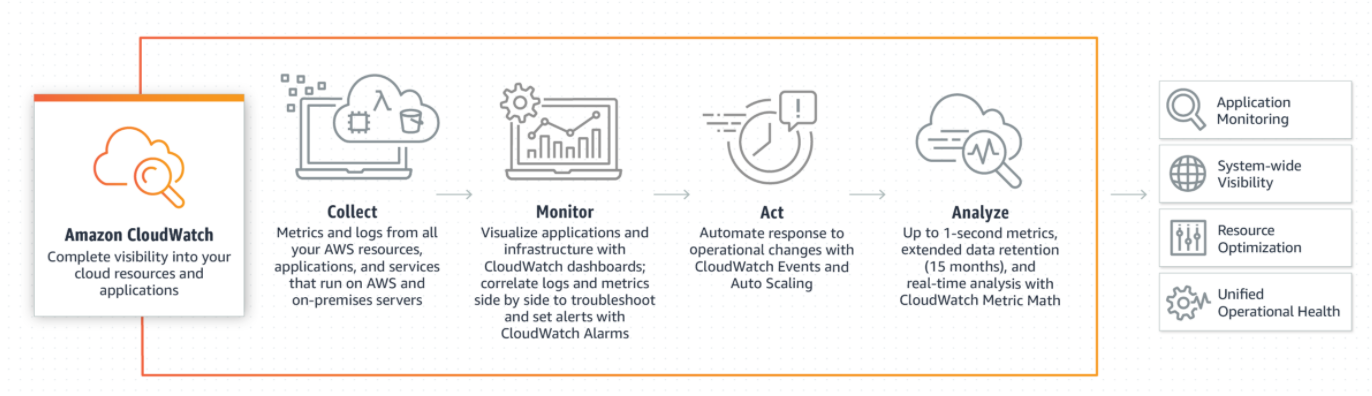
**AWS Resource Access Manager (RAM**):

* Helps you securely share your resources across AWS accounts, within your organization or organizational units (OUs) in AWS Organizations, and with IAM roles and IAM users.
* We can use AWS RAM to share transit gateways, subnets, AWS License Manager license configurations, Amazon Route 53 Resolver rules, etc.
* Many organizations use multiple accounts to create administrative or billing isolation, and to limit the impact of errors.
* With AWS RAM, you don’t need to create duplicate resources in multiple AWS accounts. This reduces the operational overhead of managing resources in every account that you own.
* Instead, in your multi-account environment, you can create a resource once, and use AWS RAM to share that resource across accounts by creating a resource share.

**(3)**

**CloudWatch**

Amazon CloudWatch monitors your Amazon Web Services (AWS) resources and the applications you run on AWS in real time



Default metrics of EC2 instance: Network usage CPU Usage

**Metrics:**

Metrics are data about the performance of your systems

Basic monitoring: which polls for every 5 minutes

Detailed monitoring: which polls for every 1 minute.

**Metric Streams** - a new feature that enables customers to create a continuous, near real-time stream of metrics to a destination of their choice

**Alarm:**

CloudWatch Alarms feature allows you to watch CloudWatch metrics and to receive notifications when the metrics fall outside of the levels (high or low thresholds) that you configure

Ex:

If CPU utilization goes beyond the static threshold alarm goes to alarm state

Three states in CW Alarm:

1. Alarm state
2. Insufficient
3. OK state

**EventBridge:**

* Earlier it was under CloudWatch 🡪 now it is a separate service 🡪 EventBridge
* An Event indicates change in AWS environment
* Event Resource: Which resource you want to monitor
* Event target: to alert the event change through notifications

**Logs:**

* CloudWatch Logs enables you to centralize the logs from all your systems, applications, and AWS services
* (A **log file** is a computer-generated data file that contains information about usage patterns, activities, and operations within an operating system, application, server or another device.)

follow the link and try to reproduce the same:

Ex: <https://www.youtube.com/watch?v=F4IE69V-iuw>

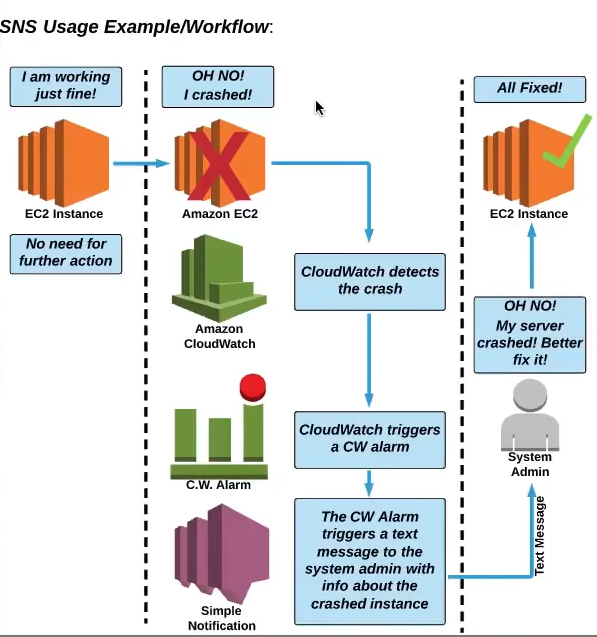
**ARN**: Amazon Resource Name

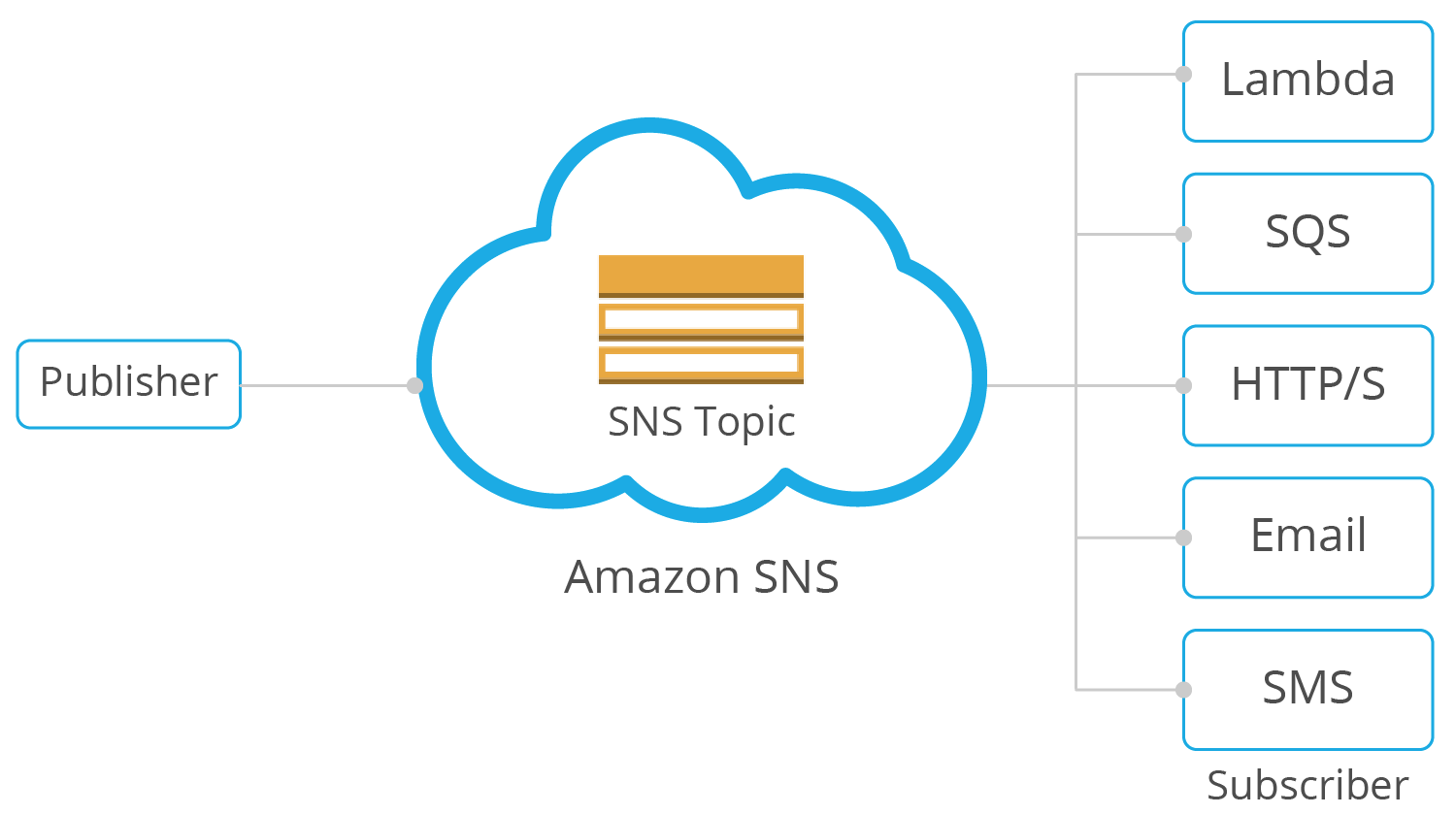
**(4)**

**Simple Notification Service**

* Amazon Simple Notification Service is a notification service provided as part of Amazon Web Service.
* It provides a low-cost infrastructure for the mass delivery of messages, predominantly to mobile users

**Topic:**

An Amazon SNS topic is a logical access point that acts as a communication channel



**(5)**

**CloudTrail**

* Auditing tool records all AWS account activity.
* Any action taken by users, roles and AWS services are recorded to cloud trial.
* Cloud trial events are kept for 90 days in event history
* You can create a trail of your own store the event history in s3 bucket.
* There are two types of events

**Management events:**

Management operations performed on AWS

**Data events:**

Currently supported S3 and Lambda: You can now record all API actions on S3 Objects and receive detailed information such as the AWS account of the caller, IAM user role of the caller, time of the API call, IP address of the API, and other details

**Insights events:**

AWS CloudTrail Insights helps AWS users identify and respond to unusual activity associated with write API calls by continuously analyzing CloudTrail management events.

**Use Case:**

1) incident response/digital forensics:- Cloudtrail will be leveraged to find out activities that were part of security incidents . In reality there will be many background activities prior to the security attack and data theft, this can be creating privileged user accounts, logging attempts on root account, VM spin ups to connect to command and control center etc...Cloudtrail will store all these events and helps us to understand the activities that were happened prior to the security event.

2) Central Event Storage:- Cloud trail stores all management and data events centrally (as a best practice, management and data events can be separate trails) which can create report and baseline the set of activities usually happen in your AWS account.

**(6)**

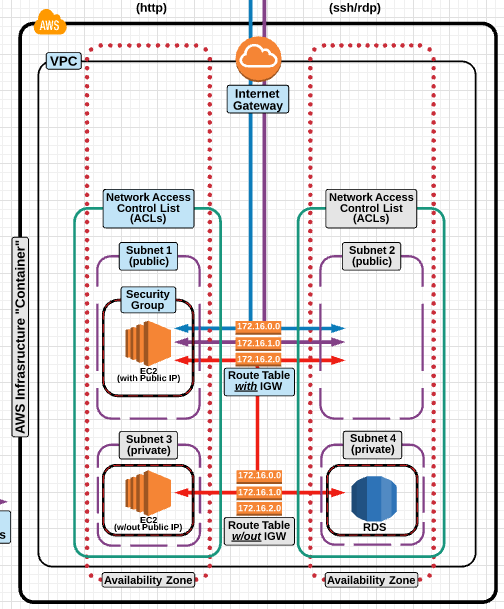
**Virtual Private Cloud**

Amazon Virtual Private Cloud (Amazon VPC) lets you provision a logically isolated section of the AWS Cloud where you can launch AWS resources in a virtual network that you define like EC2 instance Databases.

* By default, a VPC will be created in 2 AZs (1 will be invisible)

**Info:**

* You can have multiple VPC in AWS region (max 5---soft limit)
* Max CIDR/VPC is 5, for each CIDR:
  + Min size is /28 (16 IP addresses)
  + Max size is /16 (65536 IP addresses)
* VPC is Pvt. so only Pvt. ipv4 ranges are allowed



**CIDR:(Classes interdomain routing)**

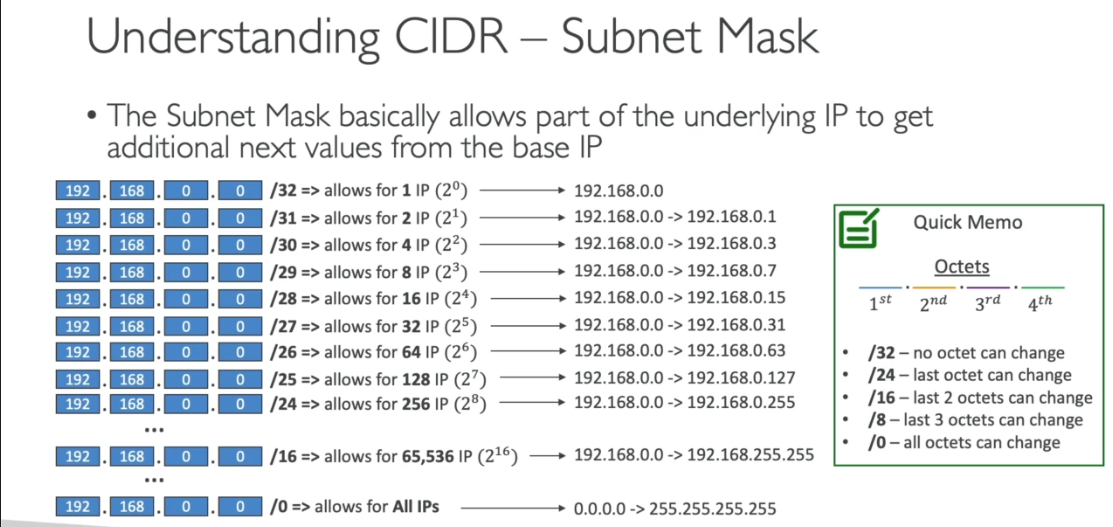
Classless Inter-Domain Routing is a method for allocating [IP addresses](https://en.wikipedia.org/wiki/IP_address) and for [IP routing](https://en.wikipedia.org/wiki/IP_routing).

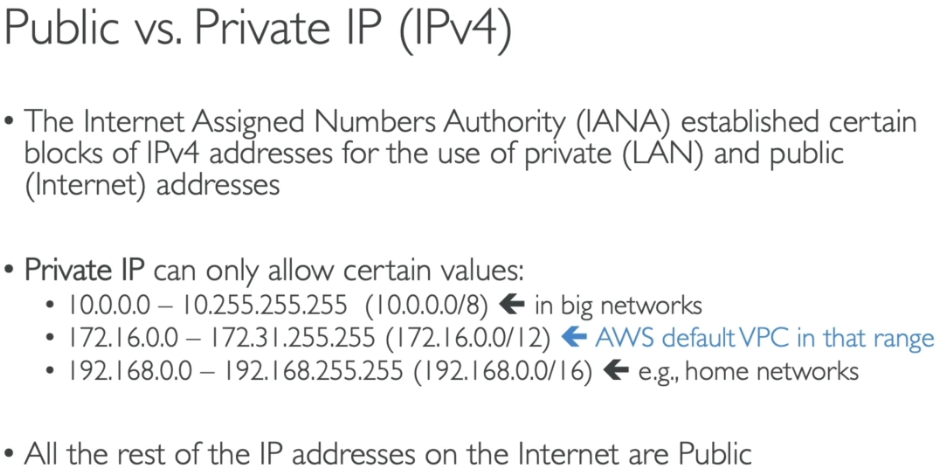
Ex: The IPv4 block 192.168.100.0/22 represents the 1024 IPv4 addresses from 192.168.100.0 to 192.168.103.255.

I.e., 2^ (32-22) = 2^10 = 1024 IPv4 addresses.

CIDR has 2 components:

* Base IP:
  + IP in range XX.XX.XX.XX (ex: 10.0.0.0,192.168.0.0)
* Subnet Mask:
  + Defines how many bits can change in the IP
  + Ex: /0,/24,/32
  + Can take 2 forms
    - /8----255.0.0.0
    - /16-----255.255.0.0
    - /24------255.255.255.0
    - /32------255.255.255.255





**VPC design:**

|  |  |  |  |
| --- | --- | --- | --- |
| VPC CIDR = 10.180.0.0/16 means we have 65536 IPv4 address | | | |
| IPV4 Address range is 10.180.0.0 ---- 10.180.255.255 | | | |
| Public subnet 1 | Public Subnet 2 | Private Subnet 1 | Private Subnet 2 |
| 10.180.0.0/24 | 10.180.1.0/24 | 10.180.2.0/24 | 10.180.3.0/24 |
| 256 IPV4 address | 256 IPV4 address | 256 IPV4 address | 256 IPV4 address |
| 10.180.0.0-10.180.0.255 | 10.180.1.0-10.180.1.255 | 10.180.2.0– 10.180.2.155 | * + 1. – 10.180.3.255 |

* A **subnet** is a range of IP addresses of resources in your VPC. You can launch AWS resources into a specified subnet.
* Some IP addresses are reserved they are (first 4 and last 1 IP are reserved)
* 10.180.0.0 Network address
* 10.180.0.1 VPC Router
* 10.180.0.2 DNS server (**DNS**. (Domain Name System) The Internet's system for converting alphabetic names into numeric IP addresses)
* 10.180.0.3 Future use
* 10.180.0.255 N/W Broadcast address
* VPC spans multiple Availability zones.
* Subnets must be associated with route table
* A **public subnet** has a route to internet (IGW)
* A **private subnet** doesn’t have route to internet. It creates higher level of security.
* **Internet Gate Way (IGW):**
* Allows resources in a VPC connect to internet
* Scales horizontally, highly available and redundant
* Must be created separately from VPC
* One VPC can only be attached to one IGW and vice versa
* You can use a **Network Address Translation (NAT) 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.
* **Route Table** contains entries that will mention that which resources are associated with which subnets and which gateways are associated with those subnets

**VPC Quota or VPC limitations**

* IPv4 CIDR blocks per VPC 4
* 5 VPC per region
* 5 IGW per region
* Elastic IP addresses per Region 5
* NAT gateways per Availability Zone 5
* Subnet per VPC 200
* Network ACLs per VPC 200
* Rules per network ACL 200

Use Case:

1. Creating Branch Office and Business Unit Networks

Quite often, there is a requirement for connecting branch offices with their own, interconnected networks. This requirement can be fulfilled by provisioning instances within a VPC with a separate subnet for different branch offices. All resources within a VPC can communicate with each other through a private IP address by default, so all offices will be connected to each other and will also have their own local network within their own subnet.

If you need to limit communication across subnets for some instances, you can use security groups to configure access for such instance.

2. Hosting a Public Facing Website

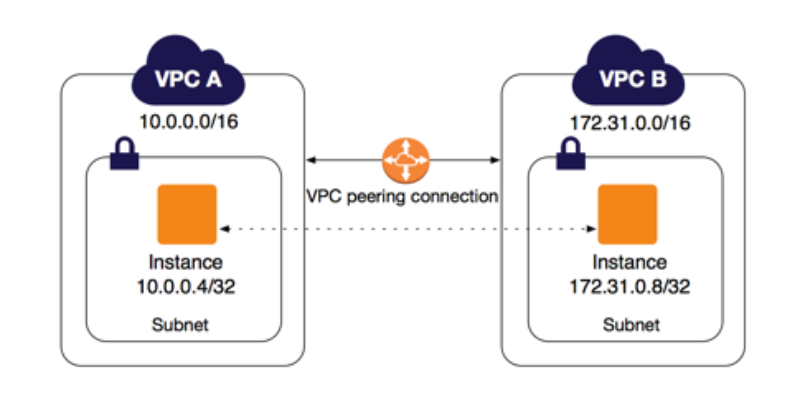
You can host a public facing website, which could be a blog, a single tier simple web application, or just a simple website using VPC. You can create a public subnet using the VPC wizard and select the VPC with a single public subnet only option, or you can create it manually. Secure your website using instance-level firewalls, known as security groups, allowing inbound traffic, either HTTP or HTTPS, from the internet and restricting outbound traffic to the internet when required at the same time

3. Disaster Recovery

As part of your **disaster recovery** (**DR**) and business continuity plan, you will need to continuously back up your critical data to your DR site. You can use a VPC to host EC2 instances with EBS volumes and store data in S3 buckets as well as in EBS volumes attached to EC2 instances securely, which can be configured to be accessible only from your network.

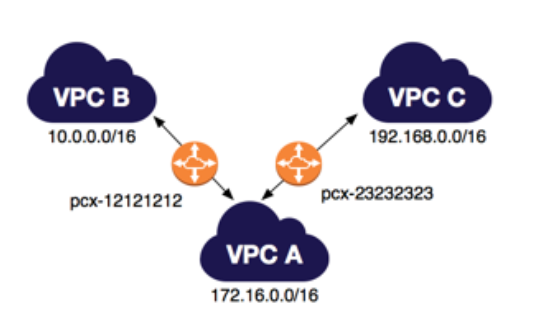
**VPC Peering**:

* A VPC peering connection is a networking connection between two VPCs that enables you 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.
* You can create a VPC peering connection between your own VPCs, or with a VPC in another AWS account.
* The VPCs can be in different regions (also known as an inter-region VPC peering connection).
* VPC peering uses IGWs



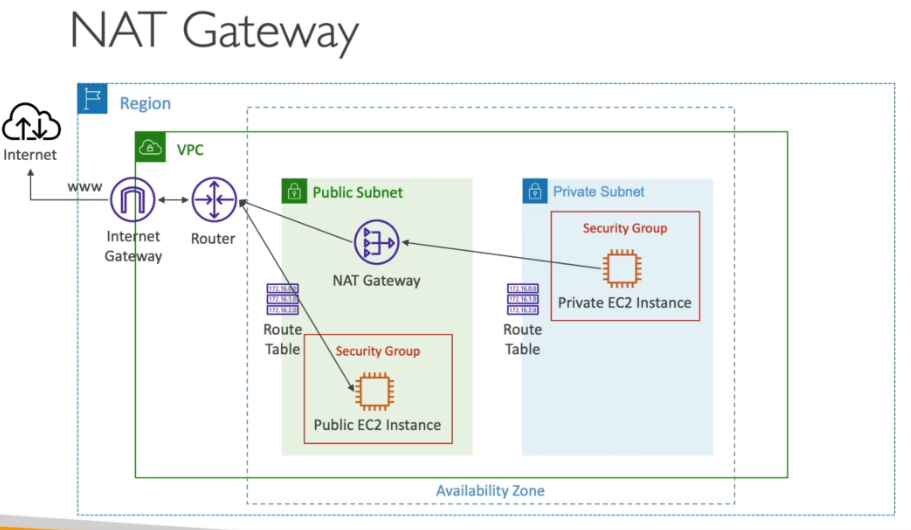
**Conditions:**

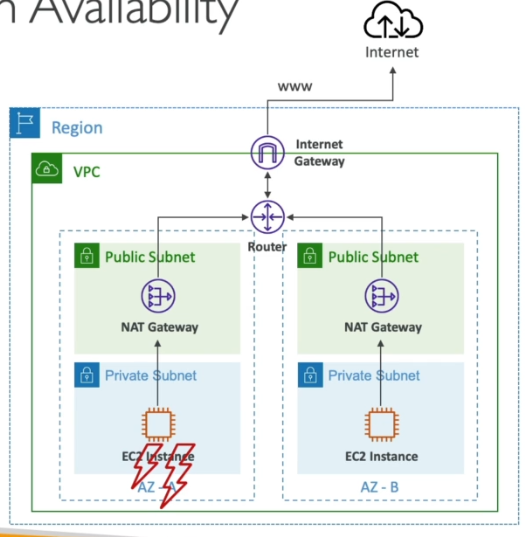
* CIDR block shouldn’t overlap
* **Transitive peering** relationships are not supported. i.e. here VPC B cannot connect with VPC C.
* If the VPCs are in different regions, inter-region data transfer costs apply.
* You cannot have more than one VPC peering connection between the same two VPCs at the same time.



**Nat gateway**: (One Way)

* A NAT gateway is a Network Address Translation (NAT) service. You can use a NAT gateway so that instances in a private subnet can connect to services outside your VPC, but external services cannot initiate a connection with those instances.
* Aws managed. Higher bandwidth. High availability. Must create multiple NatGW in, multiple AZs for fault tolerance
* Created in specific AZ and uses an **elastic IP** (public IP address which is doesn’t change frequently until you choose to release it)
* Can’t be used by ec2 in same subnet (only from other subnets)
* Requires IGW. Pvt subnet 🡪 NatGW 🡪 IGW
* No security groups required to mange





**Transit Gateway:**

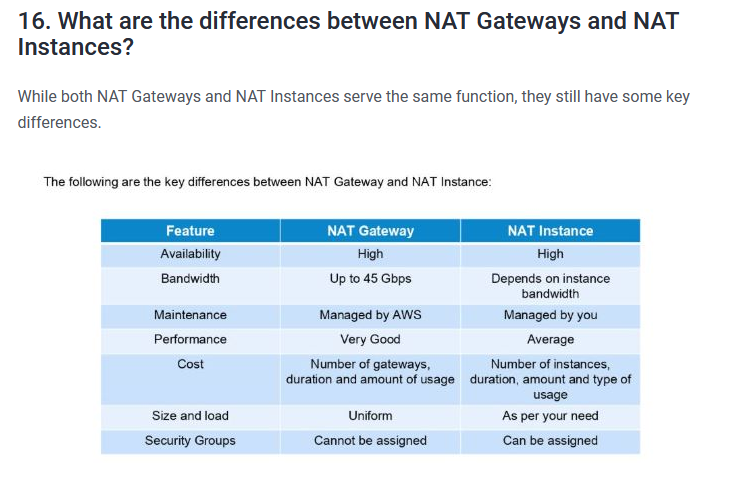
* Used to connect VPC to On-premises server.

**VPC endpoint**

VPC endpoints enables you to privately access specific AWS services from your own Amazon Virtual Private Cloud (VPC), without using public IP addresses and without requiring the traffic data to travel across the Internet.

Note: VPC endpoints are only supported within the same AWS region. You cannot use endpoints to connect an AWS service from one region to a VPC in a different region.

**NAT Instance:**

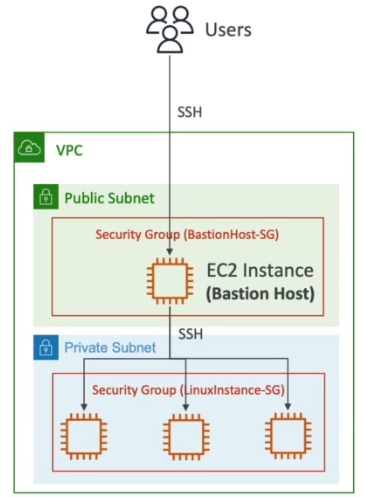


**Bastion Host**: (Two Way)

* A **bastion host** is a **server** whose purpose is to provide access to a private network from an external network.
* It is an ec2 instance launched in public subnet
* Bastion host only has port 22 traffic from the IP address you need, not from the security groups of your other ec2

AWS private Link

Enable



**Gateway endpoints**

A gateway endpoint is a gateway that is a target for a route in your route table used for traffic destined to either Amazon S3 or DynamoDB.

**VPN:**

Is mainly used to establish a secure and private tunnel from you network or device to AWS network

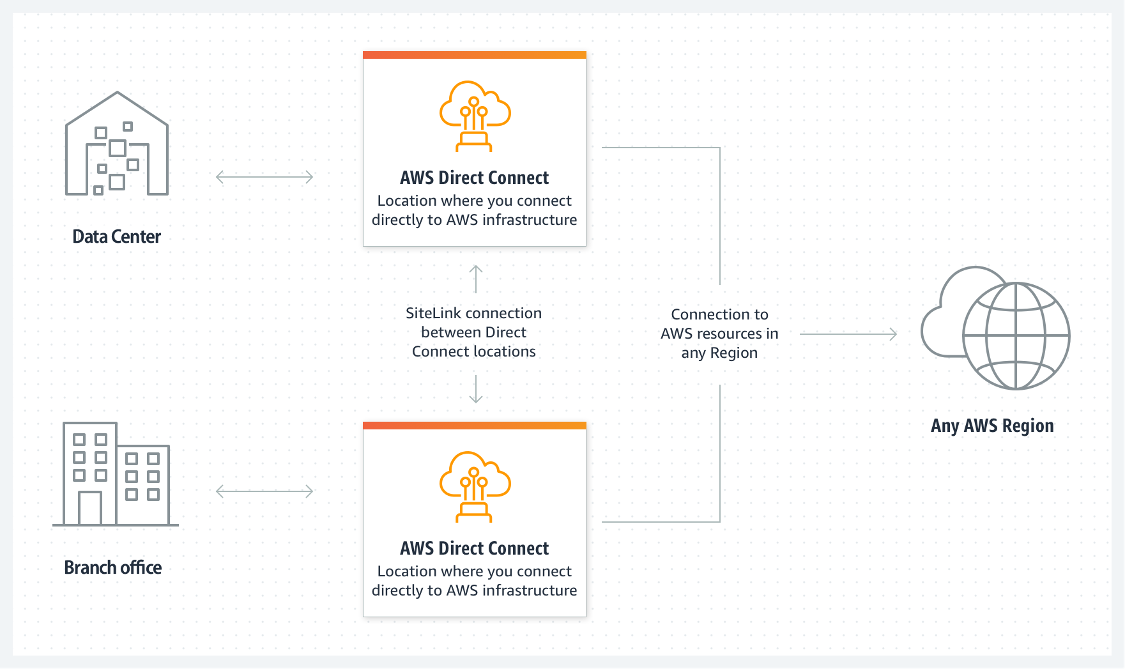
* AWS site-to-site VPN: enables you to securely connect your on-premises network to your VPC.
* AWS client VPN: enables you to securely connect users to AWS or on premises network.

Do we have another way we can connect to the resources in a private subnet?

We can setup a VPN server in the public subnet and configure it to connect to resources residing the private subnet

**AWS Direct Connect:**

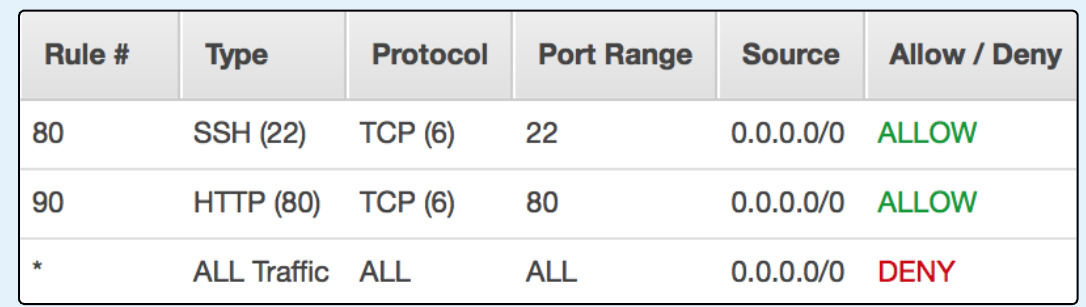
* The AWS Direct Connect cloud service is the shortest path to your AWS resources. While in transit, your network traffic remains on the AWS global network and never touches the public internet.
* With AWS Direct Connect SiteLink, you can send data between AWS Direct Connect locations to create private network connections between the offices and data centers in your global network.
* Secure your data as it moves between your network and AWS with multiple encryption options.
* Reduce your networking costs with low data transfer rates out of AWS.



**NACL:**

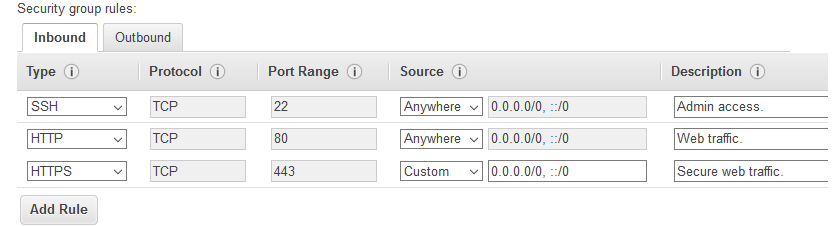
* A **Network Access Control List** (ACL) is an optional layer of security for your VPC that acts as a firewall for controlling traffic in and out of one or more subnets. (Firewall at subnet level)
* This is Optional
* Only 1 NACL is allowed for a subnet
* Stateless

1. Inbound means – incoming (Ingress)
2. Outbound means – outgoing (egress)
3. Always explicit deny take precedence over allow



**Security Group:**

* A **security group** acts as a virtual firewall for your instance to control inbound and outbound traffic.
* A firewall at Instance level
* This is mandatory
* Stateful – response is always allowed (suppose all inbound traffic is denied 🡪 yet the response which is inbound for an outbound traffic is allowed)
* Multiple SGs can be associated with EC2 instance

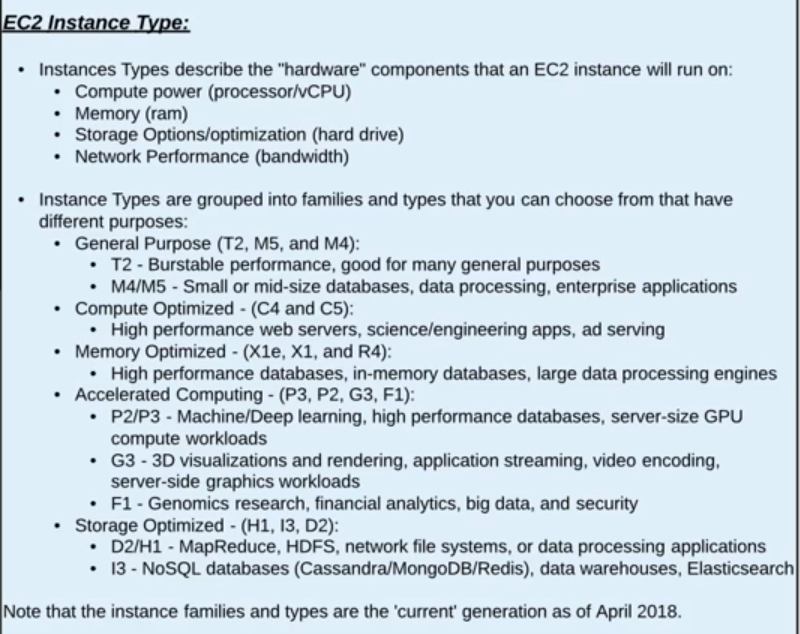


**(6)**

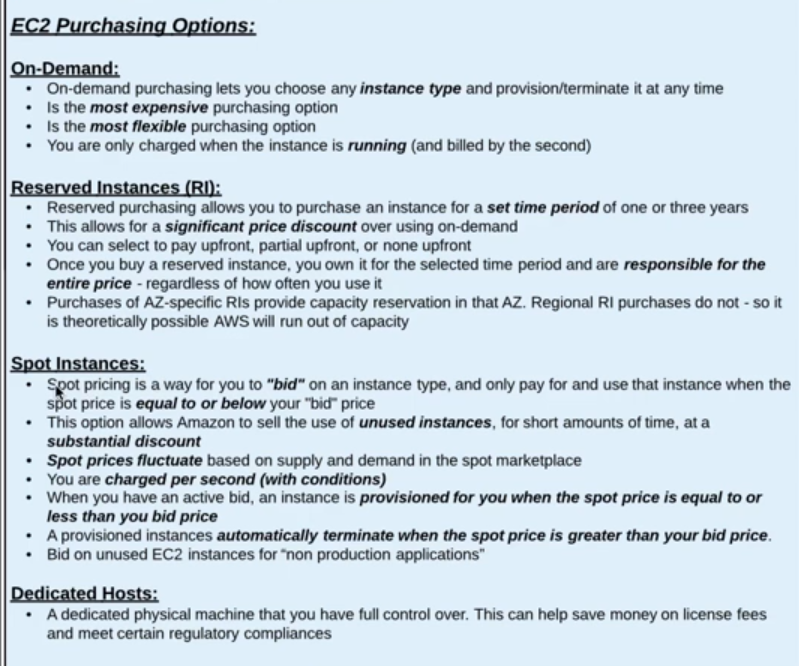
**Elastic Cloud Compute**

An **EC2 instance** is a virtual server in Amazon's Elastic Compute Cloud (**EC2**)

**EC2 instance types:**

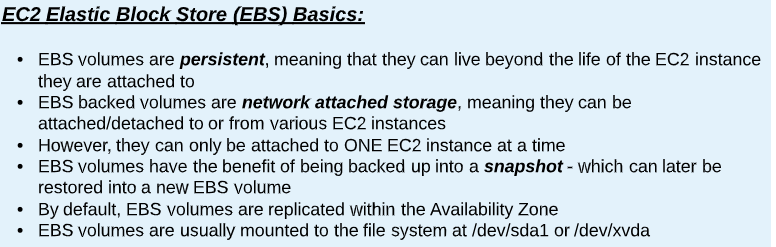
****

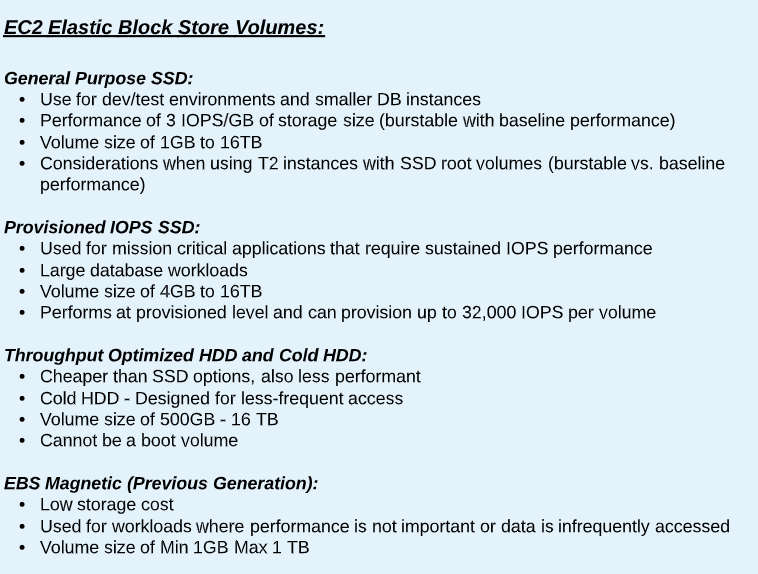
**EC2 purchasing options:**

****

**EBS (Elastic block storage)**

easy-to-use, scalable, high-performance block-storage service. provides raw block-level storage that can be attached to Amazon EC2 instances

****

**EBS Types:**

**What is burstable instance?**A burstable instance is a virtual machine (VM) instance that provides a baseline level of CPU performance with the ability to burst to a higher level to support occasional spikes in usage.

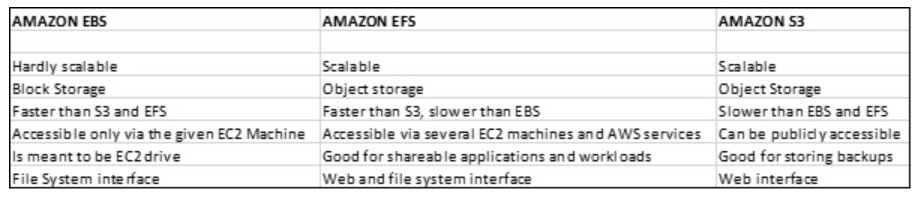
**Amazon Elastic file system (EFS):**

* Amazon Elastic file systemis a regional service storing data within and across multiple Availability Zones (AZs) for high availability and durability
* Amazon EFS is **not supported on Windows instances.**
* Whereas EBS is used only in its own availability zone.

**Q: Can we take snapshot of EFS?**

There are two options available for protecting your data by backing up your EFS file systems: AWS Backup service. The EFS-to-EFS backup solution.

**Difference between EBS v/s EFS v/s S3**



**Snapshot EBS**

* You can back up the data on your Amazon EBS volumes to Amazon S3 by taking point-in-time snapshots. Snapshots are incremental backups

**Snapshots are stored in S3**

* Launch two ec2 instance in different AZ’s (instance1 & instance2)
* Create EBS volume and attach it to instance1
* The volumes are attached to instance1
  + you can verify it by logging into instance1 and executing “lsblk” command,
  + but it’s not mounted you can verify it through by running command “df -TH”
* Mount the volume to instance1
* **Format** the disk with ext4: “**mkfs -t** ext4 /dev/xvdf”
* **Create** a directory in root:

1. “**cd** /”

2. “**mkdir** /mnt/mydisk”

* Mount the disk: “**mount** /dev/xvdf /mnt/mydisk”
* you can verify that disk is mounted by running “df -TH” command.
* Create some files
* Take a snapshot
* Unmount the disk
* **umount** /mnt/mydisk
* Detach the volume from ec2 instance.
* Delete the volume
* Create a new volume from snapshot
* Attach the volume to newly created instance2.
* Mount the volume to instance2
* Create a directory in root:

1. **cd** /

2. **mkdir** /mnt/mydisk

* **mount** /dev/xvdf /mnt/mydisk

**Assignment:** Difference between instance level snapshot and volume snapshot

Each region has different AMI

**Data life cycle Manager:**

You can use Amazon Data Lifecycle Manager to automate the creation, retention, and deletion of snapshots taken to back up your Amazon EBS volumes

**Amazon machine image (AMI):**

* An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance.
* You can launch multiple instances from a single AMI when you need multiple instances with the same configuration.

**Difference between Snapshot and AMI**

An EBS snapshot is a backup of a single EBS volume. The EBS snapshot contains all the data stored on the EBS volume at the time the EBS snapshot was created.

An AMI image is a backup of an entire EC2 instance. Associated with an AMI image is EBS snapshots. Those EBS snapshots are the backups of the individual EBS volumes attached to the EC2 instance at the time the AMI image was created.

|  |  |
| --- | --- |
| AMI | Snapshot |
| Associated with EC2 instance | Associated with EBS and S3 |
| It is a launch template  Or is a bootable copy of EC2 instance | It is a backup of data on EBS volumes |

**Elastic Load Balancer (ELB):**

Scalability and high availability:

Scalability are 2 types

* Vertical scale up/down

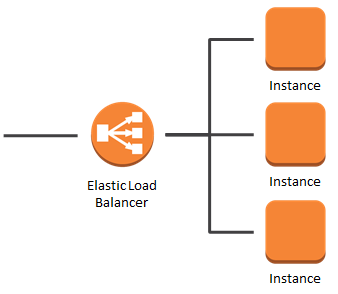
Ex. Databases (non distributed systems)

* Horizontal (elasticity) scale out/in

Ex: distributed sys (no. of instances)

Scalability is different from high availability

* Different AZ’s
* Elastic Load Balancing automatically distributes your incoming traffic across multiple targets, such as EC2 instances, containers, and IP addresses, in one or more Availability Zones
* Increases the availability of your application.
* Expose single point of access (DNS) to your application
* Enhanced security

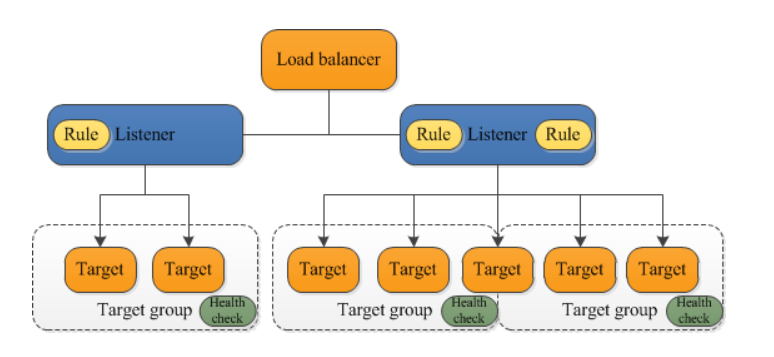


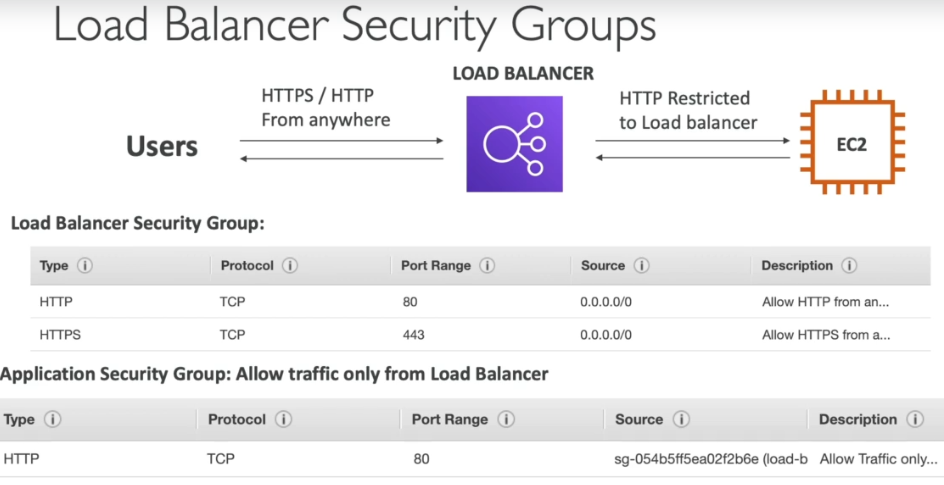
**Listener:**

* A *listener* checks for connection requests from clients, using the protocol and port that you configure.
* The rules that you define for a listener determine how the load balancer routes requests to its registered targets.

**Health checks:**

* Is done on port and route
* Response not 200 (OK), then unhealthy instance

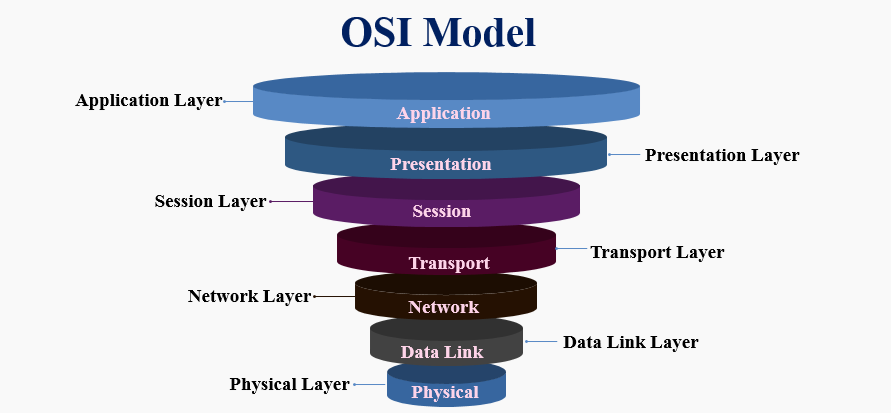
****

****

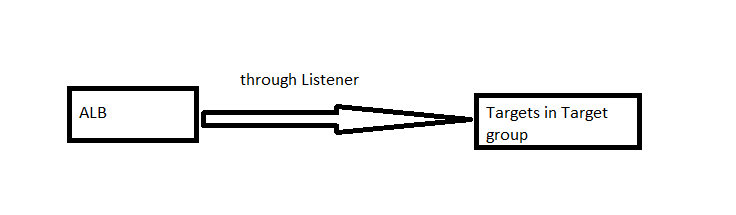
**Types of Load balancer:**

**1. Application load balancer (ALB)**:

* Used mainly for web application running http and https protocols.
* Operates at request level. functions at the application layer, the seventh layer of the Open Systems Interconnection (OSI) model.



* The default routing algorithm is round robin (123 🡪 123 🡪 123 ); alternatively, you can specify the least outstanding requests routing algorithm
* The Application Load Balancer operates at the request level only. If you're dealing with HTTP requests, which you are for your web application, we can use this.
* It also supports advanced features like host and path-based routing and query string, headers (ex: http://www.pavanisgod.com/vineel-asshole/chinna-asswhole ------“ vineel-asshole/chinna-asswhole” is path)



* listener means a process that checks connections using http or https – ports

1st traffic enters target group then goes to security group of an instance.

* Supports http https (layer 7) websocket
* Ec2, ecs, lambda functions, ip addresses (pvt)---->target groups

**2. Network Load balancer (NLB):**

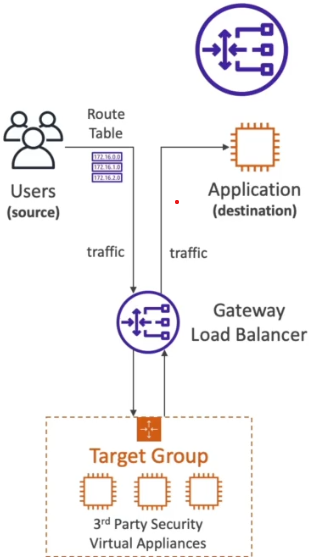
* Ultra-high Performance at very low latency.
* Operates at connection level, routing traffic to targets with-in VPC.
* Can handle millions of requests per second.
* Supports tcp (layer 4),tls(secure tcp),udp
* NLB has one static IP/AZ and supports assigning Elastic IP
* Ec2, ip addresses (pvt), ALB---->target groups

**3. Classic load Balancer:**

* Used for applications that were built in existing EC2 classic env.
* Operates both at connection & request level.
* **The Classic Load Balancer is a connection-based balancer** where requests are forwarded by the load balancer without “looking into” any of these requests. They just get forwarded to the backend section
* Supports http https (layer 7) tcp (layer 4),

**4. Gateway Load Balancer:**

* **Still under research**
* **Improve security, compliance and policy controls**
* **Operates at layer 3 (network layer) IP protocal**
* Ec2, ip addresses (pvt), ALB---->target groups

****

**Note:** We can create **target groups** in order to route to traffic to the respective paths

Target groups can be given to 🡪

* + - 1. Instances
      2. IP addresses
      3. Lambda Functions
      4. Application Load Balancer

**Example: Classic load balancer**

* Spin up an EC2 instance1 in another availability zone (az1) with http port open in Security group
* Add the below script and launch the instance.

#!/bin/bash

yum update -y

yum install httpd -y

echo '<h1> Response from server-1 </h1>' > /var/www/html/index.html

service httpd start

chkconfig httpd on

* Spin up one more EC2 instance1 in another availability zone (az2) with http port open in Security group
* Add the below script and launch the instance.

#!/bin/bash

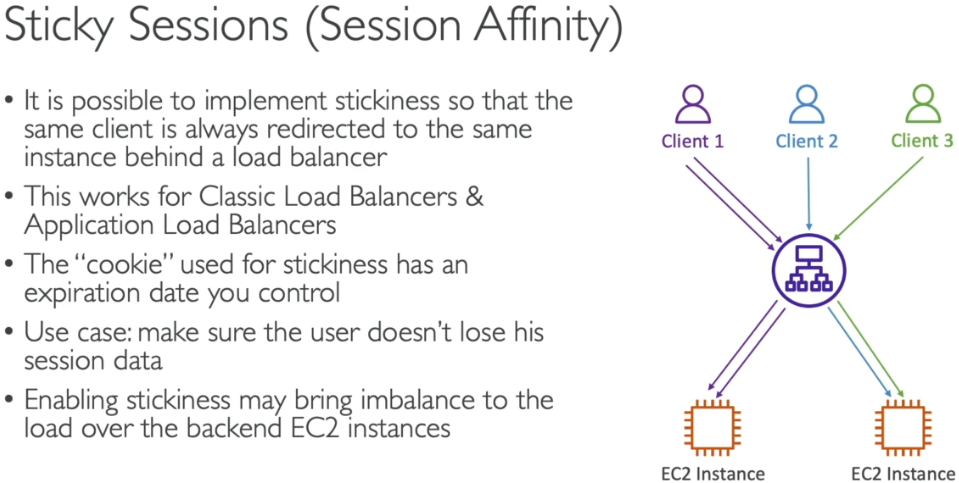
yum update -y

yum install httpd -y

echo '<h1> Response from server-2 </h1>' > /var/www/html/index.html

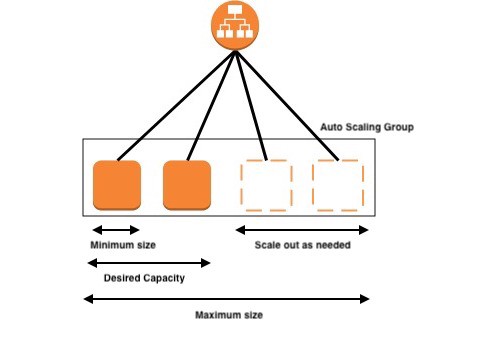
service httpd start

chkconfig httpd on



**Autoscaling**

AWS Auto Scaling monitors your applications and automatically adjusts capacity to maintain steady, predictable performance at the lowest possible cost.



**Creating an AMI image or template of an instance:**

Ec2 🡪 select instance 🡪 actions 🡪 image and template 🡪 create image or template

AMI Image or template is used for autoscaling

Template need not have load balancer for autoscaling.

Load can be – requests/CPU usage/ storage or any metric using cloud watch etc.

**Autoscaling steps involved:**

Template configuration 🡪 instance launch options 🡪 other advanced options 🡪 sizing and scaling policies 🡪 SNS notifications 🡪 review

Auto scaling can be updated through edit option in autoscaling tab.

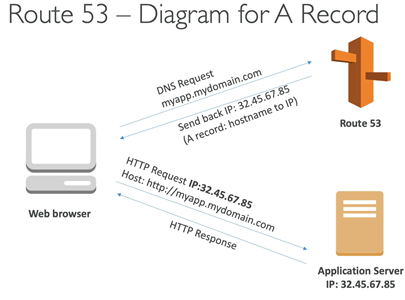
Encryption 🡪 converting data into binary

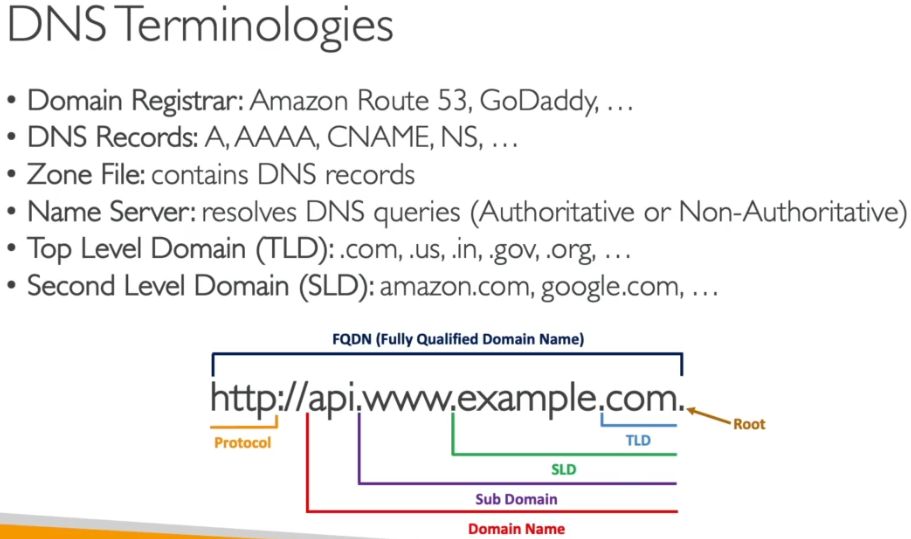
* Server-side encryption
* Client-side encryption

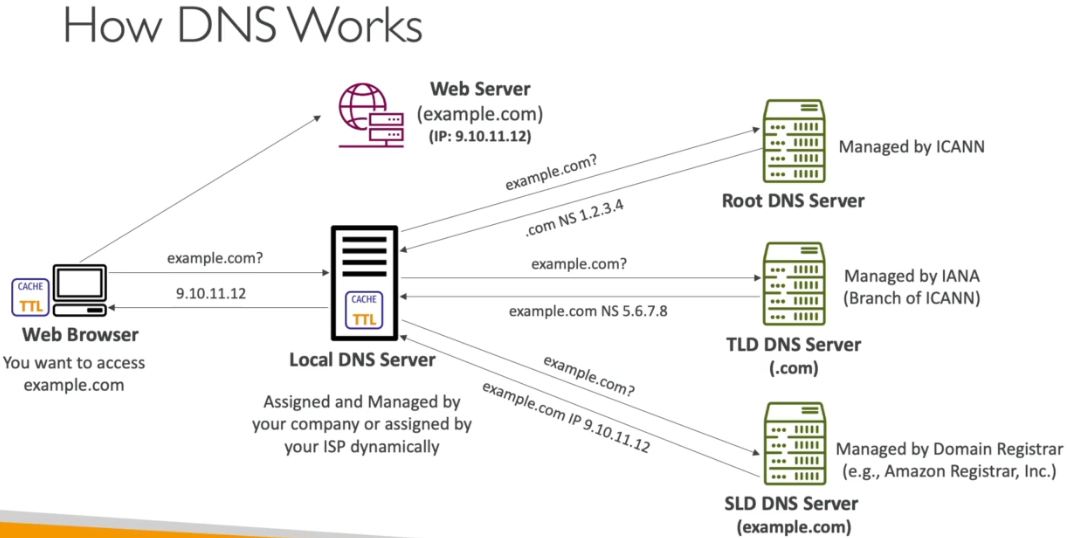
**(7)**

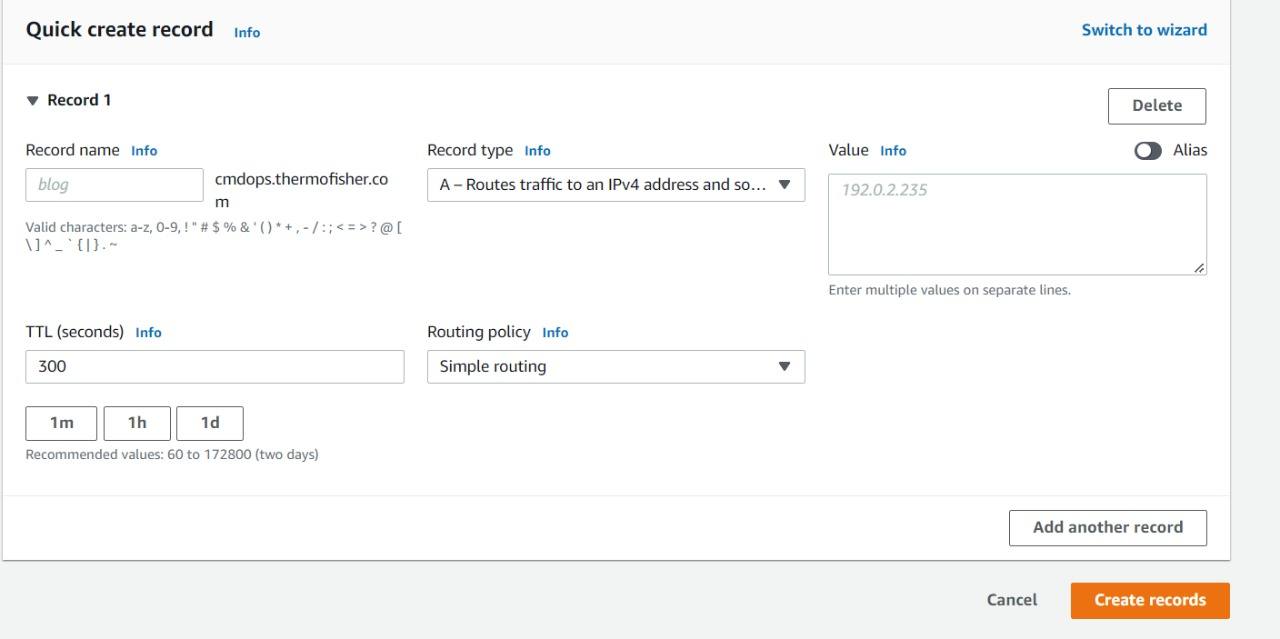
**Route53:**

Amazon Route 53 is a highly available and scalable Domain Name System (DNS) web service, where we can point IP address to domain name or point host name to another host name.

****

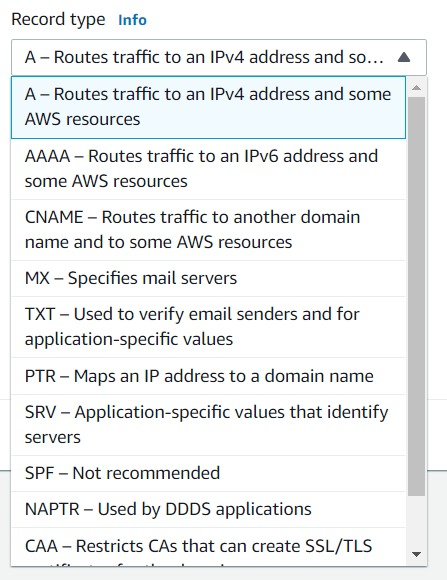
****

****



**Record Types:**

**A Record:** Maps IP address to domain name ex: 10.180.0.0 to myapp.mydomain.com

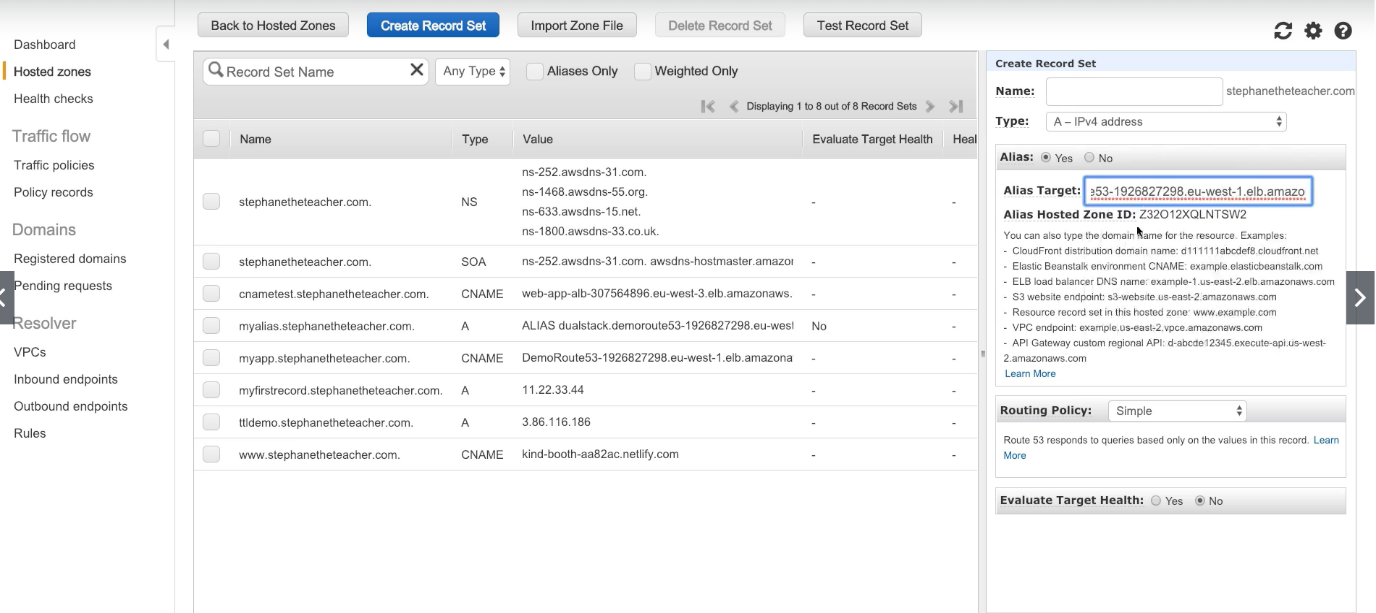


**CNAME Record:** Canonical Name

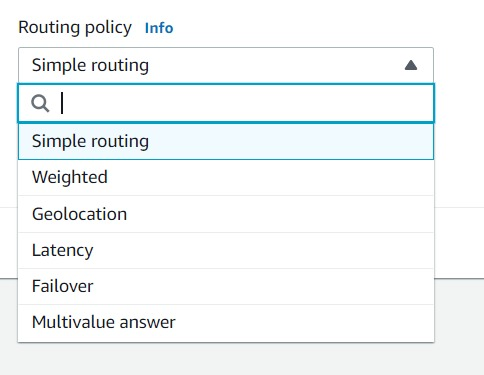
Maps hostname to another host name: us-east.2.elb.amazonaws.com to myapp.mydomain.com

**Alias Record**:

Points a host name to AWS Resource ex: myapp.mydomain.com to us-east.2.elb.amazonaws.com

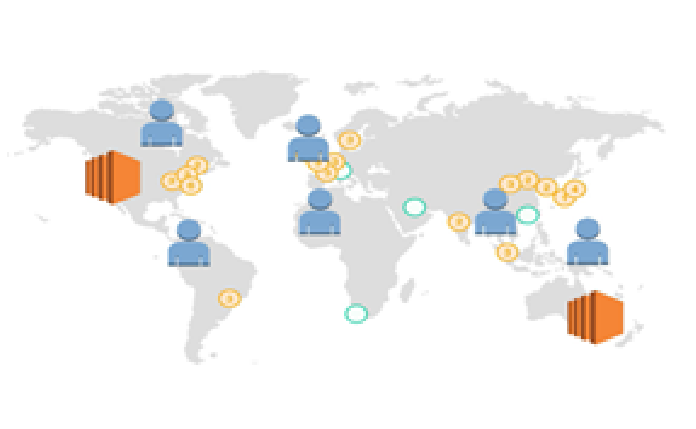


**Routing Policies:**



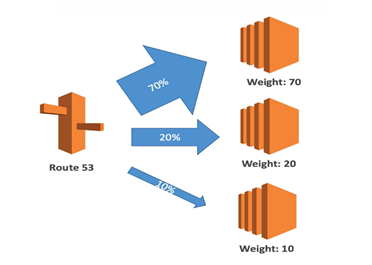
**1. Latency Routing Policy:**

Use when you have resources in multiple AWS Regions, and you want to route traffic to the region that provides the best latency.



**2. Weighted Routing Policy:**

* Use to route traffic to multiple resources in proportions that you specify.
* It can be used when there is no Load balancer



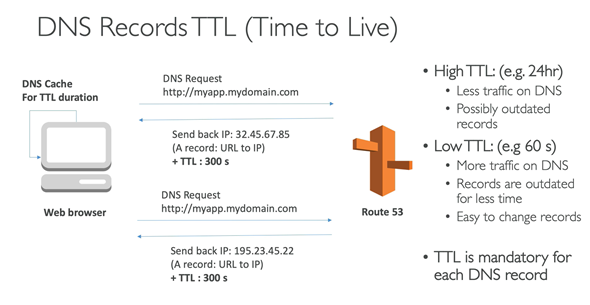
**3. Failover Routing Policy:**

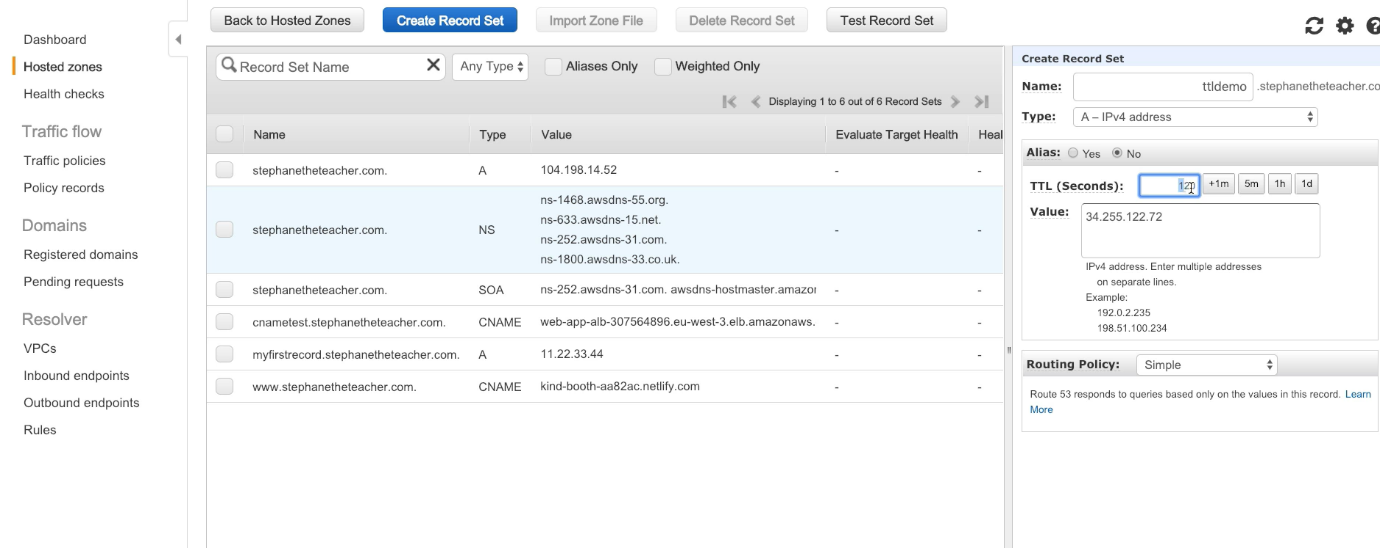
Use to route traffic to healthy resources.

**4. Geobased Routing Policy:**

Let’s you to route traffic to different AWS locations. We can select country in the dropdown.

**TTL:**

****



**(8)**

**Simple storage service (S3)**

Amazon S3 has a simple web services interface that you can use to store and retrieve any amount of data, at any time, from anywhere on the web.

**Single operation upload:**

* It’s a traditional upload where you will upload the object in one part
* A single operation upload can upload the file up to 5GB in size.

**Upload object in parts:**

* Using multipart upload, you can upload the large objects up to 5TB.
* You can use multipart upload for the objects from 5MB to 5TB in size.

**Rules for bucket naming:**

* Bucket names must be between 3 and 63 characters long.
* Bucket names can consist only of lowercase letters, numbers, dots . and hyphens -.
* Bucket names must begin and end with a letter or number.
* Bucket names must not be formatted as an IP address (for example, 192.168.5.4).
* Bucket names can't begin with xn-- (for buckets created after February 2020).
* Bucket name must be a unique DNS-compliant name

**Limitation of S3 bucket:**

* Only 100 buckets can be created per account, can hold unlimited objects
* Can hold unlimited objects

**Data Consistency Model:**

* read-after-write consistency for PUTS of new objects inyour S3 bucket in all regions
* eventual consistency for read-after-write HEAD or GETrequests
* eventual consistency for overwrite PUTS and DELETESin all regions
* strong read-after-write consistency for any storagerequest

**S3 Storage classes:**

1. Standard:

* Designed for general- and all-purpose storage
* Default storage option
* 99.999999999% object durability
* 99.99% object availability
* Most expensive storage class.

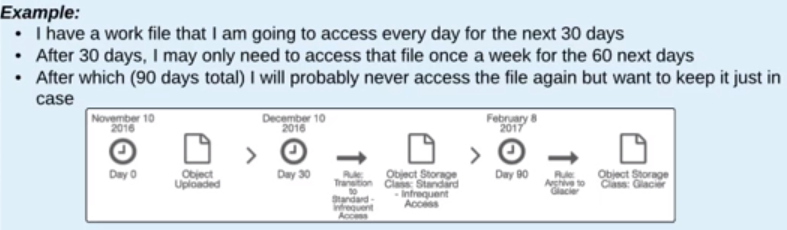
2. Reduced Redundancy storage

* Designed for non-critical objects
* 99.99% object durability
* 99.99% object availability
* Less expensive than standard

3. Infrequent access

* Designed for less frequently accessed objects.
* 99.999999999% object durability
* 99.99% object availability
* Less expensive than reduced redundancy storage
* Glacier
* Designed for long term archival storage
* May take several hours to retrieve the objects from this storage
* Cheapest s3 storage class

Refer the link: [S3 Storage Classes: S3 Standard, S3 IA, S3 RRS (msp360.com)](https://www.msp360.com/resources/blog/amazon-s3-storage-classes-guide/#:~:text=Amazon%20S3%20Storage%20Classes%20Comparison%20Table%20%20,for%20%20...%20%202%20more%20rows%20)



**Versioning:**

* Use versioning to keep multiple versions of an object in one bucket.
* Since versioning is disabled by default, need to EXPLICITLY enable.
* When you DELETE an object, all versions remain in the bucket and Amazon S3 inserts a delete marker.

**S3 Life cycle policy:**

An object lifecycle policy is a set of rules that automate the migration of the object storage class to different storage class. By default, lifecycle policies are disabled for a bucket

**1. IAM Policy (User-based Policy)**

This IAM policy grants the IAM entity (user, group, or role) it is attached to permission to perform any S3 operation on the bucket named “my\_bucket”, as well as that bucket’s contents.

{

"Version": "2012-10-17",

"Statement":[{

"Effect": "Allow",

"Action": "s3:\*",

"Resource": ["arn:aws:s3:::my\_bucket",

"arn:aws:s3:::my\_bucket/\*"]

}

]

}

**2. Bucket Policy**

This S3 bucket policy enables the root account 111122223333 and the IAM user Alice under that account to perform any S3 operation on the bucket named “my\_bucket”, as well as that bucket’s contents.

{

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Principal": {

"AWS": ["arn:aws:iam::111122223333:user/Alice",

"arn:aws:iam::111122223333:root"]

},

"Action": "s3:\*",

"Resource": ["arn:aws:s3:::my\_bucket",

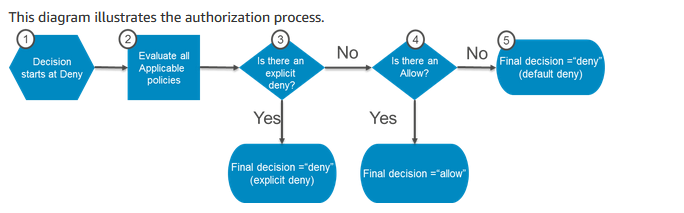
"arn:aws:s3:::my\_bucket/\*"]

}

]

}

**3.Access Control Lists (legacy)**



## **Amazon S3 Use Cases**

### Storage for Internet

### Backup and Disaster Recovery

### Analytics

### Data Archiving

### Static Website Hosting

### Security and Compliance

**S3 Encryption:**

**Two ways of protecting information with S3**

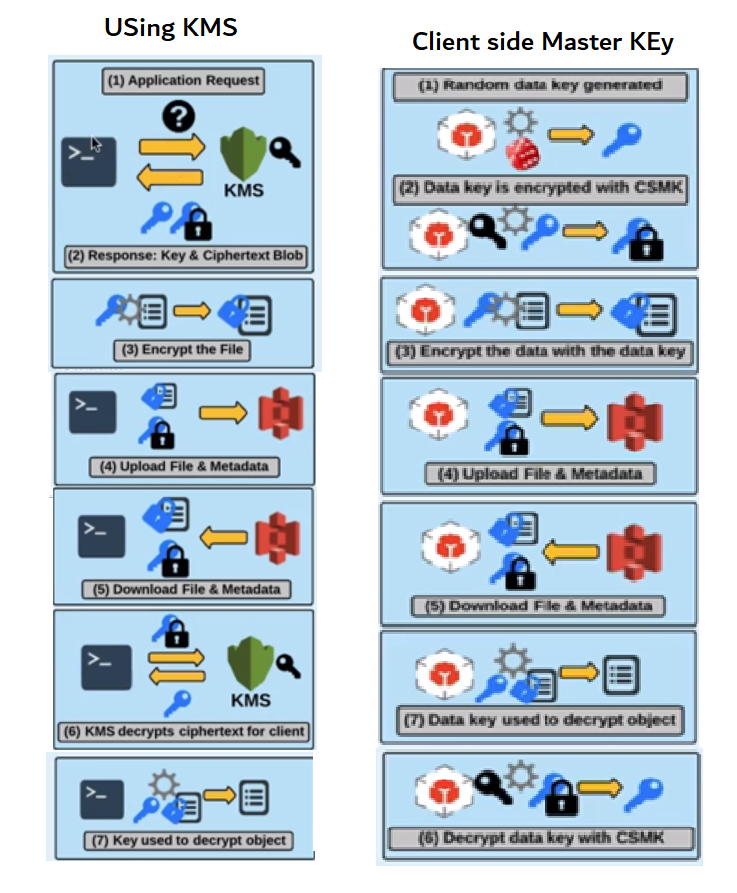
1. **Server side/At rest:**
   1. Amazon S3-Managed Keys (SSE-S3)
   2. AWS KMS-Managed Keys (SSE-KMS)
   3. Customer-Provided Keys (SSE-C)

****

1. **In-transit/Client-side encryption:**
   1. AWS KMS-managed customer master key
   2. client-side master key

****

In client side/ in-transit we have two types:

****

**Key Management services:**

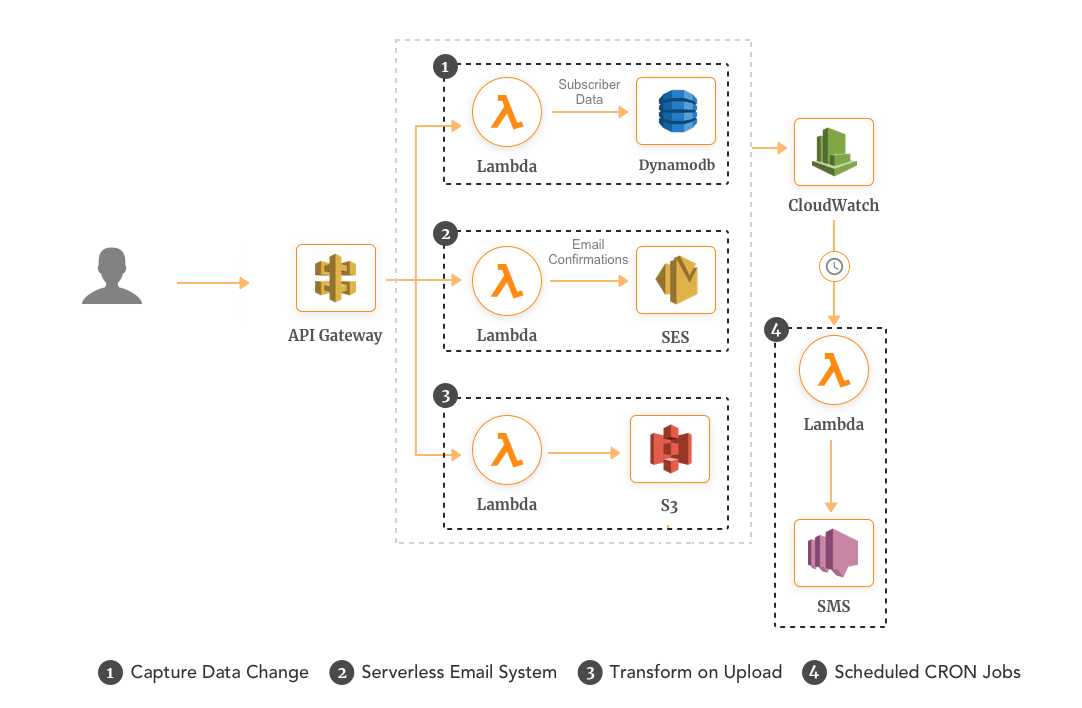
Amazon managed KMS

Customer Master Keys

**(9)**

**Lambda**

* AWS Lambda serverless compute service which lets you run code without provisioning or managing servers. You pay only for the compute time you consume.
* Just upload your code and Lambda takes care of everything required to run and scale your code with high availability
* Manage your virtual functions not really caring about the servers
* Run on demand
* Scaling is automated



**Billing:**

Pay per request first one million requests are free

$0.20 per one million requests.

compute time 0.00001667 for every GB-seconds used.

**AWS Lambda Languages:**

NodeJS, Python, Python3, Gr00vy, java, C sharp, Scala and GO

**AWS Lambda Integration**

Kinesis, API Gateway, DynamoDB, AWS S3, CloudWatch Events, CloudWatch logs, SNS

and Incognito

**AWS Lambda time limit:**

15 minutes (900 Seconds)

**Lambda Layer Version:**

* A Lambda layer is an archive containing additional code, such as libraries, dependencies, or even custom runtimes.
* Layers are deployed as immutable versions, and the version number increments each time you publish a new layer.
* When you include a layer in a function, you specify the layer version you want to use.

**Lambda Alias Version**

* A Lambda alias is like a pointer to a specific function version. Users can access the function version using the alias Amazon Resource Name (ARN).
* Versioning🡪Every time new code is deployed or the function configuration needs to be changed, it’s possible to create a new version and test before distributing the changes to function consumers.

**Use Cases:**

 **Data and analytics** – Suppose you are building an analytics application and storing raw data in a DynamoDB table. When you write, update, or delete items in a table, DynamoDB streams can publish item update events to a stream associated with the table. In this case, the event data provides the item key, event name (such as insert, update, and delete), and other relevant details. You can write a Lambda function to generate custom metrics by aggregating raw data.

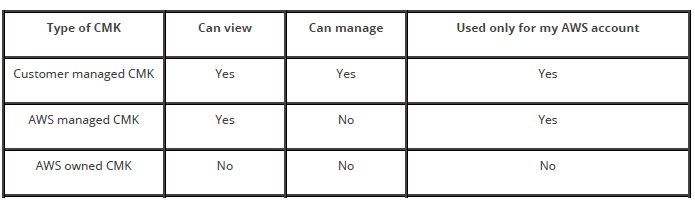
 **Websites** – Suppose you are creating a website and you want to host the backend logic on Lambda. You can invoke your Lambda function over HTTP using Amazon API Gateway as the HTTP endpoint. Now, your web client can invoke the API, and then API Gateway can route the request to Lambda.

**(10)**

**Key Management Service (KMS)**

**AWS** Key Management Service (**AWS KMS**) is a managed service that makes it easy for you to create and control customer master keys (CMKs), the encryption keys used to encrypt your data.

* Automated backups and automated snapshots don’t occur while a copy is executing in the same region for the same DB instance.



**(11)**

**AWS secrets**

AWS Secrets Manager helps you protect secrets needed to access your applications, services, and IT resources. The service enables you to easily rotate, manage, and retrieve database credentials, API keys, and other secrets throughout their lifecycle.

**(12)**

**Relational Database Service (RDS)**

Amazon Relational Database Service (Amazon RDS) makes it easy to set up, operate, and scale a relational database in the cloud.

* You can run your DB instance in several AZs, an option called a Multi-AZ deployment.
* You can have automated backups performed when you need them, or manually create your

own backup snapshot. You can use these backups to restore a database.

* Basic building block of RDS is the DB instance, which is an isolated database environment in

the cloud.

* The default backup retention period is one day if you create the DB instance using the RDS API or the AWS CLI, or seven days if you used the AWS Console.
* Automated backups and automated snapshots don’t occur while a copy is executing in the same region for the same DB instance.
* Read replicas for improved read performance

|  |  |
| --- | --- |
| Relational Database | Non-Relational Database |
| * Structured   + Stored as rows and columns | * Unstructured   + Stored as files etc. |
| * Automates few complicated tasks | * Ex:   + DynamoDB   + NoSQL DB |
| * Protects from accidental fail-over |
| * Periodical backups |
| * Disaster Recoveries |
| * Encryption |
| * Access Management |

**Which relational database engines does Amazon RDS support?**

**Amazon RDS database engines:**

* Amazon Aurora
* PostgreSQL
* MySQL
* MariaDB
* Oracle
* Microsoft SQL Server

**Encryption in RDS:**

Encryption at rest is supported for

* Amazon Aurora
* PostgreSQL
* MySQL
* MariaDB
* Oracle
* Microsoft SQL Server

**Q: Can we enable encryption on existing DB**

Encrypting existing DBs is not supported. To do this, you’ll need to create a new

encrypted instance and migrate data to it. The encryption key can be stored in KMS.

Q: **Which is the non-relational database supported in AWS**

Amazon DynamoDB is the NoSQL database supported by AWS

Q: **What is Encryption at rest?**

* At rest is server-side encryption
* Encryption in-Transit is Client-side encryption
* If master is not replicated, read replica is not encrypted

**Use cases for relational databases on Amazon include:**

 Finance data

 Transactions

 Data warehousing



**RDS backups:**

* Backups automatically enabled
* 7 days retention, can be increased to 35 days
* Transaction logs are backed up by RDS every 5 min

RDS storage autoscaling

* Dynamically increase
* Automatic scaling

|  |  |
| --- | --- |
| RDS read replicas | Multi AZ-disaster recovery |
| Upto 5 read replicas | Synchronous replication |
| Within AZ, cross AZ, cross region | One DNS name to avoid failover |
| Replication is asynchronous so reads are eventually consistent | Not used for scaling |
| Can be promoted to their own DB |  |
| No cost for data transfer to read replica within same region |  |

* Read replicas be setup as multi AZ for disaster recovery

**ELK: Elastic Search, Log Stash, Kibana**

**Elastic Search:**

* It is a free open-source engine (database tool) for searching logs, searching data, analytics purpose etc.
* We can use this for Monitoring, for connecting with AWS resources like CloudWatch, S3 etc.

**Log Stash:**

* Open Search Service – helps to upload data into AWS OSS

**Kibana:**

* Another Searching tool

**AWS Snowball:**

* AWS Snowball is a service that provides secure, rugged devices, so you can bring AWS computing and storage capabilities to your edge environments, and transfer data into and out of AWS. Those rugged devices are commonly referred to as AWS Snowball or AWS Snowball Edge devices.
* It is an edge computing, data migration, and edge storage device that comes in two options.

1. Snowball Edge Storage Optimized devices (52 vCPUs) provide both block storage and Amazon S3-compatible object storage, and
2. 40 vCPUs.

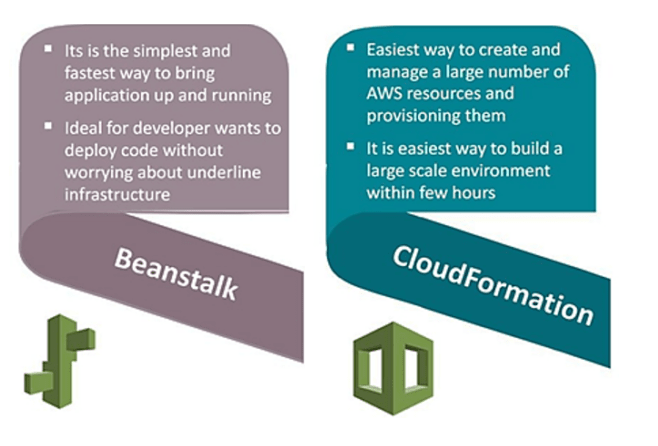
**AWS CloudFormation: (like Terraform)**

* It  helps you model and set up your AWS resources so that you can spend less time managing those resources and more time focusing on your applications that run in AWS.
* You create a template that describes all the AWS resources that you want (like Amazon EC2 instances or Amazon RDS DB instances), and CloudFormation takes care of provisioning and configuring those resources for you. You don't need to individually create and configure AWS resources and figure out what's dependent on what; CloudFormation handles that.
* Infrastructure as code

**AWS Elastic Beanstalk:**

* With AWS Elastic Beanstalk, you can quickly deploy and manage applications in the AWS Cloud without worrying about the infrastructure that runs those applications.
* It automatically handles the details of capacity provisioning, load balancing, scaling, and application health monitoring.
* Platform as service

**Beanstalk vs CloudFormation:**



**AWS Lambda vs. Elastic Beanstalk**

* Lambda can at maximum run for 15 minutes, whereas EB can run continuously. Generally, we deploy websites/apps on EB whereas lambda are generally used for triggered functionality like processing image when image gets uploaded to S3.
* Lambda can only handle one request at a time whereas number of concurrent requests EB can handle depends on your underlying infrastructure. So, if you are having say 100 requests, 100 lambdas will be created whereas these 100 requests can be handled by one underlying EC2 instance in EB
* Lambda is serverless (underlying infra is entirely abstracted from developer). Whereas EB is automation over infra provisioning. You can still see your EC2 instances, load balancer, auto scaling group etc. in your AWS console. You can even ssh/rdp to your instance and change running services. AWS EB allows you also to have your custom AMIs.
* Lambda is having issue of cold starts as in lambda, infra needs to be provisioned on demand by AWS, whereas in EB, you generally have EC2 instances already provisioned to handle your requests.

**Amazon SQS**

* A hosted queue that lets you integrate and decouple distributed software systems and components
* SQS supports both standard and FIFO queues
* SQS uses pull based (polling) not push based (SNS is push based)

**Amazon Elastic Container Service (Amazon ECS)**

* It is a highly scalable and fast container management service. You can use it to run, stop, and manage containers on a cluster.
* Your containers are defined in a task definition that you use to run individual tasks or tasks within a service
* A container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another. A Docker container image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries and settings.

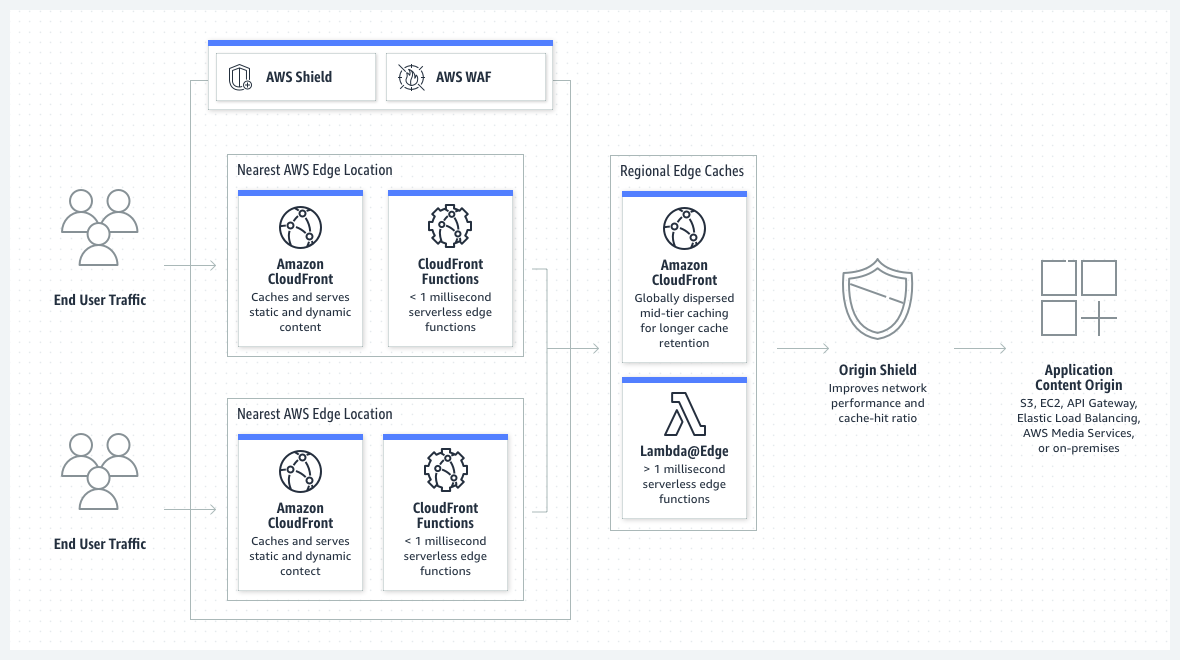
**Amazon Elastic Kubernetes Service (Amazon EKS)**

* It is a managed service that you can use to run Kubernetes on AWS without needing to install, operate, and maintain your own Kubernetes control plane or nodes.
* Kubernetes is an open-source system for automating the deployment, scaling, and management of containerized applications.

**Amazon CloudFront**

# Amazon CloudFront is a content delivery network (CDN) service with low latency and high transfer speeds built for high performance, security, and developer convenience.

* Speeds up distribution of static and dynamic web content
* Delivers content through worldwide network of data centers called edge locations
* You specify origin servers, like an S3 bucket or your own HTTP server, from which CloudFront gets your 􀃕les which will then be distributed from CloudFront edge locations all over the world.
* Objects are cached for 24 hours by default. You can invalidate 􀃕les in CloudFront edge caches even before they expire.

[](https://aws.amazon.com/cloudfront/)

## **What is UAT?**

**User Acceptance Testing (UAT)** is a type of testing performed by the end user or the client to verify/accept the software system before moving the software application to the production environment. UAT is done in the final phase of testing after functional, integration and system testing is done.

**Elasticache:**

ElastiCache is a web service that makes it easy to set up, manage, and scale a distributed in-memory data store or cache environment in the cloud.

* distributed in-memory cache environment in the AWS Cloud
* works with both the Redis and Memcached engines

|  |  |  |
| --- | --- | --- |
|  | Redis | Memchached |
| Encryption | Yes | Unsupported |
| cluster replication (create multiple copies of a primary cluster) | Yes | Unsupported |
| Mutli AZ for automatic failover | Yes | Unsupported |

**AWS private link:**

AWS PrivateLink provides private connectivity between VPCs, AWS services, and your on-premises networks, without exposing your traffic to the public internet. AWS PrivateLink makes it easy to connect services across different accounts and VPCs to significantly simplify your network architecture

These 2 developed separately, but have more recently found themselves intertwined.

* VPC Peering - applies to VPC
* PrivateLink - applies to Application/Service

With VPC Peering you connect your VPC to another VPC. Both VPC owners are involved in setting up this connection. When one VPC, (the visiting) wants to access a resource on the other (the visited), the connection need not go through the internet.

PrivateLink provides a convenient way to connect to applications/services by name with added security. You configure your application/service in your VPC as an AWS PrivateLink-powered service (referred to as an endpoint service). AWS generates a specific DNS hostname for the service. Other AWS principals can create a connection to your endpoint service after you grant them permission.

**AWS code pipeline**

AWS CodePipeline is a continuous delivery service you can use to model, visualize, and automate the steps required to release your software. You can quickly model and configure the different stages of a software release process. CodePipeline automates the steps required to release your software changes continuously.