AWS NOTES- Part B

Contents

Relational Database Service 1
Simple Storage Service (S3) 9
Route 53 19
CloudFront 23
CloudTrail 27
Encryption – KMS / (HSM) Hardware Security Module 28
SNS (Simple Notification Service) 29
Beanstalk 30

Relational Database Service

- There are two primary purpose of a databases
  - OLTP (online transaction processing).
  - OLAP (online analytical processing).
- There are two types of databases
  - **Relational Database (RDMS)** [structure database]
  - **Non-Relational Database (Non-RDMS)** [un-structure database].
- Different types of non-relational databases
  - **Columnar Databases:** These are optimized for reading columnar data, Example: Apache HBase, Apache Cassandra]
  - **Document Databases** [These are used for storing non-structural document base objects in SON or XML data. These types of databases are fit for storing varying structure data and where high read/write performance is desired at lower cost. Common use cases are: gamming application / web application / mobile application / IoT applications / Ad Tech etc.]
  - **In-Memory databases key-value store**: These are optimized to store key-value which demands heavy workload (frequent reads). These types of databases are fit for heavy workload application like – recommendation engine. ()
  - **Graph Databases:** A graph database is a database that uses graph structures for semantic queries with nodes, edges, and properties to represent and store data. A key concept of the system is the graph (or edge or relationship). The graph relates the data items in the store to a collection of nodes and edges, the edges representing the relationships between the nodes. The relationships allow data in the store to be linked together directly and, in many cases, retrieved with one operation. Graph databases hold the relationships between data as a priority. Querying relationships within a graph database is fast because they are perpetually stored within the database itself. Relationships can be intuitively visualized using graph databases, making them useful for heavily inter-connected data. (Neo4J and Amazon Neptune are example of graph databases.)
- AWS manage following service for RDS instances
  - Hosting of DB instances

- Security & patching of instances
- Automated backup of the DB instances
- Software updates of DB instances
- Synchronous replication of DB instances across multiple AZ (*this needs to be explicitly configured on the DB instances by enabling multi-AZ feature*)
- Provide ability to create read replicas to optimize performance of heavy read operation databases.

- AWS doesn't manage following service for RDS instances
  - Managing DB settings
  - Building DB schema for non-relational databases
  - DB Performance tuning
- List of AWS managed RDS databases
  - MS SQL
  - Oracle DB
  - PostgresSQL
  - MariaDB
  - AWS Aurora
  - MySQL
- Licensing model supported in AWS
  - BYOL (bring your own licensing model)
  - Licenses included
  - **Note: Max of 40 db instances can be created per AWS account.**
    - **10 out of 40 can be of Oracle or MS SQL database type, when opted for Licenses included model**
    - **All 40 DB instances can be of type Oracle or MS SQL database type if BYOL model is opted instead of Licenses included model.**
  - **Amazon RDS uses EBS volumes as databases block store**
    - **General Purpose RDS storage** is mainly suited for DB workloads for moderate I/O requirements.
    - **Provision IOPS RDS storage** is mainly use for high performance OLTP workload.
    - **Magnetic Disk RDS storage** is ONLY use for smaller databases.
  - Maximum capacity of the RDS storage capacity
    - **Up to 4 TB for MS SQL server**
    - **Up to 6 TB for other RDS engines.**

      MS SQL storage limit is because of the limitation of striped storage that can be attached to the Microsoft windows server.
  - Multi AZ configuration
    - This can be enabled when db instance is getting created.
    - When enabled, an instance of the DB will be created on a separate AZ within the same region.
    - The secondary (standby) instances will be automatically synchronized with the primary (main) instance.
    - NO read/write can be done on the secondary (standby) instances while primary db instance is still active.
    - **AWS does not provide option to select AZ when enabling multi-az option in the database. However, once created one can view the standby db instance AZ.**
    - Depending upon the instance type, it can take 1 to few minutes for the DB instance to failover.

- AWS recommended **provision-IOPS for multi-AZ base set up.**

- In case of Multi-AZ setup, the failover will be triggered on the following scenario:

  Automated failover trigger
  - Loss of Primary database AZ or Primary DB instance failure.
  - Loss of Primary instance network failure
  - Loss of Primary instance compute failure
  - Loss of Primary instances storage failure
  - **The primary DB instance change**
  - Patching of the Primary db instance OS.

  Manual failover trigger
  - This can only happen when; primary instance is rebooted with "*reboot with failover*" option.
  - The following are the reason when a primary db instance needs to be rebooted:
    - When there is a change in the parameter group.
    - When there is a change in the static DB parameter.
      Note: parameter group is the container for the DB instance configuration, which needs to be change to tune performance.
- When instance is failover to work seamlessly, its recommended to refer the DB instance with its CNAME (endpoint) instead of DB IP address.
- During failure, AWS automatically update the CNAME (endpoint) with the secondary (stand by) db instance IP address.
- On an event of any RDS db instance failure, AWS will send and SNS notification (*this needs to be enabled, not enable by default*).
  - Using *DescribeEvents* API call one can view RDS failure events of PAST 14 days.
  - Using AWS CLI, one can view RDS failure events of PAST 14 days.
  - Using AWS console one can view ONLY RDS failure events of PAST 24 Hours (1 day).
- DB instance OS level changes are done on the STANDBY first then promote the standby to PRIMARY before applying the changes to the STANDBY instance. Following are the OS level changes.
  - OS Patching
  - System Upgrade
  - DB Scaling
- **AWS can restore DB backup up to RPO of 5 min.**
  (Recover Point Objective – time in past till when the instance can be restored).
  (Recover Time Objective – time taken to recover).
- While taking snapshot and backup of an instance there will be a freeze in the I/O operation in case of a stand-alone instance, in case of a multi-AZ setup the snapshot and backup of an db instance will be initiated on the standby instance, to avoid any a freeze in the I/O operation in the primary instances.
- RDS versions can be upgraded to any other supported RDS version
  - User can trigger the upgrade (*modify db instance version*), and the change will take place in the next maintenance window.
  - User can force upgrade the db instance version, and the change will take place immediately.
    Note: During db instance upgrade the db instance will not be accessible for read/write operation for stand-alone as well as multi-AZ setup as both the instances will be simultaneously upgraded.

- **In case of multi-AZ setup use needs to make sure the Application is allowed to communicate to both the primary and secondary (standby) db instance subnets.**
- **RDS backup:**
    - **Automatic backup** (*enable by default, user can disable* if not required)
    - **Manual backup** (need to trigger manually)

Note: When AWS RDS backup is trigger – it doesn't just backup a database, entire db instance is backed up into a snapshot which can later be restored. And store the snapshot into multiple AZ for durability. The db instance snapshots are stored in S3.

Automated Backup	Manual Backup
Can be used for point in time recovery.	Cannot be use for point in time recovery.
Trigger automatically	Trigger Manually
When RDS instance is deleted automated backups are automatically deleted.	When RDS instance is deleted, manual backup still remains in S3 bucket. One needs to manually delete the backup.
Cannot be share directly with other AWS account.	Can be share directly with other AWS account.

There are no charges for enabling automated backup for a db instance, however the storage will be chargeable.

DB instance should be in "*active*" state for automated backup to take place, if the instance is in "*Storage_Full*" state the automated backup will not occur.

- **Using automated backup, it's possible to restore RDS DB instance point-in-time, however same cannot be achieve in manual backup.**
- **RDS automatic backup is taken daily, including DB transaction logs. Using automated backup & DB transaction logs it's possible to restore an DB instance up to 5 min in past point-in-time. One can define backup window.**
- **Note: Once DB is restored, ONLY default DB parameters and Security groups will be automatically restored – custom DB parameters and security group settings needs to be applied.**
- **Retention period** is the timeframe till when the db instance backup is retained.
    - Max retention period = 35 days,
    - Default retention period from AWS console = 7 days (**for aurora db its 1 day**)
    - Default retention period from AWS CLI = 1 day
    - When set retention period =0 automated backup will be disable.
- For MySQL automated backup is supported only for InnoDB storage engine, currently it's not supported for MyISAM storage engine.
- Automated backup cannot be shared within other AWS account. One need to copy the automated backup copy and share it with other AWS account.
- DB instance restore
    - When the DB instance is restored it restore back to default DB parameters, one needs to manually apply the DB parameter changes and also the security group.
    - The restore DB instance will have different endpoint then that of the original DB.
    - When restoring DB instance one can change the storage type.
- DB Subnet Group – collection on subnets where db instance can be launch. When a db instance is launch user have option to select the subnet group and a specific subnet from

the subnet group. This give better control over the NACL and security group configuration. However, one cannot control in which IP address of the specified subnet the DB instance will be launch.

- RDS database encryption
  - When RDS DB instance is created one can enable encryption.
  - For already existing DB instances
    - Create a new encrypted DB instances then migrate existing (unencrypted) DB instances data to it.
    - Restore the existing (unencrypted) DB instances to a new DB instances with encryption enable.
  - When encryption is enabled, all the following will be encrypted
    - All its snapshot
    - Backups
    - Data storage (on DB instance)
    - Read replicas created from the DB instances
- RDS DB instance costing
  - DB instance hours (for partial hours it needs to pay for full hours)
  - Storage (*EBS storage*) per GB/month
  - I/O request/month (this is only for magnetic RDS instance type)
  - Provision I/O request/month (this is only for provision IOPS SSD instance)
  - Internet database transfer
  - Backup storage - automated backup storage equal to the sized of the DB instance size is FREE for the same AZ where the RDS instance is launched. AWS store the backup in multiple AZ for durability, backup stored in AZ other than the AZ where RDS is launched will be chargeable.
  - For Multi-AZ setup additional cost will be applicable
    - RDS instance hours for the standby database
    - Storage for the standby database
    - Additional IOPS per month for synchronization
    - Data transfer between primary and secondary is NOT CHARGEABLE.
- Reserved Instance for RDS instance
  - Reserved instance should have exact match
    - Instance Type
    - Storage
    - DB Engine
    - Standalone or Multi-AZ
  - Reserved instance is region specific, cannot be move from one region to another region. However, they can be move from one AZ to another AZ within a same region.
- **Standby instance should be made in the same region (across multiple AZ), Read replication can be done across region.**
- **The Standby and Primary has same endpoint with different IP addresses, in case of failure the IPs will be swap in DNS.**
- **On an event of an RDS instance failure, RDS will send SNS notification to RDS event category group by RDS service – one needs to subscribe to these event categories.**
- One can also set ColudWatch alarms based on certain metrics (single alarm/single metrics) which can be sent to SNS or take specific actions.

- **Read Replicas:** duplicate RDS instance of the primary db instance, which can be primarily use for scaling heavy read operation. The following DB engine supports read replicas.
    - Read Replicas use cases
        - Catering to read heavy operation
        - For supporting higher I/O demand which cannot be meet by the DB engine specification
        - Supporting read operation in case of a network failure with the primary database engine.
    - **Postgres SQL**, **MariaDB**, **MySQL**, **Aurora DB**
    - MariaDB, MySQL and PostgreSQL one can have read replicas in other region, Aurora does not support multiple region read replicas.
    - One needs to enable automatic backup (*retention period not be equal to zero*)
    - For Postgres SQL, MariaDB, MySQL the max up to 5 replicas can be created for each primary database instances
    - Aurora DB supports up to 15 read replicas
    - MariaDB, MySQL supports writing in the read replicas
    - MariaDB, MySQL supports creating of read replicas of read replicas – max up to 4 replication level can be created.
    - In case of Multi AZ failover – one the primary DB instance failover to standby instance – read replicas automatically points to new primary DB instances.
    - While creating read replicas
        - Read replicas can be created from AWS console / AWS CLI
        - One can select the AZ where read replicas needs to be hosted
        - Read replicas storage & instance type can be different than that of the primary database storage and instance (but it should have minimum storage and compute of the primary database, if primary db instance is scale-out the read replicas should also be scale-out to meet the minimum requirement)
        - Maria DB and MySQL supports read replicas of read replicas, however maximum of 4 level can be created as chain.
        - Chain read replicas will have a greater lag.
    - While deleting read replicas
        - When DB instance is deleted – one needs to manually delete the read replicas.
        - If the primary DB instances is deleted – the read replicas get promoted as standalone primary databases.
    - While reapplication terminated
        - If more than 30 consecutive days replication is turn off (manually or due to error), AWS delete the read replicas automatically.
    - When Primary DB instance is lost with read replicas
        - Read replicas can be promoted as primary database – to reduce RPO and RTO.
        - Read replica can be promoted to standalone DB instances within a single AZ.
        - Once a read replica is promoted to primary db, the following parameters it inherited from the primary db
            - Backup Retention Period

        - Back window
        - DB Parameter Group
    - Once a read replica is promoted to primary db the other read replicas of the existing primary db continues to work as is without any changes.
  - InnoDB DB transaction storage engine supported for RDS replication. Non-transactional storage engine like MYISM may prevent read replicas to work as intended.
- RDS scaling
  - While scaling the RDS instance there will not be any downtime except
    - When upgrading the db engine (applicable even to multi AZ setup)
    - When changing db parameters (applicable only to standalone db instances NOT applicable to multi AZ setup).
  - RDS storage can only be increase
  - RDS storage type can be change (except MySQL)
  - RDS scaling can be set as apply immediately OR apply during change window.
- RDS Pricing
  - If an RDS engine is launched, regardless whether it's used or not it will be chargeable to the customer.
  - Licensing option in RDS are BYOL and License included.

- Amazon Arora DB:
  - Managed DB service from AWS – which is compatible with PostgresSQL and MYSQL
  - It can grow up to 64 tera bite with minimum size of 10 GB.
  - Aurora DB cluster consist of multiple DB instances – group under a cluster and can be access through cluster endpoint or through instance endpoints.
  - The data are stored under cluster volume which are stored under multiple AZ.
  - Each Aurora DB cluster has one primary db instances – which can be used for performing read & write operation. Aurora cluster can also have up to 15 read replicas, which offload read workload from the primary db instances & if the primary instance fails it acts as failover.
  - Aurora DB endpoints:
    - **Cluster Endpoints:** each Aurora DB has one single primary instance and one cluster endpoint – this endpoint can be used for both read and write operation.
    - **Reader Endpoints:** This is the read-only endpoint which load-balanced incoming request (query) and direct it to a specific Read replica. This endpoint can't be used for write operation.
    - **Custom Endpoint:** This helps in forwarding incoming request to a specific instance & define what kind of operation can be allowed on that instance.
    - **Instance Endpoints:** Individual end point of the Aurora DB instance , these are typically used for diagnostic capacity and performance issue on a specific DB instance.
  - Aurora Redundancy:
    - Aurora DB stores cluster volume in multiple AZ.
    - Aurora DB detects disk failure and can repairs the segment on their own.
    - Post startup after failure, it copies the buffer pool to make the data readily available.
    - It can recover instantaneously from its failure.
  - Aurora Fault Tolerant:

    - It synchronously synchronized primary db instance with another read replicas instance.
    - When primary instance fails it can do one of the following
        - Either create a new primary instance
        - Or Promote a read replica as primary instance.
    - Data blocks are continuously scan for error and repairs automatically.
    - Aurora MYSQL supports cross region replication either by physical or logical replication.
    - Aurora PostgresSQL doesn't support cross region replication
- Aurora DB configuration
    - Based on the DB instance classes Aurora DB performance are determined, there are two types of instance class available for Aurora DB instances
        - **Memory Optimized** – Optimized for memory intensive operation.
        - **Burstable performance** – were instance are configured to burst full CPU usages.
    - **Aurora DB Serverless configuration** which is an on demand autoscaling configuration for the Aurora DB compatible with MySQL and PostgresDB. This can startup, shutdown and scale as per the need. One need to define
        - Maximum Aurora Capacity unit: Maximum limit.
        - Minimum Aurora Capacity unit: Minimum limit.
        - Pause after inactivity: the amount of time with no DB interaction the DB instance can be scaled to ZERO.
- **Aurora DB Reboot** – NO failover occurs: When a primary instance is rebooted, its read replicas are also rebooted automatically.
- **Delete Protection flag can be enabled to avoid accidental delete**. An Aurora DB instance can't be deleted if BOTH the conditions are true.
    - If the Aurora DB cluster is NOT a read replica to another DB cluster
    - There is ONLY one DB instance.
- Aurora DB monitoring
    - **Subscriber to Amazon RDS Events** to know the changes occurred in DB instance, DB cluster, DB parameter group, DB cluster snapshot.
    - **RDS enhance monitoring:** Look at the metrics at real time for the operating system.
    - **RDS Performance insight:** Monitor your Amazon DB load to analyzed and troubleshoot database performance issue.
    - **CloudWatch Metrics – alarms, metrics and logs**
- Aurora DB Security
    - **Use IAM to control access**
    - **Security Group can be used to control which devices / EC2 instances can connect to Aurora DB cluster endpoints.**
    - **Configure SSL and TSL to connect to Cluster endpoint**
    - **User RDS encryption to secure RDS instance and the snapshot at rest.**

Aurora DB for PostgreSQL	Aurora DB for MySQL
Push button compute scaling	Push button compute scaling
Storage Autoscaling	Storage Autoscaling
Low latency read replicas	Low latency read replicas

Custom Database endpoints	Custom database endpoints
Scaling • Instance scaling – modifying instance class for better performance • Read scaling – by adding read replicas	Scaling • Instance scaling – modifying instance class for better performance • Read scaling – by adding read replicas
Aurora DB for PostgreSQL supports logical data replication, data changes in PostgreSQL Aurora DB can be replicated to the other DB using native PostgreSQL slots or other data migration tools	
	Global database is currently available for Aurora DB MySQL compatible version.

Simple Storage Service (S3)

- **Block Storage Vs Object Storage**

Block Storage	Object Storage
In block storage, data will be divided into smaller equal sized blocks and then stored within a block store like EBS (Elastic Block Store) which can be referred by an index by the block store.	In object store data are stored as whole. They are not divided into any smaller unit. Object up to 5 Tb can be stored as a whole. They can be referred by – date/object meta-data/filename/version number (optional).
Example of a Block store: EBS volume	Example of an Object store: S3
What can be store in block storage: DB transaction logs, software installed folder	What can be object in block storage: File, photo, video, music file, EBS volume snapshots, DB snapshots, AWS cloudTrail logs
ONLY the EC2 instance where its mounted to.	Accessible from anywhere in the world.

- **Data consistency model for distributed storage architecture**
  - **Immediate (Strong) Consistency model** – when a data that written to multiple system, and read immediately if the user gets same response from all the system then, it's called as immediate consistency model.
  - **Eventual Consistency model** – when a data that is written to multiple system and read immediately if the user doesn't get the same response from all the system then, it's called eventual consistency model. Its preferred for
    - High Stability
    - Higher Availability
    - High Data Durability
    - Lower cost
- In case of S3 when a new object is first written (PUT) into S3 storage it will ensure **read-after-write consistency model** (immediate consistency model).
- In case of updating or deleting an existing object (override PUT or DELETE) into S3 storage then S3 will ensure eventual consistency model.
- S3 durability is 11 9's (Chances of losing 1 in 100000000 percentage)
- Minimum size of an object that can be stored in an S3 bucket is 0 Bytes.

- Maximum size of an object that can be stored in an S3 bucket is 5TB.
- S3 bucket is non-hierarchical, i.e. one can't have another S3 bucket within S3 bucket however once can create folders within S3 bucket from web console.
- S3 bucket is a region specific – its store uploaded object in multiple location within a single region.
- **S3 bucket ownership can't be transfer from AWS account to another.**
- By default, object uploaded to a S3 bucket is private ONLY bucket owner can access the objects.
- If the object is uploaded by and IAM user, still the object is owned by the account owner NOT the IAM user.
- S3-Sub Resources
  - **Life Cycle policies:** Policies to decide object lifecycle
  - **Website:** To hold configuration to host static websites
  - **Versioning:** Maintained different version of updated objects
  - **Access Control List:** Bucket access
  - **Bucket Polices:** Bucket object access
  - **Cross region replication:** Replication across different region.
  - **Torrent:** use S3 object to be download using bit-torrent software.
  - **CORS:** Cross Origin Resource Sharing.
  - **Logging:** Bucket access logs.
- There are two types of s3 URLs
  - **Virtual HOST Type** e.g. https://[*bucket-name*]. s3-[*aws-region*]. amazonaws.com
  - **Path Type** e.g. https://s3-[*aws-region*]. amazonaws.com/[*bucket-name*] for US East (N. Virginia) Region alone the endpoint will be `http://s3.amazonaws.com/[bucket-name]`

    Starting March 20, 2019, bucket created in the region will not be reached through https://[*bucket-name*].s3.amazonaws.com.
- Data transfer charges within EC2 instances and S3 bucket within a same region doesn't incur any charges.
- S3 cross account access for object upload
  - Under cross account access, account from which the object was uploaded owns the object not the bucker owner.
  - Bucket owner still pays for the object storage, uploaded by another account.
  - Bucket owner can deny access to object which are NOT owned by them.
  - Bucket owner can delete/archive the object which are NOT owned by them.
- Whom can be grated to access S3 buckets?
  - IAM users
  - Another AWS account.
  - Authenticated Users – who are not AWS users.
- S3 access policies
  - A S3 bucket can have a Bucket Policy or bucket ACL/object ACL
  - A S3 object can have
    - Resource Based Policies
    - User Policies
- S3 Access Level Policy
  - **Account owner** – AWS resource that created the bucket. A bucket owner can delete an object regardless of who owns the object.

        - **Resource owner** – AWS account that creates the object. Object owner owns the object. If an IAM user uploads an object, it's the AWS account that has created the IAM user owns the object not the IAM user.
- Access Polices attached to S3 resources
    - Resource base policies – policies that are attached to a resource (bucket and objects).
    - User base polices – polices that are attached to a user who can access the resources.
- Resource base policies
    - ACL
        - Each bucket & object can have ACL attached to it.
        - ACL is a list of grants that identify grantees to access specific S3 resources.
        - ACL can be use only to give basic read/write grant to attached S3 resources.
    - Bucket policies
        - Bucket polices can be created to grant access to a specific s3 resources.
        - Then the AWS owner can only provide deny access to the objects that are not created by them but they can't grant read access to those objects.
        - In many case Bucket policy replace ACL policy.
- User base polices
    - Can grant access to a bucket or the any particular object to IAM users
    - Can create user, group and role which can be attached to an IAM user to grant access to S3 resources.
    - Anonymous access can't be granted using user base policies as it needs a specific user / group / role to get it attached to.
- Evaluation of an access for S3 resources
    - User context
        - S3 checks if given user has permission from the AWS account owner for IAM users.
        - S3 checks if user has access to the requested object.
    - Bucket context
        - S3 check if the bucket owner allows the use to perform the request operation on the bucket

Note

- If bucket and the object is the same AWS account then, IAM user can be granted access either by resource policy or by user policy.
- If the bucket owner and resources owner is the same – then bucket policy will be evaluated for the access.
- If the bucket and resource owner are different then -owner must use object ACL to grant permission.

- S3 ACL (bucket ACL and user ACL) can provide finite set of control – bucket policy and user policy are the one that provides granular control.
- Default ACL gives full control to the AWS account owner over the created resource (bucket or object).
- Using ACL, AWS account owner can grant access to the grantee AWS account, later AWS grantee account owner can delegate that access to its IAM users. Using ACL s3 bucket owner can't directly grant access to the grantee's IAM user.
- List of access that can be granted from ACL

Permission	When granted for a bucket	When granted for an object
READ	Allows grantees to list all objects within the bucket	Allows grantees to read the object and its metadata
WRITE	Allows grantees to write, overwrite, delete any object in the bucket	NOT applicable
READ_APC	Allows grantees to read bucket ACL	Allows grantees to read object ACL
WRITE_APC	Allows grantees to write ALC for the attached bucket	Allows grantees to write ACL for the attached object ONLY.
FULL_CONTROL	Provide grantees READ, WRITE, READ_APC and WRITE_APC access on the bucket.	Provide grantees READ, WRITE, READ_APC and WRITE_APC access on the object.

- Use cases for Object ACL and bucket ACL
    - Object ACL
        - Object ACL are used to grant access to an object to another AWS account who is not the bucket owner. (*However, bucket owner can still delete/archive the object*).
        - Bucket Policy can of MAX 20KB size – therefore to grant granular access at the object level it required to use object ACL. (*Object ACL can have up to 100 access policies in it, and only one object ACL is permitted per object.)*
    - Bucket ACL
        - AWS recommended to use bucket ACL to grant access to the S3 log delivery group.
        - Bucket ACL can be used to grant bucket policy to another AWS account
- When to use bucket policy or user policy?
    - Bucket policy (can provide partial range of permissions)
        - When AWS account owns the bucket and wants to grant access to the bucket to its users
        - When AWS account own the bucket and object and it want to grant access to the object to its users.
        - When AWS account wants to grant an access to another AWS account to access its buckets.
        - For granting access to S3 log delivery group.
    - User policy (can provide full range of permission)
        - User policy can be used to grant full range of s3 permission.
        - To grant access to the IAM users of same AWS account.
        - To delegate the grant the it receives from another AWS account that own the bucket/object.

Note: IAM user must have access permission from the parent account where it belongs to & also from the resource owner.

- Parent owner can share its access to any resource/bucket using user policy.
- Resource owner can share its access to other account users using bucket policy OR share access to other account using bucket policy, bucket ACL, and object ACL which account later can share with its users.

AWS account shares its access to another AWS account, that account CAN'T further share the access to the other AWS account.

- **S3 versioning**:
  - Once enable cannot be disable, however it can be suspended at any given time.
  - By default, GET will return the latest from the S3 bucket
  - Delete Marker - Once a versioned object is deleted it doesn't get deleted, only a delete marker gets attached to the object. Later one can delete the delete marker and make the object re-available.
  - Once versioning is enabled, it will charge for all the version that are hosted within the S3 bucket.
  - Second level security for deleting versioning delete can be enable using MFA (Multi Factor Delete) delete.
- **Copying and Updating Object:**
  - Multi Part upload – for object size large than 100MB its recommended to use multi part upload feature instead of single upload to improve performance (*speed up the upload process*). File with size 5MB to 5TB can be uploaded using multi part upload feature. [*file size more than 5GB, multi part upload is the ONLY option*.]
  - Transferring files within region does not incur transfer charges.
  - While copying storage class and encryption status can be changed.
  - There are two types of metadata – system define metadata & user define metadata, some of these metadata can be changed or added new when copying the files
  - When s3 upload is successful, it returns a HTTP 200 OK response message back.
  - When s3 uploads is successful for a file which has requested for Server-side encryption SSE using customer provide key file along with HTTP 200 OK response it also returns encryption algorithm & the MD5 encryption key used specified during upload.
- **Storage Classes**

Real time storage class

	Standard	**S3-IA**	**S3-Sigle Zone - IA**	**S3-Reduce Redundancy Storage**
Sustainability	Design to sustain data loss in two facility.	Design to sustain data loss more than 1 facility.	Design to sustain data loss in 1 facility.	Design to sustain data loss in 1 facility.
Availability	99.99%	**99.9%**	**99.95**	99.99%
Durability	99.99999999999 11 9's	99.999999999 9 9's	99.999999999 9 9's	**99.99%**
Minimum Size	0 KB – 5TB	Greater than 128 KB		
Usages	critical data	Old less frequently use data.	Non-critical data	Non-critical data / temp for downloading archive data
Retention Period	Minimum of 30 days charge is applicable.	Minimum of 30 days charge is applicable.	Minimum of 30 days charge is applicable.	

Archival Storage:

	Glacier
Sustainability	Store within a three physical availability zone within a same AWS region.
Availability	NO SLA
Durability	99.99999999999 11 9's
Minimum Size	1 Byte – 40TB
Usages	Archiving data
Retention Period	Minimum of 90 days charge is applicable.

- Once requested – glacier data will be made available in 3-4 hours, the data will be retrieved and upload to a S3-RRS storage class.
- If the data is more than 4GB, then it needs to be uploaded as multi part upload. Any data more than 100 MB can be uploaded as multi part upload.
- Data can't be upload to Glacier storage class from AWS console one can ONLY use command line interface, AWS API or AWS SDK data can be upload to Glacier storage class.
- When a lifecycle policy transitions a data from a S3 storage to Glacier it adds 32KB of data for indexing and achievable metadata. This overhead increase to 40KB (extra 8KB) if the S3 is use to load the data to glacier. [*In order to keep the overhead size small, it's advisable to bundle up the data instead of uploading it in small chunks. While bundling up the data it always required to uses right compression technique where one can easily download individual files from the uploaded bundle.]*
- Glacier doesn't allow any metadata to the uploaded data except the archive description. So, it's important to maintain a client-side repository about the archive meta data.
- Once a data is uploaded into glacier, then it cannot be changed.
- Glacier data retrieval
    - **Expedite retrieval:** fastest retrieval time 1-5 minutes/ costliest among other retrieval process.
    - **Standard retrieval:** 3-4 hours of retrieval time / up to 10 GB of data retrieval is free.
    - **Bulk retrieval:** 5-12 hours of retrieval time/ Cheapest among other retrieval process.
    - **Part retrieval:** Part of archive retrieval.
- Glacier data upload is a synchronous process- S3 will send success message back only when data is store across multiple location within Glacier.

- Glacier data download is an asynchronous process – once the data is successfully retrieved then SNS notification will be sent.
- Data retrieved from Glacier is copied to RRS – default retention period is 24 Hrs. but this can be changed depending upon the requirement (it can be extended or shorten)
- **RetrievalByteRange**: Instead of retrieving the complete archive data, one can request to download a portion of the data using **RetrievalByteRange** filed which needs to be set in the HTTP header while sending the request it should be in multiple of 100MB. In order to retrieve the correct ByteRange its advisable to maintain a repository outside glacier about the archive data.
- **Glacier Costing:**
    - No cost of moving data from EC2 instance to Glacier within a same availability zone.
    - Minimum charges of storing a data is 90 days.
    - While retrieving a data from glacier – along with the cost of data retrieval, temporary storage cost of RRS storage class will be applicable till the time retrieve data will be available.
- Lifecycle policies
    - Lifecycle policies can be applicable to entire objects OR object with specific tag or prefix.
    - There are two types of lifecycle policies
        - **Transition Action**: After expire of the define period it moves the object from one storage class to other storage class.
        - **Expire Action**: After expire of the define period it deletes the object from s3 storage class.
    - Lifecycle policies can't be used to move an object from glacier to S3-Standard or S3-IA storage class.
- S3 encryption
    - Client-Side encryption: Client encrypts the data before uploading the same into S3 storage.
    - Server-Side encryption: Client uploads non-encrypted data, S3 service encrypts the data before storing it into S3 storage – while retrieving the data S3 service decrypts the data and send non-encrypted data back the client. Depending upon who manages the encryption key there are three type of SSE encryption available
        - S3-SSE: In this type of encryption the S3 service manages its own key.
        - S3-SSE-KMS: In this type of encryption the Key Management Service manages the key on behalf of S3
        - S3-SSE-Client: In this type of encryption the Key is managed by the client, and provide to S3 services for encryption. S3 doesn't stores the client key, in case the client losses the encryption key the data is lost.
    - S3-SSE how it works?
        - S3 service request for the data key to AWS KMS service
        - KMS service generate the data-key and encrypts it with master key and send the encrypted data key and the plain-text data key back to the S3 service.
        - S3 service encrypts the data using plain-text data key, and store the encrypted data key for future reference. Once data is encrypted it deletes the plain-text data key.

        - For decryption, S3 sends the encrypted data key to the KMS service, which decrypts the encrypted data key using its master key and send back the plain-text data key back to the S3.
        - S3 use the plain-text data key to decrypts the data and send it back to the client.
        - S3 users AES (Advance Encryption Standard) 256-bit encryption for performing Server-Side Encryption.
        - S3 rotates encryption keys periodically.
        - There is NO extra changes for S3 server-side encryption.
    - S3-SSE-Client Provide key
        - Client provide the encryption key to S3 service while uploading the data, S3 service user 256-AES encryption for encrypting the data.
        - Once the data is encrypted the client key is deleted.
        - While retrieving the data, client needs to provide the same encryption key which it has provided during uploading. If they key matches, S3 decrypts the data and make it available to the client.
    - S3-KMS
        - First time, when a data is uploaded into S3-KMS it generates a Default Customer Master Key for that particular region. This key can be used for encrypting the data key which will be used for encrypting the actual object.
        - Its advisable to create own customer master key instead of using Default Customer Master Key, as it gives more flexibility to create, rotate, disable, define access control and audit the encryption key used for encrypting data.
    - S3 bucket policy can be create in order to apply SSE for all uploaded object for that particular bucket.
    - When uploading a request, one need to include **x-amz-server-side-encryption header** to enable SSE.
- Static-website hosting
    - S3 Static website hosting feature can be use to host static content from S3 bucket.
    - S3 static website hosting URL looks like http://<bucket-name>.S3-website-<AWS-Region>.amazonaws.com
    - S3 static website URL doesn't support SSL connection.
    - S3 static website URL can be routed to custom domain URL using route53 CNAME.
    - S3 static website can route an incoming request based on the prefixes or the object name.
    - S3 hosted static website can redirect a request to the whole domain or pages within the domain or to a specific object.
    - There is no need to add ELB or autoscaling group to scale out the website, AWS s3 automatically scales as per the demand.
    - There is no additional cost of hosting static website.
    - S3 static website hosting doesn't allows requester pay request.

    Difference between REST API and Static website

	Rest API	Static Website
Access control	Support both public and private content	Supports only public content
Error Message handling	Return back the HTTP base error code	Return back configured HTML error page.
Redirection Support	Not available	Can be redirect to object or to a bucket
Request Support	Support all bucket operation	Supports ONLY GET and HEAD HTTP request
Response to GET and HEAD request	Return list of Object key hosted in the bucket	Return index.html page
SSL Support	Available	Not available

- Pre-Signed S3 URL: For sharing temporary access of a specific resources with the users how don't AWS account, one can use S3 URLs.
- Pre-Signed S3 URL can be generated by using SDKs, AWS explorer for visual studio, while creating a pre signed URL its mandatory to define an expiration date.
- Pre-Signed S3 URL can be use for both uploading and downloading object into S3 buckets.
- Cross Region Replication: This can be enabled at the bucket level to replicate the bucket object in different region. This is an automatic & asynchronous process managed by AWS on your behalf.
- Cross Region Replication can be made enable to the entire bucket or to object with specific key-name or object with specific prefix.
- Cross Region Replication can be configured with life cycle management rule
- Cross Region Replication is always 1-to-1 replication.
- During the replication the storage class can be changed, by default the storage class will be applied.
- Cross Region Replication happens over SSL channel
- To enable Cross Region Replication, it required to enable versioning.
- To enable Cross Region Replication, the source bucket owner should have access to the object & the object ACL in case bucket owner and the object owner are different. Source bucket owner should also have access to replicate object at the destination bucket. This can be granted by the destination bucket owner to the source bucket owner using bucket policy.
- During any change in the object, object ACL, object meta-data or during uploading / deleting object in the source bucket Cross Region Replication will be triggered.
- Any object that are existing in the bucket before Cross Region Replication is enabled will not be replicated, however any update to the existing object will be replicated.
- If an object is deleted from the source bucket Cross Region Replication will also attached the delete marker to the replicated bucket.
- If an object with specific version is deleted from the source bucket then the Cross-Region Replication WILL NOT add the delete marker to the replicated bucket (it will only delete the object version from the source bucket), this is done to prevent the malicious delete behavior.
- Objects that are encrypted using S3-SSE KMS and S3-SSR Customer Key WILL NOT be replicated as AWS will not have the encryption key to replicate the object at the destination bucket.

- Any sub-resources added to the bucket like life cycle policy or static website hosting will not be replicated to the destination bucket.
- Any object deleted by the life cycle policy WILL NOT be deleted at the destination bucket (but one can manually configure the same lifecycle policy at the destination bucket to overcome this).
- For Cross Region Replication following will be charged will be additional to the cost of the source bucket.
    - Uploading cost of the object at the destination bucket
    - Data transfer charges across region
    - Storage cost for the destination bucket
- Cross Region Origin: This can be enabled for the static website hosting to refer content from another domain (s3-bucket), by default this will not be enabled.
- Transfer Accelerator: TO improve upload performance, instead of uploading an object to the S3 region which can be far off from the uploader, use can upload the content into a nearby edge location from there the object will be transferred to the bucket over AWS infrastructure. [this doesn't guarantee performance enhancement, if performance is enhanced then this service is chargeable.]
- By default, transfer accelerator is not enabled, bucket owner can turn on this feature once turn-on it can take up to 30 minutes to enable transfer accelerator. Once enabled user can upload their content to transfer accelerator URL instead of S3 URL. https://<bucket-name>.S3-accelarate.amazonaws.com
- S3 Performance enhancements:
    - Upload enhancements: S3 internally maintain a list of indexes of the object that are store in S3 bucket, object with similar name are stored in same or nearby partition. Therefore, when object with similar prefix (sequential filename) are stored within S3 the performance degrades as its dependent on the IOPS of the storage partition. Its advisable to add random prefix to the object-name this way the object will be scattered across different portion within S3 storage resulting in better performance.
    - Download enhancements: To improve download performance its advisable to configure CloudFront URLs. Users instead of downloading the content directly from the S3 bucket it will download from the CloudFront URL. Where data will be cached to improve performance, also network latency will be minimized as CloudFront service is distributed across regions.

PUT load (req/min)	GET load (req/min)	What needs to be done?
< 100	< 300	Noting
> 100	> 300	Add random prefix Add CloudFront distribution
>300	> 800	Open a support request with AWS, for preparing for workload

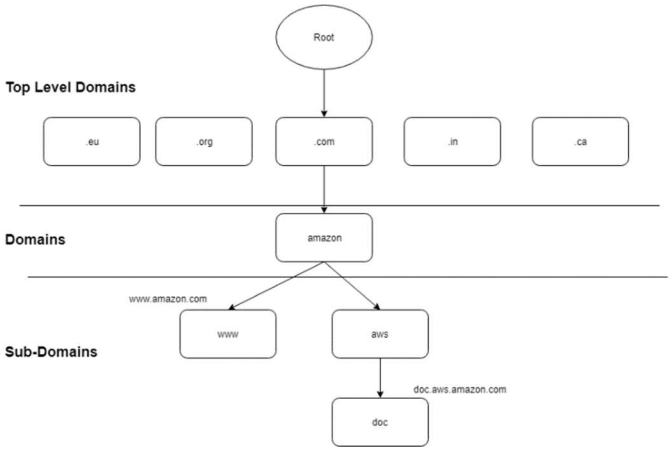
- S3 chargeable items
    - Storage charges /GB of data storage
    - Data transfer charges if applicable
        - No transfer charges within same region.
        - Data transfer to S3 is FREE
        - Data transfer to another region is chargeable
        - Data transfer to CloudFront is FREE.

- Upload/Download request (GET/PUT request) per 1000 request
    - In case of **requesterPay** the upload/download (GET/PUT) request will be paid by the request along with the data transfer fees. When this feature is enabled it doesn't support anonymous access/BitTorrent access. [requesterPay option needs to be enable at the bucket level from AWS console]
- Data Retrieval charges applicable to glacier and S3-IA

- S3 bucket event notification can be send to the following service, there is NO additional S3 charges for sending event notification. However, service charges will be applicable.
    - SNS
    - SQS
    - AWS Lambda function
- The following metrics can be recorded by **AWS CloudWatch**, filters (CloudWatch Dimensions) can be placed a the CloudWatch level to separate the CloudWatch data based on – bucket-name, storage-type, Prefix or Tags
    - S3 Request
    - Bucket Storage
    - Bucket Size
    - All request
    - HTTP 4XX and HTT5XX error responses
- Daily, bucket level CloudWatch metrics are enable by default with no additional fee, however to have detailed monitoring (1 min) can be enable at both Bucket Level and Object Level with additional charges. Up to 1000 metric configuration can be done at bucket level.
- By default, CloudTrail logs all API call at the bucket level, Object level tracking can be enabled. One need to configure the CloudTrail to deliver logs to S3 bucket to reference later.

Route 53

- AWS DNS (Domain Name System) is called Route 53
- Root-Server.org the root server (the top most server in DNS hierarchy)



- DNS Zone: this is the administrative authority that stores the address of all its register domains.
- Zone File contains: This includes the DNS Zone configure rule/mappings
- Primary Name Server: this is the server that has read write copy of the specific zone. It has the highest authority on the zone.
- Name server are the one that is responsible to answer the DNS query.

- Route 53 does the following
  - Register a domain for you
  - Its also a domain name register, it routes internet traffic to the domain resources
  - It performs health check on the register resource and reply accordingly to a DNS query.
    - If the resources are found not healthy it can also send notification while routing the resources to the alternative resources.
  - Route 53 can be use in different ways
    - As domain register AND router for internet traffic to your resources
    - As router for internet traffic ONLY, while domain in register with another domain registered. [this is needed a Route53 doesn't supports all the TLD.]
  - When a domain is register with Route 53, the following things happens. It creates itself DNS service for that domain
    - It creates hosted zone, matching to the domain
    - It allocates four unique name servers for each of the hosted zone, these name servers will be responsible for answering the DNS quires.
    - It creates & maintains a link between the domain name/hosted zone and the name server allocated in SOA (start of Authority) file.

Note: Amazon Route 53 doesn't support all TLDs, if the required TLD is not available with the Route 53 one can register their domain with another domain registrar where the TLD is supported and then create a hosted zone for the same domain within Route 53 which will allocate the name server for that domain. Once Route53 allocates the name server, replace the domain registrar name servers with the route 53 name servers. Any internet traffic that come to the domain then will be severed by the AWS name servers instead of the domain registrar name servers. This may take up to 48 hrs. to get reflected correctly based on the TTL values, as DNS servers cache name server information till TTL (time to live) counter is exhausted the DNS server will continue sending the old name servers.

- Insider the route 53 hosted zone, one needs to create record sets for delegation. Delegation means routing the internet traffic to specific name server.
- For registering a domain that is register with other domain registrar, one need to make sure the TLD (top level domain) does supported by Amazon AND, then one needs to get the authorization code from the current domain registrar.
- Hosted Zone: is a container where domain name will be mapped to name servers – this will help to route the internet traffic to the right host. It will have two entries by default – nameservers and SOA (start of the authority). SOA will contain information about the hosted zone and the nameserver will have unique nameservers for the hosted zone.
- One can transfer a register domain from one AWS account to another AWS account, by creating a support ticket with AWS. However, connected hosted zone with the account will not be transferred as a part of this transfer. DNS will still continue to work, even though the hosted zone and domain register in different AWS account.
- Supported DNS type record
  - A Record: Address Record , it maps a hostname with the IP address
  - AAAA Record: IPv6 Address Record, it maps hostname with IPV6 address.
  - **CName Record**: Maps alias name to the Record, this are use to translate the actual domain name to its alias name. CNAME can't be configure for the top node domain, only for secondary domain CNAME can be configured. For top node domain alias name can be configure instead of CNAME. When CNAME is configure for subdomain, you can't create any other record for which the value is value of the CNAME record.

- NS Records: Maps NS Servers with the Record
- **SOA Record**: Start of the authority Record, every zone has one and only one SOA record. It contains
  - **Domain Owner information:** its usually a email ID
  - **The authoritative server:** the list of name server
  - **Serial Number:** incremental serial number which increments with the change in the zone data. This number is very important to sync up primary and secondary zone server.
  - **Refreshing time/Time To Live (TTL):** How long this cached value need to be shaved is decided by the TTL.
- MX Record: Mail Exchange record, it defines where to deliver email for the users @ a domain name
- Alias Record (specific to Route 53 ONLY): this is configured for DNS Route 53 record, to route DNS queries to the AWS services for which the IP address can change for services like (Classic Load balancer. Application Load Balancer, CloudFront, S3 bucket). Each time a query comes, Route 53 will resolve alias define in the alias record entry and respond to the DNS query with the IP addresses fetches. Unlike CNAME it can be configured for the top node also, it can point to other records on the hosted zone.
- Records added to the hosted zone should have same prefix that of the hosted zone (domain name).

o NOTE: DNS quires done on the CNAME is chargeable wherein the quires done on the alias name is not chargeable.

o For Alias name one can't set TTL (time to live).

o Difference Between Alias Name and CNAME record

CNAME	Alias
CNAME can route a DNS query to any DNS Record, it does not need Route 53 as DNS service where the request is getting routed to. For Example, one can create a CNAME for mydomain.com that route its request to ABC.mydomain.com. OR it can route to mydomain.edu it doesn't need Route 53 as DNS service for ABC.mydomain.com or mydomain.edu endpoints.	In case of alias name, you can only redirect DNS quires to selected AWS resources - S3 bucket - CloudFront distribution - another record in the route 53 hosted zone that you are creating the alias record in. For example, one can create an alias doc. mydomain.com that redirects the DNS quires to S3 bucket OR it can create an alias ABC. mydomain.com to route request to another record XYZ.mydomain.com is the same/different hosted zone.
One can't create CNAME for the top node a.k.a. zone apex.	One can create alias name for top node as well for the sub domains. In most configurations, you can create an alias record that has the same name as the hosted zone (the zone apex). The one

CNAME	Alias
	exception is when you want to redirect queries from the zone apex (such as example.com) to a record in the same hosted zone that has a type of CNAME (such as zenith.example.com). The alias record must have the same type as the record you're routing traffic to, and creating a CNAME record for the zone apex isn't supported even for an alias record.
Route 53 charges for CNAME queries.	Route 53 doesn't charge for alias queries to AWS resources.
CNAME record redirects queries for a domain name regardless of record type.	Route 53 responds to a DNS query only when the name of the alias record (such as acme.example.com) and the type of the alias record (such as A or AAAA) match the name and type in the DNS query.
CNAME record can points to any DNS record hosted anywhere including the record that route 53 automatically creates when you create a policy record	Alias name record can only points to a cloudFront distribution, and ELB loadbalancer , an Amazon S3 bucket that is configure as static website OR to another record within the same Route 53 hosted zone where alias is created. One can't create a alias to that point to the record that route 53 automatically creates when you create a policy record.
CNAME record is visible in the answer section of a reply from Route 53 DNS server.	Alias name is only visible in route 53 console or in route 53 API.
CNAME record is followed by a recursive resolver.	Alias name is only followed inside Route 53, thus alias record and the target must exist in Route 53

- Route 53 routing policies
  - **Simple routing:** This is used to configure single resource for the domain example, a webserver.
  - **Failover routing:** This is used to configure active passive site, when the primary site is down queries will be automatically reply with the secondary site.
  - **Geolocation routing policy:** This is use when the routing needs to performed based on the geographic location of the requester. The content can be routed based on continent, country and based on united states. *Incase of the geolocation routing, always the location which is more accurate will be preferred. Default routing policy needs to be added for those IPs which cannot me mapped or mapping is not available.*
  - **Latency routing policy:** based on the requester location, DNS query will be replied with the less latency site. *This is different from the geolocation routing as the routing is solely depends upon latency / performance, for some reason if the*

latency is less for resources outside the geographic location then the resource will be server by that resources instead of co-located resources.

- **Weighted routing policy**: use the routing incoming traffic based on the weighted routing rule configured within the policy.
- **Geoproximity routing policy**: Based on the geo location of the requester and the resource AWS Route 53 will route the incoming request. One can optionally route more traffic or less traffic to a resource by specifying a value know as 'Bias'. By increasing or decreasing the bias value one can decide expand or shrink a region based on which it will decide where request needs to be routed to.
- **Multivalued routing policy:** Multivalued routing allows returning more than one value (IP address) in response to a DNS query, which enable the users to choose IP from list of available IPs.

- AWS Route 53 pricing
  - Per Hosted zone charges (there is no prorated charges, it will be charge for whole month)
  - Standard DNS query per million records
  - Latency based DNS query per million record has different charges as AWS need to process more information.
  - Geoproximity and Geolocation base DNS query will be charge more than latency base DNS query and Standard DNS query.
  - For performing health check additional charges will be applicable, more charges will be applicable for the health check those are to done on premises.
  - Routing based on CNAME will be chargeable.
  - Routing base on Alias name will be FREE
- **Traffic Flow: [Need to read about it - TBD]**

CloudFront

- AWS CloudFront, amazon managed content delivery service. AWS CloudFront leverages AWS edge location to distribute content there by reducing the distribution load on the origin server and improve customer experience as customer will be served from a nearby location instead of actual origin location. When customer make a request Route 53 redirect the request to the nearby edge location base on the latency , once the customer receives the request at the edge location, edge location verify if the request is available in its cache if not edge location makes a request to the origin server for the content over AWS infra structure (which is much faster then, downloading the content over internet). Once first few bytes of the request started to appearing edge location starts delivering the request to the requester. Once, the content is delivered to the requester the content will be stored in the edge location cache for severing future request.
- CloudFront performs better with Static Content as they can be cached easily. For Dynamic Content, depending on the query string content can be cached but still there will be a greater number of calls to the origin. Apart from getting caching benefits – AWS CloudFront also provide other benefits like
  - **Security to the content:** CloudFront seamlessly integrated with AWS WAF (Web Application Firewall) and AWS Shield to provide required protection from refine threats like DoSS (Denial of Service), it can also be integrated with ACM (Amazon Certificate

Manager) service to manage and maintain secure connection to the content without overhead of managing/renewing certificates.

- **Delivering content over vast network:** CloudFront provide content over vast network of Edge-location and Reginal Edge-Location spread all over the globe, ensure faster connectivity and lower latency from anywhere in the world.
- **Greater Performance:** CloudFront Edge-location and Reginal-Edge-Location ensure content can be uploaded / downloaded faster will minimum latency as Edge-location and Reginal-Edge-Location are connected to the AWS infrastructure backend which provide high speed connectivity to the AWS services.
- **Programable Content Delivery Network:** AWS Lambda@edge helps the lambda functions to run near requester location, ensure lower latency.
- **Economical:** CloudFront charges only for the content that is transferred through its network, there is additional fee associated with the configuration or setup or transferring dynamic content. When the download rate is very high, it's advisable to use CloudFront as its cheaper to use CloudFront then directly accessing the content from the S3 bucket.
- **Field level encryption:** Additional field level encryption can be turn on to encrypt sensitive data at the edge location closer to the requester and keep it encrypted till the application layer which has the password to decrypted the data for processing.
- **Applying geo-restriction or geo-blocking:** Using CloudFront one can block specific object to be access from a particular geography. **This restriction is applicable at the distribution level.**

- CloudFront is a global service.
- CloudFront servers both ingress and egress connection i.e. for uploading content to the origin and for downloading content from the origin.
- There are two type of distribution that can be configure with the CloudFront
  - Web distribution
  - RTMP (Real Time Messaging Protocol) distribution.
- CloudFront can be configured using – AWS console, AWS API, AWS SDK, AWS CLI, AWS toll for power shell.
- Note: AWS doesn't recommend to use CloudFront for PCI DSS related information to store information in CloudFront Cache, however CloudFront is HIPPA (Health Insurance Portability and Accountability Act.) certified service. CloudFront is NOT PCIDSS compliant.
- CloudFront – Edge Location and Regional Edge Location. For less frequently access data, AWS automatically moves it to the reginal Edge catch location which has more cache space then a regular Edge location. When requester request for a specific data, instead it fetch from the origin location, edge location can fetch the data from its nearby regional edge catch location adding performance enhancements to the customer experience. Regional Edge cache doesn't cache the dynamic content. Proxy method PUT/POST/PATCH/OPTION/DELETE doesn't go through reginal cache.
- Default CloudFront cache for any object is 24 hours and minimum cache time is 0 hours, this can be set by configuring TTL (time-to-live) setting in CloudFront configuration.
- Custom s3 origin (static website) will only be served through CloudFront, regular S3 origin will not go through the CloudFront distribution.
- Web Distribution can be used to share the following over HTTP or HTTPs
  - Static content like .js,.html,.css image files
  - Media content using on demand progressive download and APPLE HTTP live streaming.
  - Cannot be use for Adobe Flash Multimedia content over HTTP or HTTPs.

- RTMP distribution
  - Adobe Flash Multimedia can be share
- CloudFront web Distribution configuration setting
  - If the content is share with everyone or to restricted set of users.
  - Whether the CloudFront is distributing its content over HTTP or HTTPS
  - Whether CloudFront to forward cookies and/or query string to the origin, Whether to cache content base on all query parameters or on selected parameters.
  - Defining geo-restriction to the CloudFront URL – this will prevent content to distributed in the define geo-restriction locations. One can white list or black list countries.
  - Maximum limit of 200 web distribution per AWS account.
- CloudFront RTMP Distribution configuration setting
  - This distribution is required to send Adobe Flash Multimedia streaming videos.
  - For RTMP distribution the origin should be a S3 bucket unlike in case of web distribution where origin can be a webserver or a S3 bucket.
  - Maximum limit of 100 RTMP distribution per account.
- CloudFront Origin:
  - **Origin:** The DNS name of the S3 bucket for which CloudFront is to be configure to get objects from this origin.
  - **Origin Group**: When origin group is set, in an event of an origin failure (return specific HTTP code) the content will be severed to the request by forwarding the request to the secondary origin within the origin group. One can designate, primary origin and secondary origin in origin group.
  - **Custom Origin:** The DNS name of the HTTP server for which you want CloudFront to get object from this origin. The custom origin can server from an EC2 instance or from Elastic Load balancer or from S3 static website. For RTMP distribution, custom origin can't be used.

    Few Guidelines for the custom origin
    - Use HTTP keep-Alive header to improve performance
    - Ensure Date and Last modified header are accurate on the generated content
    - Don't use query String as those are not cached by CloudFront
    - Use Cache-Control header to made the content identifiable, whether it can be cached or not.
    - CloudFront accepts HTTP 1.1 request but makes only HTTP 1.0 request to backend origin server.
- CloudFront Configuration Steps (high level)
  - Define origin type - origin or custom origin
  - Define Access – Private or public access
  - Configure CloudFront distribution – web distribution or RTMP distribution
  - Define caching logic, and TTL.
- Cloud Front Alternative domain name: This can be done for the web-distribution or for RTMP-distribution.
  - Create a DNS to route the alternative DNS to CloudFront domain name. This can be done by configure alias in Route 53 OR for other DNS provides add a CNAME resources record set to the hosted zone for CNAME.
- Path parameters the following things can be configure within the CloudFront distribution using a path parameter
  - The path patter e.g. /*.php or /*.jpg

- For multiple origin, it can be used to configure CloudFront distribution to specify which request needs to be forwarded to which origin.
- To configure, if the query parameters needs to be forwarded to the configured origin.
- Whether, for accessing a specific file required a signed URL.
- Allow HTTP or HTTPS for accessing a specific file.
- The minimum time for which the file will remain in the CloudFront cache Regardless of the value define in the cache header which is set at the origin.

- Allowed CloudFront methods

Method Name	Description
GET	Request Data from a specific resource.
POST	Submit a data to be processed to a specific resource.
HEAD	Same as that of the GET but returns only the HTTP header but not the document body
PUT	Upload a representation for the specific URI
DELETE	Delete a specific resource
OPTIONS	Return the HTTP method that server supports.
PATCH	Submit partial modification to the object

CloudFront ONLY cache's GET, HEAD method by default optionally it can be configured to cache OPTION method. However, it cannot be configured to cache PUT POST, DELETE and PATCH methods.

- Viewer Protocol policy: what protocol needs to be set between the requester and the edge location (CloudFront location)
  - **HTTP and HTTPS** – for requester to choose the HTTP or HTTPS.
  - **Redirect HTTP to HTTPS** – redirecting viewer HTTP call into HTTPS request.
  - **Only HTTPS** – requester needs to use ONLY HTTPS protocol.
- Origin Protocol policy:
  - HTTPS Only – the communication between CloudFront and the Origin will be HTTPs
  - Match Viewer Protocol – base on the selection of the requester protocol, communication between CloudFront and the Origin will be determined. This is mostly use in conjunction with Viewer Protocol Policy: Redirect HTTP to HTTPS or HTTPS only.
    Note: for S3 static website origin – the origin policy is set to HTTP ONLY, as it doesn't support HTTPS.
- **CloudFront Object invalidation**: The process of refreshing the cache objects from the CloudFront web distribution is called Object Invalidation. This is performed at the object level and its chargeable. If a large number of objects within a same origin needs to be refresh then it's always advisable to build a new web distribution and replace the old one with the new one instead of invalidating each object as creation of web distribution is not chargeable whereas object invalidation is.
- Accessing Private Content through CloudFront Service.
  - **Signed URL** – user needs to be authenticated themselves, once authenticated successfully, they will get their signed URL to access the private content. However, within the bucket we need to configure OAI (Origin access identity) within the object which are allowed so that they can only be access by CloudFront signed URL. (This can be done for both, origin and custom origin).

- **Signed Cookies** – By inserting a singed cookie in the request header, one can allow accessing a private (restricted) content. With signed cookies the URL does not change it remains the same.

- Video Streaming – on-demand streaming & live streaming are two types of video streaming supported by AWS. Web distribution can be configured with one of the following configurations
  - Configuring on-demand with AWS element media store
  - Configuring on-demand with smooth streaming
  - Configuring on-demand with progressive download
  - Configuring on-demand with APPLE HTTP Live Streaming (HLS)

  For configuring Adobe live streaming, one need to use RTMP based distribution. In case of the adobe live streaming, there are two sperate files that gets distributed – media player and the adobe media file content. For media player download one need to use web distribution and for the adobe media file content RTMP distribution needs to be downloaded. For RTMP distribution, the origin should be always a S3 bucket.

  Note: On-demand streaming can be configured, but for Live streaming one need to involved AWS for configuration.

- **CloudFront Access Logs** – CloudFront access logs are captured by Access Logs. This needs to be enabled while configuring the distribution and can be redirected to store in a S3 bucket. CloudFront logs can be analyzed using Amazon Athena which is interactive query service.
- All the trail logs to CloudFront API, made from AWS console, AWS SDK or AWS API are stored in CloudTrail API not in Access Logs.
- **CloudFront Pricing:** The following thigs are chargeable in CloudFront
  - Changes for S3 or EC2 instances depending upon the origin.
  - Data transfer to the requester are chargeable. (Data transfer, from the CloudFront and the requester is chargeable).
  - Data transfer changes applicable from uploading content from the CloudFront edge location to the origin will be chargeable.
  - HTTPS request
  - Request for Field Level encryption
  - Invalidation of the objects
  - No charges for data transfer between the CloudFront and the origin
  - No charges for regional CloudFront caching
  - No charges for AWS ACM TSL/SSL certificates for configuring SSL connection
  - No charges for shared CloudFront certificates
  - No extra charges for using HTTPS

CloudTrail

- CloudTrail logs all management & data events of all users performed from AWS console, AWS API, AWS SDK or from AWS CLI.
- By default, CloudTrail logs all the events which can be view from CloudTrail console for 90 days, but it needs to be explicitly enable sending the logs to the specific bucket which can be use later, alternatively the logs can also be send to CloudWatch logs and CloudWatch events – which can be used to create custom metrics or CloudWatch alarm.

- Benefit of enabling CloudTrail it tracks - who make the change, when the change was made, what changes were made. It's needed for - For security & compliance needs, for tracking needs, for operational & troubleshooting needs.
- **[Important]** When enabling a CloudTrail there are option to *enable it for a specific region* or for *all region*. *AWS recommend enabling it for all regions – so that all event from different region automatically gets log to the specified bucket.*
- There are max 5 CloudTrail created per region. [*Trail that are enable for all region counts in all region*]
- Once enable, it takes 15 minutes to start logging. Once started it logs every 5 min.

Encryption – KMS / (HSM) Hardware Security Module

- The three things required for encryption are 1) **Data** that needs to be encrypted, 2) the **encryption algorithm**, and 3) the **encryption key**.
- There are two types of encryption key types – **Symmetric Encryption Key** (where encryption and decryption keys are same) OR **Asymmetric Encryption Key** (where encryption and decryption keys are different). AWS supports Symmetric Encryption Key technique.
- **KMI (Key Management Infrastructure)** – **Amazon** KMS (**Key Management Service**) is a type of KMI (Key Management Infrastructure) offered by Amazon for encryption key management. There are two parts to any KMI *1) Storage layer that protects the plaintext key(s)* and *2) Management layer that authorized access to key usages.*
- Amazon HSM (hardware security module) provides – **FIPS 140-2 Level 3** validated single tenant HSM cluster in your VPC for using and storing keys.

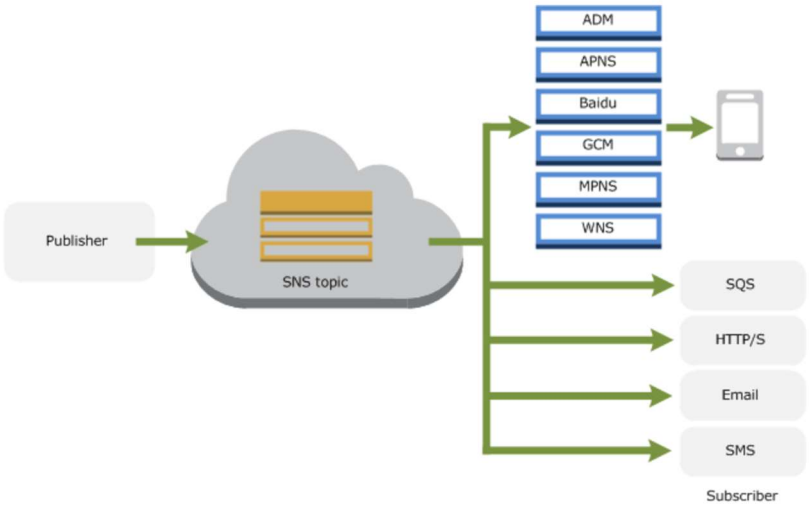
  FIPS 140-2 Level 3 – **(Federal Information Processing Standards)** are set of standards that describes for processing documents, encryption algorithm and other information technology standards for use within non-military government agencies and the government contractors and vendors who works with the government agencies.
- Amazon HSM can be used for varieties of use cases
  - Digital Right Management (DRM)
  - Public Key Infrastructure
  - Document signing
  - Cryptographic functions
- Amazon KMS – Managed service by Amazon which is supported by FIPS 140-2 level 3 validated module (HSM). This can be leveraged by most of the Amazon services for there encryption and decryption needs. CloudTrail logs all the event related to the Amazon KMS – which helps in identifying how master key are getting use and by whom. ALSO logs the uses of the encryption keys for auditing, compliance and regulation needs.
- [important] KMS is a global service however the keys are confined to a region where they are created. (*they cannot be transmitted outside the region where they have been created*).
- KMS maintained multiple copy of the key across different availability zone across region to provide a durability of 99.999999999% of availability. User can import their own keys into KMS, in that case use needs to maintain their own key to re-imports the key if keys are lost due to KMS failure. KSM can't be use to export master key.
- Customer Master Key (CMK) can be use to generate, encrypt or decrypt data up-to 4kb (4086 bytes) – they are used to encrypt data key which then can be use to encrypt data of any size. *[this is typically known as envelop encryption]*.

- Customer Master Key (CMK) can't be exported outside of the KMS. Data key can be exported outside the KMS for data encryption.
- There are two types of CMK
  - **Customer Managed – CMK**
    - Customer needs to managed enabling and disabling of the CMK
    - Customer needs to rotate cryptographic material
    - Customer needs to create IAM policies to govern access to the CMK
    - Customer needs to use the CMK for their cryptographic operations, and can allow AWS service to use CMK on their behalf.
  - **AWS Managed – CMK**
    - AWS create, managed and use the CMK on behalf of the user
    - AWS keep the CMK unique for a particular region
    - ONLY those services that creates CMK uses AWS managed CMK for encryption purposes.
    - AWS managed CMK can be easily identifiable by their specific naming convention - aws/[service-name] e.g. aws/lamda or aws/rds etc.
- Default Master Key
  - When the first time an encrypted resource is created a default customer master key is created.
  - AWS manage the policies for the default customer master key to ensure new feature in supported service automatically.

SNS (Simple Notification Service)

- AWS fully managed push notification service.
- Single message sent to all subscribers – Fan out message to all its subscriber immediately.
- Reliability of the SNS – AWS store SNS messages into 3 or more Availability Zone in a same region.
- Security of the SNS – AWS authenticate users before allowing access to the SNS topic. AWS also suggest use of secure channel (SSL) to connect to SNS topic over network to secure the message while transit.
- Authentication of the SNS topic – AWS requires all its publishers to signed the message with the secret key of the AWS ID & it validates the signature included in the request messages. By default, ONLY SNS owner can only publish messages to its SNS or give permission to other AWS account to publish messages to the SNS queue.
- SNS notification can be use to push notification to the both mobile & desktop. SNS Mobile push notification service supports on the following platform.
  - Apple Devices
  - Google Devices
  - Fire OS
  - Windows device
  - Android Devices over china through Baidu Cloud Service.
- To received SNS push notification mobile devices should have the APP installed on their application, in case of Apple, Android, Fire OS platform it also required to provide explicit permission on the device to received push notification.
- Currently, amazon push messaging supports following platforms
  - Amazon Device Messaging Service (ADM)

- Apple Push notification service (APNS)
- Google Cloud Messaging (GCM) – for windows 8+ and 8.1+ phones
- Windows push notification service (WPNS) – for windows 8+ and 8.1+ phones
- Microsoft Push Notification Service (MPNS) – for windows 7+ phones
- Baidu Cloud Service for pushing notification to android device in China



- Up to 10 message attributes can be pass along with the message body (payload) which can be used or message filtering. Attribute can carry only string values not JSON object.
- SNS direct messaging – send specific messages to a single endpoint there by allowing to deliver message directly to an intended endpoint.
- CloudTrail logs SNS request details, which can be audited later – however only authenticated API calls are ONLY logged in the CloudTrail.

Beanstalk

- Amazon BeanStalk provide easy way to deploying application as it manages all the underline capacity provisioning, complexity of instances, load balancing, application health monitoring, application logging for hosting application in its supported language.
- To use Elastic Beanstalk, create an application, upload an application version in the form of an application source bundle (for example, a Java .war file) to Elastic Beanstalk, and then provide some information about the application. Elastic Beanstalk automatically launches an environment and creates and configures the AWS resources needed to run your code. After your environment is launched, you can then manage your environment and deploy new application versions.
- Supporting subset of platform for deploying applications
    - Supported languages: JAVA, .NET, Node.js, Python, Ruby, GO
    - Supported platforms: Web container, Docker container

CloudFormation	OpsWorks	BeanStalk
Need control more on the infrastructure	Need more control over deploying application	Less control over deploying application
More towards – configuring infrastructure.	Give more control over deploying application	Less control, less complexity for application deployment.
		In case of BeanStalk application health monitoring is included.

Stack	Stack	Environment

- There is no additional charge for Elastic Beanstalk. You pay only for the underlying AWS resources that your application consumes.
- Elastic beanstalk can also be used to build custom platform, one can create their own custom build platform and use it like other build platform available to them.
- Instances running on Amazon Beanstalk doesn't have persistence local storage. When an instance is terminated the data stored in the EC2 instance will be lost. (RDS instance created a within amazon beanstalk will lose its stored data once the instance is terminated).
- Elastic Beanstalk can use other persistence store to store its data like – S3, RDS, dynomoDB, RDS, EFS.
- If the underline aws resources created by Elastic beanstalk are terminated or modified directly then there is a possibility that the full environment becomes unusable – rebuilding the environment will terminate existing resources and recreate the new resources with same configuration.
- Within 6 weeks (42 days) of a termination of an Elastic Beanstalk environment – it can be rebuilt. When rebuild, aws recreate all the environment with the same name and configuration (if available with the same ID).
- **Application** – In case of Amazon Beanstalk, application is the container that run on the amazon beanstalk environment. It comprises of – versions of source code, saved configuration logs and other artifacts. Amazon beanstalk application is logical collection of – environment, versions & configuration. Its conceptually equivalent to a folder. Deleting an application, deletes all the source code version, saved configuration and the environment.
- **Application version** – refers to a unique, specific labeled, iterable version of a deployable code, which points to a specific deployable file in a S3 bucket within Amazon beanstalk environment. Application can have multiple version within a same Amazon beanstalk environment, but only one running version.
- **Environment** – the container that run a specific application version. Multiple application version of a same application can be run simultaneously in different environment.
- **Environment Tier** – User selects environment tier, based on the environment tier value amazon beanstalk provision resources to support the application. Web application runs web environment, backend processing application runs on worker application.
- **Environment Configuration** – These are the collection of parameters and the settings based on which an environment behaves to cater the needs of an application. Any change in the environment configuration – amazon beanstalk will apply the changes to the underline AWS resources on your behalf, else it will terminate the instances and create a new resource with the new configuration.
- **Saved Configuration Template:** Starting point for creating unique configuration for the amazon beanstalk environment. Configuration template can be update using beanstalk console, AWS CLI or through API. When using saved configuration within CLI or through API they are referred as Template.
- **Platform:** It's a combination of Operating System, programming language runtime, application server and elastic beanstalk components. Beanstalk provides various different platforms; user needs to create their application targeting a specific platform. It provides platforms for different programming language, application servers and docker container.
- **Elastic bean cloud Shared Responsibility Model –** It applies the patches and mirror release on your behalf. Update major releases as available platform within 30 days of the releases.
  Customer Responsibility:

- Implementing application level security or any data or components that was downloaded separately and not available as part of the platform.
- Keep platform updated, migrating retired platform to supported platform
- Act on failed update notification, (*when a update failed, amazon beanstalk notifies the customers, for them to act on it*)
- Patching OS in case customer opt out from Elastic Beanstalk to patch platform.
- Managing security for the AWS services that are hosted outside the beanstalk platform.