**AWS Solutions Architect**

**EC2**

Private vs public vs Elastic IP

Elastic IP enables us to have a fixed public IP. But there are better design solutions available.

Continue using Existing Licenses == Dedicated Host

EC2 Placement Groups

When you launch a new EC2 instance, the EC2 service attempts to place the instance in such a way that all your instances are spread out across underlying hardware to minimize correlated failures. You can use placement groups to influence the placement of a group of interdependent instances to meet the needs of your workload. Depending on the type of workload, you can create a placement group using one of the following placement strategies:

* Cluster – packs instances close together inside an Availability Zone. This strategy enables workloads to achieve the low-latency network performance necessary for tightly-coupled node-to-node communication that is typical of HPC applications.
* Partition – spreads your instances across logical partitions such that groups of instances in one partition do not share the underlying hardware with groups of instances in different partitions. This strategy is typically used by large distributed and replicated workloads, such as Hadoop, Cassandra, and Kafka.
* Spread – strictly places a small group of instances across distinct underlying hardware to reduce correlated failures.

No charges for creating placement groups.

[AWS Elastic Network Interface](https://aws.amazon.com/blogs/aws/new-elastic-network-interfaces-in-the-virtual-private-cloud/)

An elastic network interface is a logical networking component in a VPC that represents a virtual network card. It can be attached to EC2 on same AZ.

EC2 Hibernate

When you hibernate an instance, Amazon EC2 signals the operating system to perform hibernation (suspend-to-disk). Hibernation saves the contents from the instance memory (RAM) to your Amazon Elastic Block Store (Amazon EBS) root volume. Amazon EC2 persists the instance's EBS root volume and any attached EBS data volumes. When you start your instance:

- The EBS root volume is restored to its previous state

- The RAM contents are reloaded

- The processes that were previously running on the instance are resumed

- Previously attached data volumes are reattached and the instance retains its instance ID

To enable EC2 Hibernate, the EC2 Instance Root Volume type must be an EBS volume and must be encrypted to ensure the protection of sensitive content.

EC2 Nitro

The AWS Nitro System is the underlying platform for our next generation of EC2 instances that enables AWS to innovate faster, further reduce cost for our customers, and deliver added benefits like increased security and new instance types. It has better networking options, enhanced networking, HPC, IPv6.

It’s a must for 64,000 IOPS – max 32k on non-nitro

vCPU

Capacity Reservation

On-Demand Capacity Reservations enable you to reserve compute capacity for your Amazon EC2 instances in a specific Availability Zone for any duration. This gives you the ability to create and manage Capacity Reservations independently from the billing discounts offered by Savings Plans or Regional Reserved Instances.

By creating Capacity Reservations, you ensure that you always have access to EC2 capacity when you need it, for as long as you need it. You can create Capacity Reservations at any time, without entering into a one-year or three-year term commitment. The capacity becomes available, and billing starts as soon as the Capacity Reservation is provisioned in your account. When you no longer need it, cancel the Capacity Reservation to release the capacity and to stop incurring charges.

By default, there’s no discount on capacity reservation. You can combine Capacity Reservation with Savings plan or reserved instances to get it.

**Spot Instance**

A Spot Instance is an unused EC2 instance that is available for less than the On-Demand price. Because Spot Instances enable you to request unused EC2 instances at steep discounts, you can lower your Amazon EC2 costs significantly. The hourly price for a Spot Instance is called a Spot price. The Spot price of each instance type in each Availability Zone is set by Amazon EC2 and adjusted gradually based on the long-term supply of and demand for Spot Instances.

A Spot Instance request is either one-time or persistent. If the spot request is persistent, the request is opened again after your Spot Instance is interrupted. If the request is persistent and you stop your Spot Instance, the request only opens after you start your Spot Instance. Therefore the option - "If a spot request is persistent, then it is opened again after your Spot Instance is interrupted" - is correct.

Spot Instances with a defined duration (also known as Spot blocks) are designed not to be interrupted and will run continuously for the duration you select. You can use a duration of 1, 2, 3, 4, 5, or 6 hours. In rare situations, Spot blocks may be interrupted due to Amazon EC2 capacity needs. Therefore, the option - "Spot blocks are designed not to be interrupted" - is correct.

If your Spot Instance request is active and has an associated running Spot Instance, or your Spot Instance request is disabled and has an associated stopped Spot Instance, canceling the request does not terminate the instance; you must terminate the running Spot Instance manually. Moreover, to cancel a persistent Spot request and terminate its Spot Instances, you must cancel the Spot request first and then terminate the Spot Instances. Therefore, the option - "When you cancel an active spot request, it does not terminate the associated instance" - is correct.

**Notes:**

* Dedicated Hosts are costlier than Dedicated Instances.
* An EC2 instance can be launched from either an instance store-backed AMI or an Amazon EBS-backed AMI. Instances that use Amazon EBS for the root device automatically have an Amazon EBS volume attached. By default, the root volume for an AMI backed by Amazon EBS is deleted when the instance terminates. The default behavior can be changed to ensure that the volume persists after the instance terminates. To change the default behavior, set the DeleteOnTermination attribute to false using a block device mapping.

**Storage**

EBS – It is a network drive, not physical drive. Can be attached to EC2 instance in same availability zone only.

EBS Snapshot – Used to move EBS data from one AZ/region to another. There’s EBS snapshot archive through which we can move a snapshot to archive tier that is 75% cheaper. But it takes 24 to 72 hours to restore. You can also specify a Recycle Bin for snapshots which retain snapshots deleted from last 1 day to 1 year.

[Types of EBS volumes](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-volume-types.html)

Only gp2/gp3(General Purpose SSD) and io1/io2(High performance SSD) can be used as root volumes. St1 (low cost HDD) and sc1 (lowest cost HDD) cannot be used.

**EBS Multi-Attach – io1/io2 (Provisioned IOPS SSD only)**

It allows to attach same EBS volume to multiple EC2 instances in the same AZ. Up to 16 Linux instances built on Nitro System. Can attach a multi-attach enabled EBS to windows also, but the OS does not recognize the data on the volume that is shared between the instances, which can result in data inconsistency.

Each instance has full read & write permission to the volume.

Use Case

* Achieve higher application availability in clustered Linux apps
* App must manage concurrent write ops.

Must use file system which is cluster-aware. Standard file system like XFS and EXT4 are not designed to be accessed simultaneously by multiple servers.

**EBS Encryption**

When you create encrypted EBS volume, you get following:

* Data at rest is encrypted inside the volume
* All the data in flight moving between the instance and the volume is encrypted
* All snapshots are encrypted
* All volumes created from the snapshot are encrypted

EBS encryption leverages keys from KMS (AES 256)

Copying an unencrypted snapshot allows encryption

**EBS DLM (Data Lifecycle Manager)**

It automates creation, retention and deletion of EBS snapshots.

**EFS (Elastic File system)**

It is managed NFS (network file system) available across AZ and can be connected to multiple EC2. Highly available, scalable, expensive (3\*gp2 - $), pay per use.

It is only compatible with Linux based AMI and not windows. It has encryption at rest using KMS. Capacity planning is not required, it scales automatically.

You can connect to Amazon EFS file systems from EC2 instances in other AWS regions using an inter-region VPC peering connection, and from on-premises servers using an AWS VPN connection.

Performance Modes

2 modes-

General Purpose – For majority of work loads. Ideal for latency sensitive use cases, like web-serving environment, Content Mgmt, and general file serving. This is selected by default.

Max I/O – For high throughput and operations per sec. The scaling is done with a tradeoff of slightly higher latencies for file metadata operations. Highly parallelized apps and workloads, such as big data analysis, media processing etc.

Performance mode once selected cannot be changed. Modes don’t have any charge.

**Notes**

* Other than HDD, all EBS volumes can be used as boot volumes. Also Instance Store can be used. EFS cannot be used.
* With Amazon EFS, you pay only for the resources that you use.
* Q - You have created a file system using Amazon Elastic File System (EFS) which will hold home directories for users. What else needs to be done to enable users to save files to the EFS file system?

A - Create a subdirectory for each user and grant read-write-execute permissions to the users. Then mount the subdirectory to the users’ home directory" is the correct answer.

**High Availability and Scalability**

**Elastic Load Balancers**

You need to enable AZ for your load balancers. When you enable it, the ELB creates a load balancer node in the AZ. If you register targets in an AZ but do not enable it, these registered targets do not receive traffic. It is recommended to enable multiple AZ for all load balancers. With an Application LB, it is a requirement that you enable at least 2 or more AZ.

AWS has 4 kinds of Elastic Load Balancers:

Classic (v1 - Deprecated) – Introduced in 2009. Works on HTTP, HTTPS(Layer 7), TCP(Layer 4), SSL (secure TCP)

Application(v2) – Introduced in 2016. Works on HTTP, HTTPS, WebSocket.

Network(v2) – Introduced in 2017. Works on TCP, TLS (secure TCP), UDP

Gateway – Introduced in 2020. Operates at Layer 3 (Network Layer) – IP Protocol. Uses GENEVE protocol on port 6081

Elastic Load balancers are region specific, so it is necessary to have them in each region in front of EC2.

**Load Balancer Listener**

A listener checks for the connection request 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. Each rule consists of a priority, one or more actions, and one or more conditions. When the conditions for a rule are met, then its actions are performed. You must define a default rule for each listener, and you can optionally define additional rules.

Listeners support the following protocols and ports:

* Protocols: HTTP, HTTPS
* Ports: 1-65535

You can use an HTTPS listener to offload the work of encryption and decryption to your load balancer so that your applications can focus on their business logic. If the listener protocol is HTTPS, you must deploy at least one SSL server certificate on the listener.

If you must ensure that the targets decrypt HTTPS traffic instead of the load balancer, you can create a Network Load Balancer with a TCP listener on port 443. With a TCP listener, the load balancer passes encrypted traffic through to the targets without decrypting it.

**Application Load Balancers**

It works on Layer 7 (HTTP)

ALB is given a DNS name and not a fixed IP address so that underlying infra can change without affecting the Route53 record.

Target Groups – This is where the request is sent once received by ALB. These could be the target groups:

* EC2 instances (can be managed by ASG) – HTTP
* ECS tasks (managed by ECS itself) – HTTP
* Lambda functions – HTTP request is translated into a JSON Event.
* Private IP addresses

ALB can route to multiple target groups. Health checks are done at target group level.

Target Group Routing

Also, we have routing tables to target different target groups:

Routing based on path in URL (example.com/users & example.com/posts)

Routing based on hostname in URL (one.example.com & other.example.com)

Routing based on Query String, headers (example.com/users?id=123&order=false)

**Network Load Balancer(v2)**

It works on Layer 4 and allows to:

* Forward TCP & UDP traffic to your instances
* Handle millions of requests per second
* Less latency ~ 100ms (vs 400 ms for ALB)

NLB has one static IP per AZ, and supports assigning Elastic IP (helpful for whitelisting specific IP)

Target Groups for NLB

* EC2
* IP Address – must be private
* ALB

**Cross Availability Zone Load Balancing**

ALB: Always on (can’t be disabled). No charges for inter AZ data.

NLB: Disabled by default. You pay charges ($) for inter AZ data if enabled.

CLB: Disabled by default. No charges for inter AZ data if enabled.

\*\* Both ALB and ASG can have EC2 instances in multi AZ

**Server Name Indication (SNI)** – It solves the problem of loading multiple SSL certificates onto one web server (to serve multiple websites). It’s a newer protocol and requires the client to indicate the hostname of the target server in the initial SSL handshake. The server will then find the correct certificate or return the default one. It only work for ALB and NLB, CloudFront and not for CLB.

**Auto Scaling Group**

They are free, you only pay for the underlying EC2 instance.

An Auto Scaling group can launch On-Demand Instances, Spot Instances, or both. You can specify multiple purchase options for your Auto Scaling group only when you configure the group to use a launch template. (We recommend that you use launch templates instead of launch configurations to make sure that you can use the latest features of Amazon EC2. It also enables versioning)

When instances are launched, if you specified multiple Availability Zones, the desired capacity is distributed across these Availability Zones. If a scaling action occurs, Amazon EC2 Auto Scaling automatically maintains balance across all the Availability Zones that you specify.

**Dynamic Scaling Policies**:

* Target Tracking Scaling: Eg – I want the average ASG CPU to stay around 40%
* Simple/Step Scaling: When CloudWatch alarm is triggered (example Queue Length>10), then add 2 units. Or when CloudWatch alarm is triggered (example QueueLength<2), then remove 1 unit.
* Scheduled Actions: Anticipate a scaling based on knows usage patterns.

AWS recommends you use Target Tracking over step-scaling for most use cases.

Let’s say we have 2 different types of scaling policies applied and both trigger at the same time, then AWS chooses the policy that provides the largest capacity of resources.

**Predictive Scaling**: Continuously forecast load using ML algo and schedule scaling ahead.

**Difference between Step Scaling and Simple Scaling**

The main difference between the policy types is the step adjustments that you get with step scaling policies. When step adjustments are applied, and they increase or decrease the current capacity of your Auto Scaling group, the adjustments vary based on the size of the alarm breach.

In most cases, step scaling policies are a better choice than simple scaling policies, even if you have only a single scaling adjustment.

The main issue with simple scaling is that after a scaling activity is started, the policy must wait for the scaling activity or health check replacement to complete and the cooldown period to end before responding to additional alarms. Cooldown periods help to prevent the initiation of additional scaling activities before the effects of previous activities are visible.

In contrast, with step scaling the policy can continue to respond to additional alarms, even while a scaling activity or health check replacement is in progress. Therefore, all alarms that are breached are evaluated by Amazon EC2 Auto Scaling as it receives the alarm messages.

Scaling Cooldown in Simple Scaling – After a scaling activity happens, you are in a cooldown period (default 300 sec) where ASG doesn’t launch or terminate any further instances.

[ASG Default Termination Policy](https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-instance-termination.html)

1. Find the AZ with max number of instances.
2. If there are multiple instances in this AZ, delete the one with oldest config. If launch config is not there, then oldest launch template.
3. Closest to billing hour.

So, ASG tries to balance the number of instances across AZ by default.

[Health Check Considerations](https://docs.aws.amazon.com/autoscaling/ec2/userguide/ec2-auto-scaling-health-checks.html#health-check-considerations)

[Temporarily remove instances from your Auto Scaling group](https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-enter-exit-standby.html)

**Notes**

* ALB are a great fit for micro services and container-based application (example: docker & Amazon ECS)
* Only Network Load Balancer provides both static DNS name and static IP. While, Application Load Balancer provides a static DNS name but it does NOT provide a static IP. The reason being that AWS wants your Elastic Load Balancer to be accessible using a static endpoint, even if the underlying infrastructure that AWS manages changes.
* When using an Application Load Balancer to distribute traffic to your EC2 instances, the IP address you'll receive requests from will be the ALB's private IP addresses. To get the client's IP address, ALB adds an additional header called "X-Forwarded-For" contains the client's IP address.
* You can’t attach an Elastic IP address to ALB.
* ASG can launch instance in multi-AZ but not multi region.
* If ALB has not registered any targets with the target group, you get HTTP 503: Service Unavailable.
* Q - An application is running on Amazon EC2 behind an Elastic Load Balancer (ELB). Content is being published using Amazon CloudFront and you need to restrict the ability for users to circumvent CloudFront and access the content directly through the ELB.

How can you configure this solution?

A - Create a VPC Security Group for the ELB and use AWS Lambda to automatically update the CloudFront internal service IP addresses when they change. Cannot use OAI as it is for S3 only.

* LB – Connection Draining and Idle Timeout, Deregistration Delay
* Q - A website uses web servers behind an Internet-facing Elastic Load Balancer. What record set should be created to point the customer’s DNS zone apex record at the ELB?

A - Create an A record that is an Alias, and select the ELB DNS as a target.

You cannot create an A record pointing to DNS name of the load balancer.

**Databases**

**RDS**

Managed Database service. Can’t SSH into the server. Storage backed by EBS (gp2 or io1).

Since it’s a managed service, storage increase automatically (RDS Storage Auto Scaling available since 2019). You must set Maximum Storage Threshold though

RDS Backups

* Automatically enabled in RDS.
* Automated Backups:

Daily full backup of the DB (during maintenance window)

Transaction logs are backed-up by RDS every 5 mins

Therefore, we can restore to any point in time (from oldest backup to 5 mins ago)

7 days retention (can be increased to 35 days)

DB snapshots – These are different from automated backups as these are created manually.

* Retention of snapshot is so long as you want.

RDS Read Replicas

* Can create 5 Read Replicas. Within AZ, Cross AZ or Cross Region
* There will be async replication, so reads are eventually consistent.
* Each replica can be promoted to their own DB.
* Apps must update conn string to leverage read replicas.

RDS Read Replica – Network Cost

Usually, there’s a network cost when data goes from one AZ to another.

For RDS read replicas within the same region, you don’t pay that fee. But cross region will incur some $

RDS Multi AZ

In this, there is a sync replication from Master DB (M) to another Standby database (S)(not used currently). There’s one DNS name and there’s automatic failover. So no manual intervention.

We can go from single AZ to multi AZ with zero downtime, with a single click.

RDS Read Replica vs Multi AZ

Very important topic. First imp difference is that Read replica is async replication while RDS Multi AZ is sync replication. Secondly, in the multi-AZ deployment, there’s a standby DB to which we don’t have read access from the application, it’s just for failover use. Third, Read Replica is for both HA and DR, while Multi AZ is for DR only.

**RDS Security - Encryption**

At Rest Encryption

* Possibility to encrypt the master and read replica with AWS KMS – AES 256 encryption.
* Encryption must be defined at launch time.
* If the master is not encrypted, RR cannot be encrypted.

In Flight Encryption

* SSL certificates to encrypt data to RDS in flight.
* Provide SSL options with trust certificates when connecting to database.

Encrypting RDS

* Snapshots of un-encrypted RDS DB are un-encrypted.
* Snapshots of encrypted RDS DB are encrypted.
* Can copy an unencrypted snapshot into an encrypted one.

So, to encrypt an un-encrypted RDS DB:

* Create a snapshot of un-encrypted DB.
* Copy the snapshot and enable encryption for this new one.
* Restore the database from the encrypted snapshot.

Access Management

* IAM policies help control who can manage AWS RDS (through the RDS API)
* Traditional username and password can be used to login to DB.
* IAM based auth can be used to login into RDS MySQL & PostgreSQL (Better Method)

**Notes**

* Ques – Can we setup Read replica’s as Multi AZ for DR? Ans – Yes
* Multi AZ deployment can be of 2 types – [Multi AZ DB instance](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Concepts.MultiAZSingleStandby.html) (one standby for failover, no read access to them) and [Multi AZ DB Cluster Deployment](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/multi-az-db-clusters-concepts.html) (2 standby for failover, allows read access).
* All read replicas have their own connection strings. Their DNS end points are different. But in case of Multi AZ, the master and failover share the same DNS end point.
* Each RDS Read Replica can be promoted to their own DB.
* Always a good idea to use IAM roles to connect to DB/RDS/S3 instead of storing credentials on Ec2 config file.

You can create read replica of RDS, but not for the DB created on EC2.

**Amazon Aurora**

It is a proprietary technology from AWS (not open sourced)

Postgres and MySQL are both supported as Aurora DB (that means your drivers will work as if Aurora was Postgres or MySQL)

It is “AWS Cloud Optimized” and claims 5x performance improvement over MySQL on RDS and 3x the performance of Postgres on RDS.

Storage grows in increments of 10GB, up to 128 TB.

Aurora can have 15 RR, and the replication process is faster.

You can’t SSH into the instances.

Failover in Aurora is instantaneous. It’s HA native

It stores 6 copies of your data (not DB) across 3 AZ. There is a shared storage volume here.

Generally, without specific configurations, Aurora is like multi-AZ RDS, where we have just 1 Master DB. And there’s automatic failover for master in less than 30 seconds.

Can have Master + 15 RR

Support for Cross Region Replication

**Working of Aurora**

Client apps talk to a “Written Endpoint”, which points to the master DB. If this master DB corrupts, writer endpoint will point to any of the RR.

You can have Auto scaling on the RR.

So, it can get difficult to keep track of all RR. So, we have a “reader endpoint”, which has connection to Load Balancing.

**Aurora Custom Endpoints**

Instead of Reader Endpoint, you can define your custom endpoints to do some specific tasks. Like some set of RR for analytical queries, some for reporting. Can you have different instance class per endpoint.

**Aurora Serverless** – Don’t need to provision capacity, DB grows automatically. Pay per second use.

**Aurora Multi-Master** – In case you need immediate failover for writer nodes (HA). In this, every node does R/W vs promoting a RR as the new master

**Aurora Cross Region Read Replica**

* Useful for DR
* Simple to put in place

But these days, we prefer to use another service called Aurora Global DB.

**Aurora Global Database**

* 1 primary region
* Up to 5 secondary (read-only) regions, replication lag is less than 1 second.
* Up to 16 RR per secondary region
* Helps for decreased latency.
* Promoting another region (for DR) has an RTO of < 1 minute

**Aurora Machine Learning**

Aurora is integrated with SageMaker (ML Models) and Comprehend (sentiment analysis)

Use Case : Fraud Detection, Ads Targeting, Sentiment Analysis, Product Recommendation.

**Notes:**

* For Amazon Aurora, each Read Replica is associated with a priority tier (0-15). In the event of a failover, Amazon Aurora will promote the Read Replica that has the highest priority (the lowest numbered tier). If two or more Aurora Replicas share the same priority, then Amazon RDS promotes the replica that is largest in size. If two or more Aurora Replicas share the same priority and size, then Amazon Aurora promotes an arbitrary replica in the same promotion tier.

**ElastiCache**

Managed Caching service, on top of Redis and Memcache. Redis is better supported than Memcache.

**ElastiCache – Redis Sorted Set**

For example, Gaming leaderboard is computationally complex thing to do.

There are Redis Sorted sets which guarantee both uniqueness and element ordering.

So, each time a new element is added, it’s ranked in real time, then added in correct order.

**DynamoDB – Read/Write Capacity Modes**

Control how you manage your table’s capacity (read/write throughput)

* Provisioned Mode (default)  
  You specify number of reads/writes per second

Need to plan capacity beforehand

Pay for provisioned Read Capacity Units (RCU) & Write Capacity Units (WCU)

Possibility to add auto-scaling mode for RCU and WCU

* On-Demand Mode

Read/Write automatically scales up/down with your workloads

No capacity planning needed

Pay for what you use, but more $

Great for unpredictable workloads

**DynamoDB Streams**

Ordered stream of item-level modifications (create/update/delete) in a table.

Stream records can:

* Send to Kinesis Data Stream
* Read by AWS Lambda
* Read by Kinesis Client Library applications

Data retention up to 24 hours

Use Cases

* React to changes in real time (welcome email to users)
* Analytics
* Insert into derivatives tables

**DynamoDB – Indexes**

There are 2 types:- Global Secondary Indexes (GSI) and Local Secondary Indexes (LSI). High Level: allow to query on attributes other than primary key

**Notes:**

* DynamoDB TTL (time to live) is like expiry time for DynamoDB tables.
* DynamoDB has the concept of transactions.

**Route 53**

Geolocation Routing Policy – Use when you want to route traffic based on the location of your users. You can use this to create records in a private hosted zone.

Geoproximity Routing Policy – Use when you want to route traffic based on the location of your resources and, optionally, shift traffic from resources in one location to resources in another.

IP-based Routing Policy – Use when you want to route traffic based on the location of your users, and have the IP addresses that the traffic originates from.

Route 53 Hosted Zone

A hosted zone is a container for records, and records contain information about how you want to route traffic for a specific domain, such as example.com, and its subdomains (acme.example.com, zenith.example.com). A hosted zone and the corresponding domain have the same name. There are two types of hosted zones:

Public hosted zones contain records that specify how you want to route traffic on the internet.

Private hosted zones contain records that specify how you want to route traffic in an Amazon VPC.

CNAME vs Alias

CNAME only works for non-root domain, while alias works for both. You should also note that Route 53 doesn't charge for alias queries to AWS resources, but Route 53 does charge for CNAME queries. Additionally, an alias record can only redirect queries to selected AWS resources such as S3 buckets, CloudFront distributions, and another record in the same Route 53 hosted zone; however a CNAME record can redirect DNS queries to any DNS record. So, you can create a CNAME record that redirects queries from app.covid19survey.com to app.covid19survey.net.

**S3**

Max object size – 5TB. If uploading more than 5GB, must use “multi-part upload”.

There are 4 methods of encrypting objects in S3:

* SSE-S3 (Amazon S3 Managed Keys) – Encrypts S3 objects using keys handled & managed by AWS.
* SSE-KMS – Leverage AWS Key Management Service to manage encryption keys. More secure than SSE-S3 and some additional benefits. Audit trail available.
* SSE-C (Customer Provided Keys) – When you want to manage your own encryption keys. HTTPS must be used. Key must be provided in HTTP header for every HTTP request made.
* Client-Side Encryption – Client library such as Amazon S3 Encryption Client must be used to encrypt the objects before sending to S3. In this, keys/CMK can only be stored in either AWS KMS or in your application.

Encryption in transit – Use SSL/TLS and use HTTPS.

Link - <https://docs.aws.amazon.com/AmazonS3/latest/userguide/serv-side-encryption.html>

**S3 Security**

User Based

* IAM policies: which API calls should be allowed for a specific user from IAM console.

Resource Based

* Bucket Policies – bucket wide rules from the S3 console – allow cross account. These are JSON based policies which have allow/deny effect. It can be used to force objects to be encrypted at upload time.
* Object Access Control List (ACL) – finer grain
* Bucket Access Control List (ACL) – less common

Note: An IAM principal (user or role) can access an S3 object if:

* The user IAM permissions allow it OR the resource policy allows it
* AND there’s no explicit DENY

You can configure S3 to obtain access only from CloudFront using Origin Access policy

**Networking:** It supports VPC endpoints.

**Logging and Audit:** S3 access logs can be stored in other S3 bucket. Also, API calls can be logged in AWS Cloud Trail.

**User Security:** MFA Delete in versioned bucket to delete object.

We need to enable CORS when hosting static websites on S3.

**Forcing Encryption**

Different Ways of Forcing Encryption in S3:

* Use Default Encryption
* Use Bucket Policy and refuse any API call to PUT an S3 object without encryption headers

Bucket policies are evaluated before Default Encryption.

S3 Access Logs: Set some other bucket for logging access to the current bucket.

**S3 Replication**

* Must enable versioning in source and destination.
* Cross Region Replication (CRR)
* Same Region Replication (SRR)
* Buckets can be in different accounts
* Copying is Asynchronous
* Must give proper IAM permission to S3

After activating S3 Replication, only new objects are replicated. Optionally, you can replicate existing objects also using S3 Batch Replication.

For delete operations, can replicate delete markers from source to target. But deletions with version ID are not replicated (to avoid malicious deletes)

S3 Pre-Signed URLs – Can also upload data using this, bypassing the web server.

S3 Storage Classes

S3 Lifecycle Rules

* S3 One Zone – IA costs 20% less than S3 Standard – IA. But immediate availability of data will be less
* But you cannot move an object to One Zone – IA before keeping it in Standard tier for at least 30 days.
* S3 Intelligent Tiering, S3 Standard IA, S3 One Zone IA storage classes have minimum 30 day storage charge.

[General Considerations](https://docs.aws.amazon.com/AmazonS3/latest/userguide/lifecycle-transition-general-considerations.html)

**S3 Performance**

If you use SSE-KMS, performance might be impacted by KMS limits. But you can request quota increase using the Service Quotas Console.

Multi-Part Upload – Recommended for files > 100MB, must for > 5GB. Can help parallelize uploads.

S3 Transfer Acceleration – Increase transfer speed by transferring file to an AWS edge location which will forward the data to the S3 bucket in target region. Compatible with multi-part. Usually, this is preferred mode for objects > 1GB, otherwise cloudfront

For downloads, can use S3 byte-range fetches

**S3 Select & S3 Glacier**

Retrieve less data using SQL by performing server-side filtering.

Can filter rows & columns (simple SQL statements).

Less network transfer, less CPU cost client side.

**S3 Event Notification**

For events like object created, Removed, Replication etc, we can create Event Notifications which can call following:

* SNS
* SQS
* Lamda
* EventBridge (which can further call 18 other services)

**Access to S3 Buckets**

For S3 bucket in same account as the user, we can use either bucket policy or user policy.

For cross account access, we can use bucket policy only. Bucket policy is a type of resource based policy.

**Notes**

* Buckets can be in one of the 3 states: unversioned (default), versioning-enabled, or versioning suspended. Once you version-enable a bucket, it can never return to an unversioned state. You can only suspend versioning.
* Let’s say you want to move data from on-prem to AWS for archival purpose. You can’t move it from Snowball directly to Glacier. It must go to S3 standard first. You can configure 0 day lifecycle policy so that you’re not charged to s3 standard usage.
* To encrypt an object at the time of upload, you need to add a header called x-amz-server-side-encryption to the request to tell S3 to encrypt the object using SSE-C, SSE-S3, or SSE-KMS.
* To enforce object encryption, create an S3 bucket policy that denies any S3 Put request that does not include the x-amz-server-side-encryption header. There are two possible values for the x-amz-server-side-encryption header: AES256, which tells S3 to use S3-managed keys, and aws:kms, which tells S3 to use AWS KMS–managed keys.

**EC2 Instance Metadata**

You can visit <http://169.254.169.254/latest/meta-data>/ from EC2 instance to “learn about themselves” without using an IAM role for that purpose. You can retrieve the IAM role from the metadata but not IAM policy.

Metadata = Info about EC2 Instance

Userdata = Launch script of the EC2 instance

**ECS**

2 Launch types:

* EC2 – Each EC2 instance within the ECS cluster will have ECS Agent to run container.
* Fargate (Serverless)

ECS with EC2 launch type is charged based on EC2 instances and EBS volumes used. ECS with Fargate launch type is charged based on vCPU and memory resources that the containerized application requests.

ECS Service also auto scales based on several policies like Target tracking etc.

**EKS**

Amazon EKS is a managed service that can be used to run Kubernetes on AWS. Kubernetes is an open-source system for automating the deployment, scaling, and management of containerized applications. Applications running on Amazon EKS are fully compatible with applications running on any standard Kubernetes environment, whether running in on-premises data centers or public clouds. This means that you can easily migrate any standard Kubernetes application to Amazon EKS without any code modification.

**CloudFront**

It is a CDN, improves read performance, content is cached at the edge. 216 point of presence globally (edge locations). Ddos protection, integration with Shield, WAF.

**Cloud Front Origins**:

S3 bucket

* For distributing files and caching them at the edge.
* Enhanced security with CF Origin Access Identity (OAI)
* CF can be used as ingress (to upload files to S3)

Custom Origin (HTTP)

* ALB
* EC2 instance
* S3 website (must first enable the bucket as a static S3 Website)

**CloudFront Geo Restriction**

You can restrict who can access your distribution:

Whitelist: Allow your users to access your content only if they’re in one of the countries on a list of approved countries.

Blacklist: for banned countries

With Amazon CloudFront you can set the price class to determine where in the world the content will be cached. One of the price classes is “U.S, Canada and Europe”.

Custom Origin - A custom origin can point to an on-premises server and CloudFront is able to cache content for dynamic websites. CloudFront can provide performance optimizations for custom origins even if they are running on on-premises servers. These include persistent TCP connections to the origin, SSL enhancements such as Session tickets and OCSP stapling.

Additionally, connections are routed from the nearest Edge Location to the user across the AWS global network. If the on-premises server is connected via a Direct Connect (DX) link this can further improve performance.

Origin Groups: Used to set up a secondary origin (e.g EC2 or S3) so that in case the primary origin fails, we can failover to secondary.

You can configure a single CloudFront web distribution to serve different types of requests from multiple origins. For example, if you are building a website that serves static content from an Amazon Simple Storage Service (Amazon S3) bucket and dynamic content from a load balancer, you can serve both types of content from a CloudFront web distribution.

**Field-level encryption** allows you to enable your users to securely upload sensitive information to your web servers. The sensitive information provided by your users is encrypted at the edge, close to the user, and remains encrypted throughout your entire application stack. This encryption ensures that only applications that need the data—and have the credentials to decrypt it—are able to do so.

To use field-level encryption, when you configure your CloudFront distribution, specify the set of fields in POST requests that you want to be encrypted, and the public key to use to encrypt them. You can encrypt up to 10 data fields in a request. (You can’t encrypt all of the data in a request with field-level encryption; you must specify individual fields to encrypt.)

**Notes:**

* Video on Demand/Live Streaming -> Use CloudFront to speed up.

**AWS Lambda**

It currently supports 1000 concurrent executions per AWS account per region. You need to contact AWS Support to raise this account limit. Also, there is a time limit of 15 mins per execution. Environment variables can be 4KB.

You can deploy Lambda function as a container image.

Lambda functions always operate from an AWS-owned VPC. By default, your function has the full ability to make network requests to any public internet address — this includes access to any of the public AWS APIs. For example, your function can interact with AWS DynamoDB APIs to PutItem or Query for records. You should only enable your functions for VPC access when you need to interact with a private resource located in a private subnet. An RDS instance is a good example.

Once your function is VPC-enabled, all network traffic from your function is subject to the routing rules of your VPC/Subnet. If your function needs to interact with a public resource, you will need a route through a NAT gateway in a public subnet.

**Lambda Layer**

Lambda layers provide a convenient way to package libraries and other dependencies that you can use with your Lambda functions. Using layers reduces the size of uploaded deployment archives and makes it faster to deploy your code.

A layer is a .zip file archive that can contain additional code or data. A layer can contain libraries, a custom runtime, data, or configuration files. Layers promote code sharing and separation of responsibilities so that you can iterate faster on writing business logic.

**API Gateway**

Q - When using throttling controls with API Gateway what happens when request submissions exceed the steady-state request rate and burst limits?

A - API Gateway fails the limit-exceeding requests and returns “429 Too Many Requests” error responses to the client

**Lambda@Edge**

Let’s say you have deployed a CDN using CloudFront. Here, you can use Lambda@Edge to run global AWS Lambda alongside. It can also be used to implement request filtering before reaching your application.

4 types of requests you can modify:

Viewer Request: After CF receives a request from a viewer.

Origin Request: Before CF forwards the request to the origin

Origin Response: After CF received the response from the origin

Viewer Response: Before CF forwards the response to the viewer

Use Cases

* Website security and Privacy
* Dynamic Web Application at the Edge
* SEO
* Intelligent Route across origins and Data centers
* Bot Mitigation at Edge
* Real-Time Image Transformation
* A/B Testing
* User Authentication and Authorization
* User Prioritization
* User Tracking and Analytics

**Messaging**

**SQS**

The name of a FIFO queue must end with .fifo

By default, FIFO queues support up to 300 messages per second (300 send, receive, or delete operations per second). When you batch 10 messages per operation (maximum), FIFO queues can support up to 3,000 messages per second.

You cannot convert a standard queue to FIFO queue. You need to delete and recreate

Default Message Visibility Timeout = 30 seconds. If a message is not processed within the visibility timeout, it will be processed twice. So, let’s say a consumer is processing a message and it needs extra time to process it, we can call the “ChangeMessageVisibility” API to get more time.

Delay Queue – Max 15 mins delay. By Default it is 0 seconds. Can override using “DelaySeconds” param.

Long Polling – used to decrease API calls to SQS, while increasing the efficiency and latency of your app. Wait time can be from 1 to 20 sec. WaitTimeSecond – API

SQS Request-Response System – While sending a request they add “CorrelationID” and “Reply to” queue. There is a client called SQSTemporary Queue Client.

**SQS Access Policy:**

2 use cases

* Cross Account access
* Publish S3 Event Notifications to SQS Queue

**SQS + SNS Fan Out Pattern**

Use Case – Let’s say you want your application to send same message to several SQS queues. If you’re application do it, there could be problems in case of app failure.

In the Fan out pattern, the idea is that your app push once in SNS, receive in all SQS queues that are subscribers. Make sure SQS Access Policies allow for SNS to write.

Second Use Case – S3 Events to Multiple Queues: S3 had limitation that, for the same combo of event type (eg object created) and prefix (eg image/) you can have only 1 S3 Event rule (and it can write in just one SQS). We can use fan out here to send the event to multiple queues.

Can do message filtering in SNS

**Monitoring**

**CloudWatch**

CloudWatch Metric

By default, EC2 instance metric every 5 mins. But can enable EC2 Detailed monitoring (cost), you get data every 1 min. EC2 memory usage is by default not pushed, must be pushed from inside the instance as a custom metric

Custom Metric

CloudWatch Dashboards

CloudWatch Logs – Log groups, Log streams, can send logs to S3, KDS, KDF, Lambda, ElasticSearch.

CloudWatch Logs for EC2 – By default, no logs from EC2 to CloudWatch. Need to run CloudWatch agent on EC2 to push the log files. Make sure IAM permissions are correct. Can be set on-prem server too.

CloudWatch Unified Agent – Next level metrics

CloudWatch Alarm – to trigger notifications for any metric. Alarm States: OK, INSUFFICIENT\_DATA, Alarm.

CloudWatch Alarm Targets

* Stop, Terminate, Reboot or Recover an EC2 instance.
* Trigger Auto Scaling Action
* Send Notification to SNS (from which you can do pretty much everything)

CloudWatch Events

Amazon EventBridge (Next evolution of CloudWatch Events)

CloudTrail – Provides governance, compliance, and audit for AWS account. Enabled by default. Gets history of events/API calls made within your AWS account by console, SDK, CLI, AWS Services.

If a resource is deleted in AWS, investigate CloudTrail first.

2 types of events: management and data.

CloudTrail Insights

AWS Config – Helps with auditing and recording compliance of AWS account. Helps record configurations and changes over time. You can receive SNS notification for any change. It is a per region service.

It does not prevent actions from happening.

Example of Config Rule – evaluate if each EBS disk is of gp2. Evaluate if each EC2 instance is t2.micro

Can you AWS managed config rules (over 75) or custom.

**Security**

**AWS KMS**

KMS uses Symmetric keys (AES-256 keys). Ability to audit key usage using Cloud Trail.

3 types of customer Manager keys (CMK):

* AWS Managed Service Default CMK: free
* User keys created in KMS: $1/month
* User keys imported (must be 256-bit symmetric): $1/month

Never ever store your secrets in plaintext, especially in code.

It can help in encrypting up to 4KB of data per call. If >4KB, use envelope encryption.

**KMS Key Policies**

They are like S3 bucket policies.

Default KMS Key Policy:

* Created if you don’t provide a specific KMS Key policy
* Complete access to the key to the root user = entire AWS account.
* Give access to the IAM policies to the KMS Key

Custom KMS Key Policy:

* Define users, roles that can access the KMS key.
* Define who can administer the key.

To give access to KMS to someone:

* Make sure the Key policy allows the user
* Make sure the IAM policy allows the API calls

KMS keys are bound to a specific region. So, if you want to copy (eg an encrypted EBS) from one region to another, do this:

* The EBS volume is currently encrypted with let’s say KMS Key A. Create its snapshot, same Key A will be used.
* Copy this snapshot across the region, and now specify Key B.
* Restore the snapshot, and it’ll be encrypted with key B.

Copying encrypted Snapshots across accounts:

* Create a Snapshot, encrypted with your own CMK.
* Attach a KMS Key Policy to authorize cross-account access.
* Share the encrypted snapshot.
* (in target) create a copy of the snapshot, encrypt it with a KMS key in your account.
* Create a volume from the snapshot.

**Notes:**

* What if you delete a CMK accidentally? Deleting it is potentially a dangerous and destructive thing to do. Therefore, AWS KMS enforces a waiting period. To delete a CMK in KMS you schedule key deletion. Can set waiting period from a minimum of 7 days to max 30 days. Default – 30. During the waiting period, CMK status and key state is Pending deletion.
* AWS Secrets manager allows force rotation of secrets every X days, but not SSM Parameter store. Also it has integration with Amazon RDS and Aurora. Secret Manager is newer and better service.

**AWS CMK (Customer Master Keys)**

Customer Master Keys are the primary resources in AWS KMS. You can use a CMK to encrypt and decrypt up to 4 KB of data. However, most commonly, you’ll use CMK to generate, encrypt, and decrypt the data keys that you use outside of AWS KMS to encrypt your data.

A customer master key (CMK) is a logical representation of a master key. The CMK includes metadata, such as the key ID, creation date, description, and key state. The CMK also contains the key material used to encrypt and decrypt data.

AWS KMS supports symmetric and asymmetric CMKs. A symmetric CMK represents a 256-bit key that is used for encryption and decryption. An asymmetric CMK represents an RSA key pair that is used for encryption and decryption or signing and verification (but not both), or an elliptic curve (ECC) key pair that is used for signing and verification.

AWS KMS supports 3 types of keys:

* Customer managed CMK
* AWS managed CMK
* AWS owned CMK

**KMS Automatic Key Rotation**

This exists for Customer Managed CMK only, not AWS managed CMK.

If enabled, key rotation happens every 1 year.

Previous key is kept active so you can decrypt old data.

New key has same CMK ID (only the backing key is changed).

**KMS Manual Key Rotation**

When you want to rotate key every 90 days, 180 days etc.

New key has different CMK ID

Better to use aliases in this case to hide the change of key for the application

**AWS Shield (Work at network and transport layer 3 and 4, also application layer 7)**

AWS Shield is a managed Distributed Denial of Service (DDoS) protection service that safeguards applications running on AWS.

There are two tiers of AWS Shield - Standard and Advanced:

Shield Standard (Layer 3, 4)

All AWS customers benefit from the automatic protections of AWS Shield Standard, at no additional charge. AWS Shield Standard defends against most common, frequently occurring network and transport layer DDoS attacks that target your web site or applications.

Shield Advanced (Layer 7 also)

For higher levels of protection against attacks targeting your applications running on Amazon Elastic Compute Cloud (EC2), Elastic Load Balancing (ELB), Amazon CloudFront, AWS Global Accelerator and Amazon Route 53 resources, you can subscribe to AWS Shield Advanced.

AWS Shield Advanced also gives you 24x7 access to the AWS Shield Response Team (SRT)

AWS Shield Advanced is available globally on all Amazon CloudFront, AWS Global Accelerator, and Amazon Route 53 edge locations. You can protect your web applications hosted anywhere in the world by deploying Amazon CloudFront in front of your application.

**AWS WAF (Layer 7, HTTP)**

Protect your web applications from common web exploits.

Based on conditions that you specify, such as the values of query strings or the IP addresses that requests originate from, CloudFront responds to requests either with the requested content or with an HTTP status code 403 (Forbidden).

Deploy on Application Load Balancer, API Gateway, Cloud Front

Define Web ACL:

* Rules can include: IP Address, HTTP Headers, HTTP body, URI strings
* Protect from common attacks – SQL injection and XSS
* Size Constraints, geo-match (block countries)
* Rate based rules (to count occurrences of events) – for DDoS protection.

You can use AWS WAF with your Application Load Balancer to allow or block requests based on the rules in a web access control list (web ACL). Geographic (Geo) Match Conditions in AWS WAF allows you to use AWS WAF to restrict application access based on the geographic location of your viewers.

**AWS Firewall Manager**

Manages rules in all accounts of AWS organization.

* WAF rules
* AWS Shield Advanced
* Security Groups for EC2 and ENI resources in VPC

**Guard Duty**

Intelligent Threat discovery to protect AWS Account. Uses ML, anomaly detection, 3rd Party data. One click to enable (30 day trial), no need to install software.

Input data includes:

* CloudTrail Event Logs – unusual API calls, unauthorized deployments.
* VPC Flow Logs
* DNS Logs
* Kubernetes Audit Logs

**Amazon Inspector**

It is for Automated Security Assessments. For:

* EC2 Instance: Leverage AWS System Manager agent, unintended network accessibility, analyze the running OS against known vulnerabilities.
* Container push to Amazon ECR: Assessment of container as they are pushed

Reporting and integration with AWS Security Hub

Send findings to Amazon Event Bridge

**AWS STS – Security Token Service**

Allows to grant limited and temporary access to AWS resources.

Token is valid up to 1 hour (must be refreshed)

Assume Role

* Within your account: for enhanced security
* Cross account access: assume role in target account to perform actions there

AssumeRoleWithSAML

* Return credentials for users logged with SAML

AssumeRoleWithWebIdentity

* Returns creds for users logged with an IdP (facebook, Google, IODC compatibles..)
* AWS recommends against using this and use Cognito instead.

GetSessionToken

* For MFA, from a user or AWS account root user

**Identity Federation (It’s a generic Term)**

Identity federation is a system of trust between two parties for the purpose of authenticating users and conveying the information needed to authorize their access to resources. In this system, an identity provider (IdP) is responsible for user authentication, and a service provider (SP), such as a service or an application, controls access to resources. By administrative agreement and configuration, the SP trusts the IdP to authenticate users and relies on the information provided by the IdP about them. After authenticating a user, the IdP sends the SP a message, called an assertion, containing the user's sign-in name and other attributes that the SP needs to establish a session with the user and to determine the scope of resource access that the SP should grant. Federation is a common approach to building access control systems that manage users centrally within a central IdP and govern their access to multiple applications and services acting as SPs.

You can use two AWS services to federate your workforce into AWS accounts and business applications: AWS Single Sign-On (SSO) or AWS Identity and Access Management (IAM). AWS SSO is a great choice to help you define federated access permissions for your users based on their group memberships in a single centralized directory. If you use multiple directories or want to manage the permissions based on user attributes, consider AWS IAM as your design alternative.

**Identity Federation in AWS**

Federation lets users outside the AWS to assume temporary role for accessing AWS resources.

These users assume identity provided access role.

Federation can have many flavors:

* SAML 2.0 (Not preferred, preferred way is Single Sign on)
* Custom Identity Broker
* Web Identity Federation with Amazon Cognito
* Web Identity Federation without Cognito (preferred way - Cognito)
* Single Sign On
* Non-SAML with AWS Microsoft AD

Using federation, you don’t need to create IAM users (user management is outside of AWS)

These flavors connect with AWS STS to give a URL with token which can be used by the user.

**AWS Directory Services**

**AWS Managed Microsoft AD**: when you have users on both on-prem and AWS

**AD Connector**: Users are on-prem only and we only want to proxy request

**Simple AD:** Cannot be joined with on-prem, users are on AWS only.

**AWS Organizations**

Global Service, allows to manage multiple AWS accounts

The main account is called master account, you can’t change it. Other accounts are member a/c.

Member accounts can only be a part of 1 organization

Consolidated billing across all accounts – single payment method

Pricing benefits from aggregated usage (volume discounts for EC2, S3..)

**Service Control Policies (SCP)**

Whitelist or blacklist IAM Actions

Applied at OU or Account level

Does not apply to the Master Account

SCP is applied to all the Users and Roles of an account, including root.

It doesn’t affect service linked roles, which enable other AWS services to integrate with AWS Org.

SCP must have explicit allow (does not allow anything by default)

Use case

* Restrict access to certain services (eg can’t use EMR)
* Enforce PCI compliance by explicitly disabling services

If you want to move master account of old org to join a new org, first move all member accounts. Then delete the organization, and then invite the old master account to the new org.

**IAM Advanced**

Resource based policies can be applied to S3 bucket, SNS Topics, SQS Queues.

IAM Role vs Resource Based Policy

Case Study:

Let’s say there’s a user in Account A who wants to access S3 bucket in account B.

There are 2 ways of doing this. Either make the Account A assumeRole in Account B with STS or create S3 bucket policy to allow access from account A.

Both ways are different and here’s how it’ll behave:

* When you assume a role, you give up your original permission and take the permissions assigned to that role.
* When using resource based policy, the principal doesn’t have to give up his permissions.

Example: User in Account A needs to scan a DynamoDB table in Account A and dump it in an S3 bucket in Account B. So, in this case Resource Based Policy is preferred.

**IAM Policy Evaluation Logic**

First, we evaluate all applicable policies. Then SCP. Resource Based Policy. Identity Based Policy on the principal. IAM Permission Boundary. Session Policies

As soon as you get any explicit deny, the decision would be deny.

**HPC (High Performance Computing)**

Use EC2 in cluster setup.

Use Elastic Fabric Adapter (EFA). It is an AWS Elastic Network Adapter (ENA) with added capabilities. It is ideal for tightly coupled app as it uses the Message Passing Interface (MPI). (Elastic Network Interface (ENI) is the basic type of adapter).

ENA – provides Enhanced Networking, does provide high bandwidth and low inter-instance latency but does not support the features for a tightly-coupled app that EFA does.

**AWS Storage Gateway**

It is a hybrid cloud storage service that provide on-premises access to virtually unlimited cloud storage.

Storage Gateway provides a standard set of storage protocols such as iSCSI, SMB, and NFS, which allow you to use AWS storage without rewriting your existing applications. It provides low-latency performance by caching frequently accessed data on premises, while storing data securely and durably in Amazon cloud storage services. Storage Gateway optimizes data transfer to AWS by sending only changed data and compressing data. Storage Gateway also integrates natively with Amazon S3 and Amazon FSx for Windows File Server cloud storage, which makes your data available for in-cloud processing, AWS Identity and Access Management (AWS IAM) for securing access management to services and resources, AWS Key Management Service (AWS KMS) for encrypting data at rest in the cloud, Amazon CloudWatch for monitoring, and AWS CloudTrail for logging account activity.

**3 types**

* **File Gateway** – Uses SMB and NFS protocol. Enables you to store and retrieve objects in S3 using file protocols such as NFS and SMB. Objects written through S3 File gateway can be directly accessed in S3. Integration with AD for user auth.

It offers 2 additional features:

Amazon S3 File Gateway - Amazon S3 File Gateway presents a file-based interface to Amazon S3, which appears as a network file share. It enables you to store and retrieve Amazon S3 objects through standard file storage protocols. File Gateway allows your existing file-based applications or devices to use secure and durable cloud storage without needing to be modified. With S3 File Gateway, your configured S3 buckets will be available as Network File System (NFS) mount points or Server Message Block (SMB) file shares. Your applications read and write files and directories over NFS or SMB, interfacing to the gateway as a file server. In turn, the gateway translates these file operations into object requests on your S3 buckets. Your most recently used data is cached on the gateway for low-latency access, and data transfer between your data center and AWS is fully managed and optimized by the gateway. Once in S3, you can access the objects directly or manage them using S3 features such as S3 Lifecycle Policies and S3 Cross-Region Replication (CRR). You can run S3 File Gateway on-premises or in EC2.

Amazon FSx File Gateway - enables you to store and retrieve files in Amazon FSx for Windows File server using the SMB protocol.

* **Volume Gateway** – provides block storage to your on-premise apps using iSCSI connectivity. Data is stored in S3 but can create EBS Snapshots also.

It is of 2 types:

Cached volumes – You store your data in S3 and retain a copy of frequently accessed data subset locally. It offers substantial cost saving on primary storage and also provides low latency access to frequently used data.

Stored Volumes – If you need low latency access to entire dataset, first configure your on-premise gateway to store all your data locally. Then asynchronously back up point-in-time snapshots of this data to Amazon S3. This offers inexpensive backup that you can recover later. Great for DR.

* **Tape Gateway** – provides your backup apps with an iSCSI Virtual Tape Library (VTL) interface, consisting of a virtual media changer, virtual drives and virtual tapes.

**Storage Gateway – Hardware Appliance**

Using SG means you need on-prem virtualization. Otherwise, you can use a Storage Gateway Hardware Appliance. Can buy it on amazon.com. Works with all 3 types of gateways.

So if you don’t need on-prem virtualization => Hardware Appliance

[**AWS Datasync vs Storage Gateway**](https://www.linkedin.com/pulse/aws-storage-gateway-vs-datasync-service-when-use-which-sandeep-bhatia/)

Crux – It you want to store data on both on-prem and cloud, and need quick/low latency access to it, use Storage gateway. But it you want to perform some action on data, like data analytics etc, use Datasync. Also Storage gateway always send data to S3, while datasync can send to EFS, EBS, Fsx etc

**Virtual Private Cloud (VPC)**

* You cannot associate an ALB directly to a private subnet. Need to have a public subnet in between.

**VPC Endpoint**

Allows to inter-connect AWS services privately, without going over internet. It is powered by AWS PrivateLink

Gateway Endpoint – S3 and DynamoDB. Provisions a gateway and must be used as a target in route table.

Interface Endpoint – All other AWS services. Provisions an ENI as an entry point.

**Route Table**

A route table contains a set of rules, called routes, that determine where network traffic from your subnet or gateway is directed.

Main route table—The route table that automatically comes with your VPC. It controls the routing for all subnets that are not explicitly associated with any other route table.

Subnet route tables - Your VPC has an implicit router, and you use route tables to control where network traffic is directed. Each subnet in your VPC must be associated with a route table, which controls the routing for the subnet (subnet route table). You can explicitly associate a subnet with a particular route table. Otherwise, the subnet is implicitly associated with the main route table. A subnet can only be associated with one route table at a time, but you can associate multiple subnets with the same subnet route table.

Gateway route table—A route table that's associated with an internet gateway or virtual private gateway.

**Internet Gateway**

It is a horizontally scaled, redundant, and highly available VPC component that allows communication between your VPC and the internet.

It serves 2 purposes:

* To provide a target in your VPC route tables for internet-routable traffic
* To perform Network Address Translation (NAT) for instances that have been assigned public IPV4 addresses.

IG supports IPV4 and IPV6 traffic.

To enable access to or from the internet for instances in a subnet in a VPC, do following:

* Attach an IG to your VPC.
* Add a route to your subnet’s route table that directs internet-bound traffic to the internet gateway. If a subnet is associated with a route table that has a route to an internet gateway, it’s known as public subnet. Otherwise, private subnet.
* Ensure that instances in your subnet have a globally unique IP address (public IPV4 address, Elastic IP address, or IPV6 address)

**NAT Gateway**

You can use 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.

To create a NAT Gateway, you must specify the public subnet in which the NAT Gateway should reside. You must also specify an Elastic IP address to associate with the NAT gateway when you create it. The Elastic IP cannot be changed after you associate it with the NAT Gateway. After you’ve created a NAT Gateway, you must update the route table associated with one or more private subnet to point internet-bound traffic to the NAT gateway. Each NAT Gateway is created in a specific AZ and implemented with redundancy in that Zone.

**NAT Instance**

It is the historical way for resources in private subnet to access internet. It required a lot of manual configurations.

Features:

* It supports Port Forwarding
* Security Groups can be associated with it.
* It can be used as a bastion host.

**NACL**

Stateless, can have both deny and allow (SG are stateful, can only add allow rules).

NACL process rules in order from the lowest numbered rule to the highest until they reach an allow or deny.

**Site to Site VPN (Also called as IPSec VPN Connection)**

Site to Site VPN will not work unless you implement “Route Propagation” for VPG in route table that is associated with your subnet.

**Virtual Private Gateway** – A virtual private gateway is the VPN endpoint on the Amazon side of your Site-to-Site VPN connection that can be attached to a single VPC.

**Customer Gateway** - An AWS resource which provides information to AWS about your customer gateway device.

**Customer Gateway Device** - A physical device or software application on your side of the Site-to-Site VPN connection.

**Direct Connect**

Provides a dedicated private connection from a remote network to your VPC. Dedicated connection must be setup between your data center and AWS Direct Connect location.

Also, you need to setup Virtual Private Gateway on your VPC. So now you access public resources (s3) and private (EC2) on same connection.

Direct connect often take more than a month to establish a new connection.

Connection types:

* Dedicated Connections: 1 Gbps and 10 Gbps capacity. Physical ethernet port dedicated to a customer. Request made to AWS first, then completed by AWS Direct Connect Partners.
* Hosted Connections: 50 Mbps, 500 Mbps, to 10 Gbps. Connection request are made via AWS Direct Connect Partners. Capacity can be added or removed on demand. 1,2,5, 10 Gbps available at select AWS Direct Connect Partners.

**Direct Connect – Encryption**

Data in transit is not encrypted but is private. AWS Direct Connect + VPN provides an IPsec-encrypted private connection.

**Direct Connect – Resiliency**

High Resiliency for Critical Workloads – Connects to 2 AWS Direct Connect Locations

Maximum Resiliency for Critical Workloads – Connects to 2 AWS Direct Connect Locations with 2 connections per location.

**Direct Connect Gateway** – If you want to connect with 2 or more VPC in many different regions (same account)

**AWS Private Link (VPC Endpoint Services)**

Let’s say you want to expose a service in your VPC to other VPC:

Option 1: Make It public

Goes through public www. Tough to manage access

Option 2: VPC peering

Must create many peering relations. Also opens the whole network even though not required.

Solution: AWS PrivateLink, which powers VPC Endpoint.

It is the most secure & scalable way to expose a service to 1000s of VPC (own or other account).

It does not require VPC peering, Internet gateway, NAT or route table.

Working: Need to have a Network/Gateway load balancer on Application Service side and ENI on Consumer Application VPC side. And this ENI and Load balancer are a part of AWS PrivateLink.

**VPC Sharing (Part of Resource Access Manager)**

It allows multiple AWS accounts to create their application resources, such as Amazon EC2, RDS etc into a shared, centrally managed VPC. In this model, the account that owns the VPC (owner) shares one or more subnets with other accounts (participants) that belong to same organization from AWS Org.

After a subnet is shared, the participants can view, create, modify, and delete their application resources in the subnets shared with them. Participants cannot view, modify or delete resources that belong to other participants or the VPC owners.

**VPC Console Wizard**

It supports following configs:

* VPC with single public subnet.
* VPC with public and private subnet.
* VPC with public and private subnet and AWS Site to Site VPN access.
* VPC with private subnet only and Site to Site VPN access

**VPC CloudHub**

If you have multiple AWS Site-to-Site VPN connection, you can provide secure communication between these sites using AWS VPN CloudHub. This enables your remote sites to communicate with each other, and not just with VPC. Low cost hub and spoke model.

**Notes:**

* Private access to public services such as Amazon S3 can be achieved by creating a VPC endpoint in the VPC. For S3 this would be a gateway endpoint. The bucket policy can then be configured to restrict access to the S3 endpoint only which will ensure that only services originating from the VPC will be granted access.
* In NACL, If you accept traffic from the internet, then you also must establish a route through an internet gateway. If you accept traffic over VPN or AWS Direct Connect, then you must establish a route through a virtual private gateway.
* Sharing a VPC Subnet == Resource Access Manager

**Kinesis Data Stream**

It is to capture, process, and store data stream.

It can maintain the order of incoming data.

There are producers here, who need either SDK, KPL (Kinesis Producer Library), or Kinesis Agent to send data to KDS.

KDS has a stream which has configurable number of shards. The record which is sent to a shard is made of a Partition Key and data blob (up to 1 MB).

Each shard can receive either 1 MB/sec or 1000 msg/sec per shard from producer.

Kinesis Consumers, in addition to Partition ID and data blob, receive a sequence no.

Consumer should have SDK, or KCL (Kinesis Consumer Library) or it could be any managed service like Lambda, KDF, KDA

2 modes for consumers to receive data:

* Shared Consumer – 2 MB/sec per shard for all consumers
* Enhanced Consumer – 2 MB/sec per shard per consumer

Data that share same partition ID goes to same shard

**Kinesis Data Firehose**

It is a fully managed service to load data stream into AWS data stores.

It receives data from KDS (majorly), Cloudwatch Events, AWS IoT, Kinesis Agent, Client or App.

Receives a record of 1 MB.

You can do data transformation using Lambda.

They KDF do batch writes (so it’s near real time, not exact real time – 60 seconds).

Destinations:

* AWS Destinations
  + Amazon S3
  + Amazon Redshift (COPY through S3)
  + Amazon ElasticSearch
* 3rd Party Partner Destinations
  + Splunk
  + Mongo DB
* Custom Endpoint
  + HTTP Endpoint

Firehose cannot directly write into a DynamoDB table.

Pay for data going through firehose, so no provisioning required.

**Kinesis Data Analytics**

To analyze data stream using SQL and Apache Flink.

**Kinesis Video Stream**

To capture, process and store video stream

**AWS Step Functions**

AWS Step Functions lets you coordinate and orchestrate multiple AWS services such as AWS Lambda and AWS Glue into serverless workflows. Workflows are made up of a series of steps, with the output of one step acting as input into the next. A Step Function automatically triggers and tracks each step, and retries when there are errors, so your application executes in order and as expected.

It represents work flow as JSON state machine.

**AWS DMS**

Larger data migrations with AWS DMS can include many terabytes of information. This process can be cumbersome due to network bandwidth limits or just the sheer amount of data. AWS DMS can use Snowball Edge and Amazon S3 to migrate large databases more quickly than by other methods.

When you're using an Edge device, the data migration process has the following stages:

1. You use the AWS Schema Conversion Tool (AWS SCT) to extract the data locally and move it to an Edge device.

2. You ship the Edge device or devices back to AWS.

3. After AWS receives your shipment, the Edge device automatically loads its data into an Amazon S3 bucket.

4. AWS DMS takes the files and migrates the data to the target data store. If you are using change data capture (CDC), those updates are written to the Amazon S3 bucket and then applied to the target data store.

You must create EC2 which runs DMS.

**Notes**

* You can copy an Amazon Machine Image (AMI) within or across AWS Regions. You can copy both Amazon EBS-backed AMIs and instance-store-backed AMIs. You can copy AMIs with encrypted snapshots and also change encryption status during the copy process. You can copy AMIs that are shared with you.
* To have a highly available app, configure an ALB in front of an ASG to deploy EC2 instances on multi-AZ.
* If you are using an EC2 and Amazon RDS database instance, the best way to store the DB username and password is – Create an IAM Role with permissions to access the DB and attach this role to EC2. Don’t store the credentials in disk. The instance will then obtain temporary security credentials from AWS STS which is much more secure.
* IAM Principals = Either IAM Users or IAM Roles
* Global Accelerator is a good fit for non-HTTP use cases, such as gaming (UDP), IoT (MQTT), or Voice over IP, as well as for HTTP use cases that specifically require static IP addresses or deterministic, fast regional failover.
* You have multiple AWS accounts within a single AWS Region managed by AWS Organizations and you would like to ensure all EC2 instances in all these accounts can communicate privately. Which of the following solutions provides the capability at the CHEAPEST cost?

Ans - Create a VPC in an account and share one or more of its subnets with the other accounts using Resource Access Manager

* Private IP ranges are: 192.168.0.0 - 192.168.255.255 (65,536 IP addresses) 172.16.0.0 - 172.31.255.255 (1,048,576 IP addresses) 10.0.0.0 - 10.255.255.255 (16,777,216 IP addresses).
* AWS supports six types of policies: identity-based policies, resource-based policies, permissions boundaries, Organizations SCPs, ACLs, and session policies.
* If the IAM role that you create for the Lambda function is in the same AWS account as the bucket, then you don't need to grant Amazon S3 permissions on both the IAM role and the bucket policy. Instead, you can grant the permissions on the IAM role and then verify that the bucket policy doesn't explicitly deny access to the Lambda function role. If the IAM role and the bucket are in different accounts, then you need to grant Amazon S3 permissions on both the IAM role and the bucket policy. Therefore, this is the right way of giving access to AWS Lambda for the given use-case.
* Kinesis Agent is a stand-alone Java software application that offers an easy way to collect and send data to Kinesis Data Streams or Kinesis Firehose.
* SQS long polling is expensive compared to short polling. By default, short polling is enabled.
* To call IAM API, either use Query API or Access key ID & Secret Access Key.
* Prefer IAM policy over Bucket Policy (especially for S3).













