# AWS Solutions Architect Study Guide

Blueprint

* 130 minutes
* 65 Questions
* Multiple Choice
* Passing score 720
* Valid 3 years
* Scenario Based

Read the AWS Security Best Practice white paper.

# Security

## AWS IAM

* Allows you to manage users and their level of access to the AWS console.
* Universal/global can be viewed by any region
* Arn: partition:service:region:account\_id:

**What does IAM give you?**

* Centralized control of your AWS account
* Shared access to your AWS account
* Granular Permissions
* Identity Federation (including Active Directory, Facebook, linkedin, etc.)
* Multifactor Authentication
* Provides temporary access for users/devices and services as necessary.
* Allows you to set up your own password rotation policy
* Integrates with many different AWS Services
* Supports PCI DSS Compliance

**Core Concepts**

* Users: end users (think people)
* Groups: A collection of users under one set of permissions. (a group of users and apply policies to them collectively
* Roles: You can create roles and then assign them to AWS resources.
* Policies: A document that defines one (or more) permissions (json form key value pairs)

Under security, identity and compliance

* Power user: Access to all AWS services except the management of groups and users within IAM.

**Best Practices**

* Each user has their own account and then create a group

**Two types of AWS access**

* Programmatic access: How applications access AWS and how you access AWS from the command line.
* AWS console Access: How you access the system via AWS console

.

**Web Identity Federation**

* Let’s you give your users access to AWS resources after successfully authenticating with a web-based identity provider such as Amazon, Facebook or Google.
* Following successful authentication, the user receives an authenticati4on code from the Web ID provider, which they can trade in for temporary AWS security Credentials.

**STS AssumeRoleWithWebIdentity**

* Assume-role with-web-identity: An API provided by STS (Security Token Service)
* Returns temporary security credentials for users authenticated by a mobile or web application or using a Web ID provider like Amazon, Facebook, Google etc.
* For mobile applications, Cognito is recommended.
* Regular web apps can use the assume-role-with-web-identity API.
* Once the user has authenticated the assume-role-with-web-identity API calls.
* If successful, STS will return temporary credentials enabling access to AWS resources
* AssumedRoleUSER ARN and AssumedRoleID – are used to programmatically reference the temporary credentials – not an IAM role or user

# A picture containing diagram Description automatically generated

# Storage

## S3 - Simple Storage Service

**Summary**

* **Object based** storage; NOT block storage. (No databases or OS)
* **Object based just means files**
* **Secure, durable, highly scalable object storage.**
* Files can be from 0 Bytes to 5 TB
* Limited to 100 buckets per account
* Unlimited storage
* **Must have unique DNS name** (all lower case) even though buckets are global
* Read after write consistency for PUTS of new objects
* Eventual Consistency for overwrite PUTS and DELETES (can take some time to propagate)
* Supports at least 3,500 requests per second to add data and 5,500 requests per second to retrieve data.
* Upload to s3 receives an HTTP 200
* LifeCycle Management: automates moving objects between the different storage tiers

**Key Fundamentals of S3 Object**

* Key Value Store {name: “hi.txt”, data: “Hello World”}
* Version ID (for versioning)
* Metadata (data about data you are storing)
* Sub resources - bucket specific configuration
  + Policies, Access Control Lists (IAM stuff)
  + CORS
  + **Transfer Acceleration** - **Use CloudFront to optimize transfers**

**Data Consistency**

* **Read after write** consistency for **PUTS** of **new** objects (means when uploaded to s3 immediately you can read it)
* **Eventual** consistency for overwrites **UPDATES** (PUT existing)and **DELETES** (**can take some time to see changes**)
* 5TB to put one singular item. 5GB max for PUTs if it is a multi-part upload
* Multi-part uploads recommended for files over 100MB but required for files over 5GB

**S3 Byte-Range Fetches**

* Parallelize downloads by specifying byte ranges

**Storage Classes**

1. **S3 Standard** - 99.99% availability, 99. (9\*11) durability
2. **S3 Infrequently Accessed(IA)**- 99.9% availability, Lower fee than S3 but you are charged a retrieval fee, good for files accessed every 1 - 6 months
3. **S3 Intelligent Tiering** –Using machine learning to optimize costs by automatically moving data to the most cost-effective access tier, without performance impact or operational overhead
4. **S3 One Zone IA** - Same as IA but only stored in one availability zone 99.5% availability cost is 20% less
5. **Glacier** - VERY cheap, used for archiving, takes 3-5 hours to access. Retrieval time of 12hour is acceptable. Configurable retrieval time
6. **S3 Glacier Deep Archive** – Amazon S3’s lowest cost storage class where retrieval time is 12 hours.

**Charged For**

* Storage (per GB)
* Requests (Get, Copy, Put)
* Storage management (Analytics, Tagging)
* Data transfer pricing
* Transfer Acceleration
* Cross Region Replication Pricing

**Security**

* By default, buckets are private
* Bucket policies - applied at **bucket** level
* Access Control lists (ACL) - applied at **object** (file) level
* S3 buckets can be configured to create logs (even to another bucket) for access control audits
* Two ways to access file, using open and clicking link (clicking link will be error if file isn’t public)
* Two ways to access file, using open and clicking link (clicking link will be error if file isn’t public)
* Encrypts with KMS

**Data Consistency**

* **Read after write** consistency for **PUTS** of **new** objects
* **Eventual** consistency for overwrites **UPDATES** (PUT existing)and **DELETES** (can take some time to see changes)
* 5TB to put one singular item. 5GB max for PUTs if it is a multi-part upload

**S3 Encryption**

* In transit. Encrypting data when uploading and downloading files. Uses SSL/TLS
* Each object encrypted with own unique key

**Customer side encryption**

* SSE-C customer provided key where cust manages own key and AWS encrypts and decrypts for you.

Client-side encryption

* AES-256 use server-side encryption with amazon s3 managed keys **(SSE-s3**)
* AWS-KMS use server-side encryption with AWS-KMS managed keys (**SSE-KMS**)
* /\* makes the item a wild card

**S3 Cors**

* Enables cross origin access for your AWS resources
* By default, resources in one bucket cannot reach resources in another

**2 Main approaches to performance Optimization for S3:**

* GET-Intensive workloads-use cloudfront
* Mixed-workloads- avoid sequential key names for 23 objects. Add a random prefix like hax hash to the key name to prevent multiple objects from being stored on the same partition.

**S3 Object Lock**

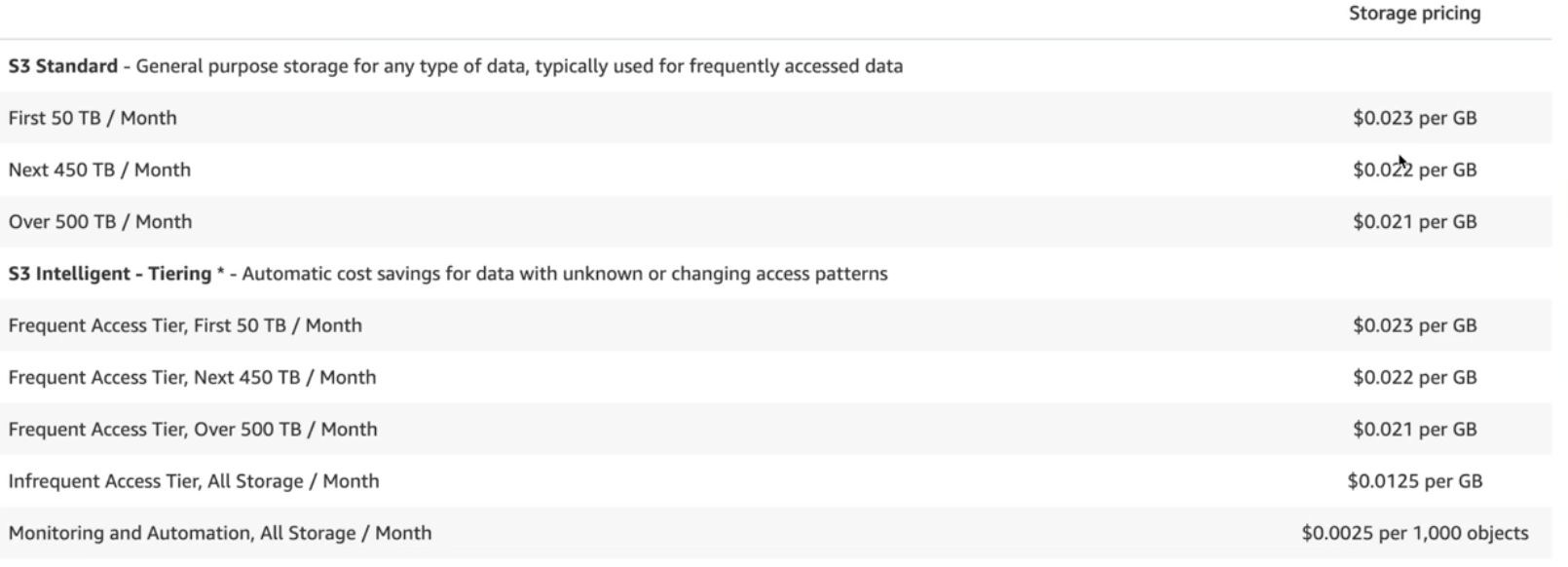
* Use s3 object lock to store objects using a write once read many **(WORM)** models.Helps prevent objects from being deleted or modified for a fixed amount of time indefinitely. Can be locked across bucket or individual items.
  + **Governance mode:** users can’t overwrite or delete an object version or alter its lock settings **unless they have special permissions.**
  + **Compliance mode:** A protected object version **cannot be overwritten or deleted by any user.**
  + **Retention Period:** periods an object version for a fixed amount of time
  + **Legal Hold:** Prevents an object version from being overwritten or deleted but doesn’t have an associated retention period

**Glacier Vault Lock**

* Allows you to easily deploy and enforce compliance controls for individual S3 Glacier vaults with a vault lock policy. You can specify controls such as WORM. Once locked, the policy can no longer be changed.

**How to get the best value out of S3**

1. S3 standard (avoid)
2. S3 IA
3. S3 interlligent tiering
4. S3 One Zone -IA
5. S3 Glacier
6. S3 Glacier Deep Archive



Graphical user interface, application, Teams

Description automatically generated

**Prefixes**

* Between the bucket name and bucket object for example my bucket/folder3/myfile.jpg the prefix is folder3
* **3,500** PUT/COPY/POST/DELETE and **5,500** GET/HEAD requests per second per prefix.

**S3 Select and Glacier select**

* Get data by rows or columns
* Save money on data transfer and increase speed
* Enables applications **to retrieve only a subset of data** from an object by using simple SQL expressions. This helps achieve drastic performance increases can be as much as 400% improvement

**3 Ways to share s3 Bucket access across accounts**

* Using bucket policies & IAM (applies across the entire bucket) programmatic access only
* Using bucket ACL’s & IAM (individual objects). Programattic access only
* Cross-account Iam roles. **Programmatic and console access**

**Cross Region Replication**

* Must enable bucket replication in order for cross-region replication to work.
* Cross replication does not back track it only replicates from that point in time forward it was enabled.
* Versioning must be enables on both the source and destination buckets
* All subsequent updated files will be replicated automatically
* Delete markers are not replicated

**Transfer Acceleration**

* **Use CloudFront to optimize transfers**
* Google s3 acceleration tool and there is a tool to test this

**Multi-part upload**

* Multipart upload delivers the ability to begin an upload before you know the final object size.
* Multipart upload delivers the ability to pause and resume object uploads.
* Multipart upload delivers quick recovery from network issues.
* Multipart upload delivers improved throughput.

## Amazon FSx for Windows file server

* A managed windows server that runs windows server message block (SMB)-based file services.
* **Designed for windows and windows application**
* Designed for windows and windows apps

## Amazon FSx for Lustre

* Fully managed file system that is **optimized for compute-intensive workloads**, such as **high-performance computing, machine learning, media data processing workflows, and electronic design automation (EDA)**
* Launch and run a lustre file system that can process massive data sets up to hundreds of gigabytes per second of throughput, millions of IOPS, and sub millisecond latencies.

## Amazon EFS

* A managed NAS filer for EC2 instances based on Network File System (NFS) version 4
* **Native to Unix and Linux**
* Pay for the storage you use (no pre-provisioning required)
* Scale up to petabytes
* Can support thousands of concurrent NFS connections
* Read after write consistency
* When mounted on Amazon EC2 instances, an Amazon EFS file system provides a standard file system interface and file system access semantics, allowing you to seamlessly integrate Amazon EFS with your existing applications and tools
* Multiple Amazon EC2 instances can access an Amazon EFS file system at the same time

**EFS vs. FSx for Windows vs. FSx for Lustre**

**EFS**

* **When you need distributed, highly resilient storage for linux instances and linux-based applications**

**Amazon FSx for Windows**

* **When you need centralized storage for windows based applications** such as sharepoint, Microsoft SQL server, workspaces, IIS Web Server or any native Microsoft Application.

**Amazon FSx for Lustre**

* **When you need high-speed, high capacity distributed storage.** This will be for apps that do high performance compute (HPC), Financial modeling etc. Remember FSx for lustre can store data directly on S3

# AWS EBS Elastic Block Store

**Summary**

* Storage volumes that you attach to EC2 instances.
* Non-root volumes can be detached from ec2 instances while running
* Automatically replicated within its availability zone to protect you from component failure, offering high availability and durability
* Automatically in the same AZ as EC2 instance
* Termination protection turned off by default
* Default action is for the root EBS volume to be deleted when. the instance is terminated
* To enable encryption at rest using Ec2 and EBS you must configure encryption when creating the EBS volume(

**Snapshots**

* Snapshot are point in time copies of volumes
* Snapshots are incremental- only the blocks that have changed since the last snapshot are moved to S3
* Stop instances before snapshots
* EBS volumes can be changed on the fly, this includes changing the size and storage type
* **To move ec2 volume from one region or AZ to another take a snapshot of it, create an AMI from the snapshot and use the AMI to launch the EC2 instance**
* Can share unencrypted snapshots
* **Snapshots of encrypted volumes are encrypted**
* Can be created via api, command line and console
* Snapshots of root devices of an EBS volume used by a registered AMI cannot be deleted. AWS protects you from accidentally deleting the EBS Snapshot, since it could be critical to your systems. To delete, first remove the AMI, then the snapshot can be deleted.
* Elastic Block Storage (EBS) is recommended block level storage for EC2 instances if you were running a database on an EC2 instance.

**EBS Volumes are AVAILABILITY ZONE locked!** One of the only things that are AZ specific

Calendar

Description automatically generated

## Storage Gateway

* A service that **connects an on-premises software appliance with cloud-based storage** to provide seamless and secure integration between an orgs on-prem IT environment and AWS’s storage.
* Enables you to securely store data to the AWS cloud for scalable and cost-effective storage
* A virtual or physical device that replicates your data into AWS
* AWS Storage Gateway’s software appliance available for download as a VM image that you install on a host in your datacenter. Storage. Gateway supports either VMware ESXi or Microsoft Hyper-V. Once you’ve inst
* **3 different storage types:**
  + **File gateway:** **NFS & SMB** for flat files, stores files directly on S3.
    - **Connects from server through an NFS mount connects to storage Gateway then uses direct connect/internet to connect to S3 directly.**
  + **Volume Gateway: (iSCSi) storing versions of your virtual or hard-disk drives in S3.**
  + Two volume types
    - **Stored Volumes:** **Entire dataset is stored on site** and is asynchronously backed up to S3.
    - **Cached Volumes:** Entire Dataset is stored on S3 and **the most frequently accessed data** is cached on site.
  + **Tape Gateway Virtual Tape Library (VTL): helps you move your tape-based backup and archiving workflows to AWS** cloud easily. Uses NetBackup, Backup Exec, Veeam etc

# Data Transfer

## AWS Snowball

* **Petabyte** scale data transport solution that uses secure appliances to transfer large amounts of data in and out of AWS.
* 1/5th the cost of using high speed internet
* Snowball edge is a 100TB data transfer device with on-board storage

## AWS Snow Mobile

* an **Exabyte-scale** data transfer service used to move extremely large amounts of data to AWS

## AWS DataSync

* Connects to Nas or file system
* Connects to s3 , EFS or amazon FSX to copy data to and from AWS
* Uses:
  1. To move large amounts of data from on-premises to AWS
  2. Used with NFS and SMB- compatible file systems
  3. Replication can be done hourly, daily or weekly
  4. Install DataSync agent to begin replication
  5. Can be used to replicate EFS to EFS

## AWS Athena

* Interactive query service which enables you to analyze and query data located in S3 using standard SQL.
* Serverless
* Pay per query/ per TB scanned
* Supports CSV, TSV, JSON on text-files
* Supported open-source columnar formats such as Apache ORC and Apache Parquet
* Supported compressed data includes snappy, zlib, LZO and GZIP
* Works directly with data stored in s3
* When to use:
  + Query log files stored in s3 ex. ELB logs, S3 access logs etc
  + Generate business reports on data stored in S3
  + Analyze AWS cost and usage reports
  + Run queries on click-stream data

## AWS Macie

* Uses **AI** to analyze data in S3 and helps identify PII
* Analyze cloud trail logs for suspicious API activity
* Includes dashboards, reports and alerting
* Great for PCI-DSS compliance and preventing ID theft

# Compute

## EC2 Elastic Compute Cloud

* Web service that provides resizable compute capacity in the cloud. Server in the cloud
* When setting up an instance you select which subnet you’d like to place your EC2 instance in and each subnet is tied to a specific AZ. You cannot move an instance between Availability Zones, without setting up a copied version of the instance.
* Whilst they exist in Regions, they are not portable across the whole region, nor across the whole globe
* Underlying hypervisors (virtual machine monitor or VMM, is software that creates and runs virtual machines (VMs). A hypervisor allows one host computer to support multiple guest VMs by virtually sharing its resources, such as memory and processing.) are xen and nitro.
* Instance Metadata and User Data can be retrieved from within the instance via a special URL example. http://169.254.169.254/latest/meta-data/local-ipv4 and [**http://169.254.169.254/latest/meta-data/public-ipv4**](http://169.254.169.254/latest/meta-data/public-ipv4)**.**
* IAM Roles can be attached to instances in the stopped or running state or replaced for instances in the running state.

**Pricing**

* **On Demand** - Fixed rate, by the hour/sec with no commitment. When you want low cost and flexibility, best for short term and testing.
* **Reserved** - 1-to-3-year commitment extreme discount. Different types that give you discounts up to 75%. Reserved Instances **are not physical instances, but rather a billing discount applied to the use of On-Demand Instances** in your account.
  + **3 Types**:
    - **Standard reserved instances**: Offer up to 75% off on demand instances. The more you pay up front and the longer the contract, the greater the discount.
    - **Convertible Reserved Instances**: Offer up to 54% off on demand capability to change the RI as long as the exchange results in the creation of reserved instances of greater or equal value.
    - **Scheduled Reserved Instances**- enable you to purchase capacity reservations that recur on a daily/weekly/monthly basis. For stuff that runs on a schedule
* **Spot** - server auto spins up and shuts off based on price, like stock market, applications that are only feasible at very low compute prices.  (bitcoin mining?)
  + **Spot blocks stop your instance from being terminated even when price goes over your max**. Can be set between **1 to 6 hours** currently.
  + Once instance goes over spot price you have **2 mins** to choose whether or not you should terminate your EC2 instances.
  + Spot fleet is a collection of spot instances and optionally, on demand instances.
* **Dedicated Hosts** - Physical EC2 server dedicated for your use. For regulatory and licensing that require dedicated server, stupid oracle.
  + The **tenancy of an instance can only be changed between variants of ‘dedicated' tenancy hosting. It cannot be changed from or to default tenancy hosting**. Type **options are dedicated and host**
  + You can change the instance tenancy attribute of a VPC from dedicated to default. Modifying the instance tenancy of the VPC does not affect the tenancy of any existing instances in the VPC. The next time you launch an instance in the VPC, it has a tenancy of default, unless you specify otherwise during launch. You can modify the instance tenancy attribute of a VPC using the AWS CLI, an AWS SDK, or the Amazon EC2 API only.
* Can run custom scripts via **USER DATA**
* **To retrieve instance metadata, use** [**http://169.254.169.254**](http://169.254.169.254)**/latest/meta-data**
* **To retrieve instance userdata use** [**http://169.254.169.254**](http://169.254.169.254)**/latest/userdata**
* This metadata includes public and private IP apparently
* Snapshot is a photograph of the disk
* Bootstrap scripts run when the instance is first created

**Placement Groups**

* Way of placing ec2 instances
* Name must be unique
* Supports only the following instance types: compute optimized, GPU, Memory optimized, storage optimized)
* Three types:
  + - 1. **Clustered placement Group:** a grouping of instances **within a single availability zone.** Placement groups are **recommended for apps that need low network latency, high network throughput, or both. Think close together**
      2. **Spread placement group:** group of instanced on distinct underlying hardware. Recommended for apps that have a small number (up to 7)critical instances that should be kept from each other**. Think individual instances.** Opposite of clustered
      3. **Partitioned:** ec2 divides each group into logical segments called partitions. Amazon ec2 ensures that each partition within a placement group has its own set of racks. Each rack has its own network and power source. No two partitions within a placement group share the same racks, allowing you to **isolate the impact of hardware failure within your app. Think multiple instances HDFS, HBAse and Cassandra**
* Homogenous(same) instances within a clustered placement group recommended
* Cannot merge placement groups
* Can move existing instance into a placement. Must be in a stopped state and you can move or remove an instance using the CLI or SDK but not via console.

**Security**

* Encryption - Must create a snapshot of the EC2 volume. Create a copy of that volume (checking box to enable encryption). Create an AMI of the copied snapshot then redeploy
* ec2config - For windows instances this service sets a random admin password and encrypts instance using the Ec2 public key
* Apps that run on Ec2 retrieve temp creds from the instance metadata

**Amazon Machine Images(AMI)**

Select your AMI based on:

* Region (see Regions and Availability Zone’s)
* Operating system
* Architecture (32-bit or 64-bit)
* Launch permissions
* Storage for the root device volume)
  + **Instance store (Ephemeral Storage)**
  + EBS backed Volumes

All AMI’s are backed by either:

* Amazon EBS – The root device for an instance launched from the AMI is an Amazon EBS volume created from an EBS snapshot. Can be stopped and no data lost. Can keep root device volume
* Amazon Instance Store – The reboot device for an instance launched from the AMI is an instance store volume created from a template stored in Amazon S3. Cannot be stopped only terminated or rebooted if the underlying host fails you will lose your data.

**EC2 Hibernate**

* When the OS is told to perform hibernate (suspend the disk)
* The instance boots much faster. The OS doesn’t need to reboot because the in-memory state (RAM) is preserved on persistent storage EBS
* Useful for long running processes and services that take time to initialize
* Root volume must be encrypted to hibernate
* Instance RAM must be less than 150GB
* Instance families include C3, C4, C5, M3, M4, M5, R3, R4 and R5
* Available for windows, amazon linux 2 AMI and ubuntu
* Instances cannot be hibernated for more than 60 days
* Available for on-demand instances and reserved instances

Elastic Beanstalk can be used to provision multiple identical environments

**Common exam Q:** If spot price goes over threshold amazon auto terminates and you don’t get charged for partial hour. But if you terminate it yourself you will be charged for the hour

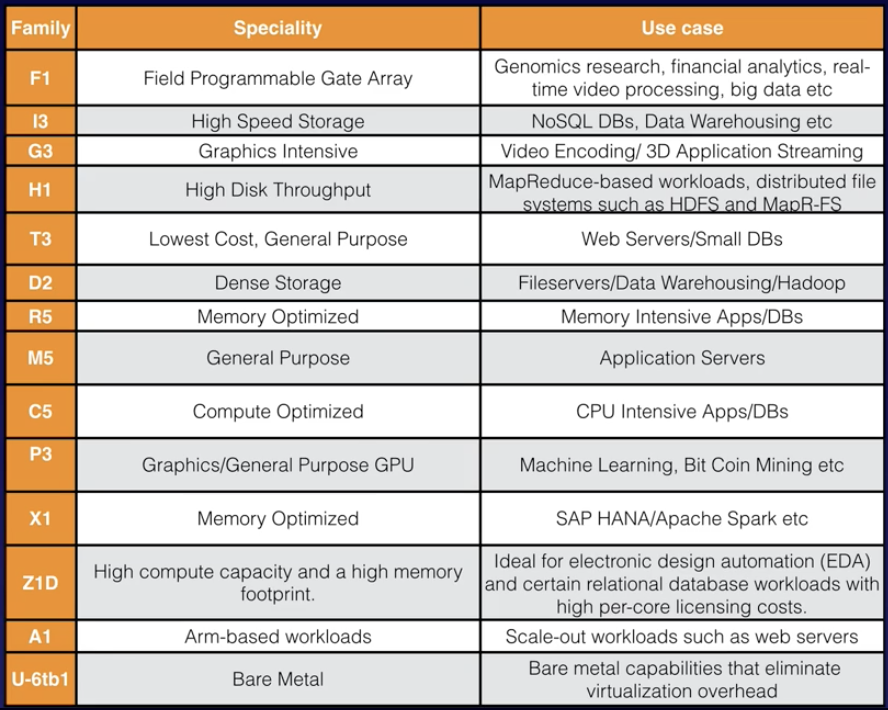
**Troubleshooting:**

* Your ALB can’t connect to Ec2, it times out. The problem is your **security groups**. You need to add a security group rule that allows inbound traffic
* **Detailed** monitoring is available at **1 min intervals** for upcharge **(default mon is 5 min)**

**Security Groups**

* 5 per instance
* All inbound traffic is blocked by default
* All outbound traffic is allowed
* Changes to security groups take effect immediately
* Can have multiple EC2 instances within a security group
* A security group acts as a virtual firewall for your instance to control inbound and outbound traffic.
* **Stateful**
* **Update immediately**
* **Have to use network acls to block specific IP’s**
* **Can specify allow but not deny**
* **You can sell your unwanted reserved instances.**

**Fight DR. mcpxz in au to remember instance types**

  
  
HPC/ High Performance Compute on AWS

Ways to achieve HPC

* Data Transfer
* Compute and networking
* Storage
* Orchestration and automation

**Ways to get our data into AWS**

* Snowball, snow mobile (terabytes/petabytes worth of data
* AWS dataSync to store on S3, EFS, FSx for windows etc.
* Direct connect a dedicated network connection from on premise to AWS

**Compute and network services to achieve HPC**

* EC2 instances that are GPU or CPU optimized
* Ec2 fleets (spot instances or spot fleets)
* Placement groups (cluster placement groups)
* Enhanced networking
* Elastic network adapters
* Elastic fabric adapters

Storage services to achieve HPC

* Instance attached storage
  + EBS: scale up to 64,000 IOPS with provisioned IOPS (PIOPS)
  + Instance Store: scale to millions of IOPS; low latency
* Network Storage
  + S3: distributed object based storage not a file system
  + EFS: Scale IOPS based on total size, or use provisioned IOPS
  + FSx for lustre: HPC-optimized distributed file system, millions of IOPs, which is also backed by S3

Orchestration and automation to achieve HPC

* AWS Batch: enables engineers to efficiently run hundreds of thousands of batch computing jobs on AWS. Supports multi-node parallel jobs which allows you to run a single job that spans multiple EC2 instances. Easily schedule jobs and launch ec2 instances according to your needs.
* AWS ParallelCluster: Open-source cluster management tool that makes it easy for you to deploy and manage HPC clusters.

# Security

## WAF - Web Application Firewall

Summary:

* Web application firewall that lets you monitor HTTPS and HTTPS requests that are forwarded to CloudFront, an application load balancer and API gateway
* Protection for common web attacks
* Configure Web ACLs (Access Control Lists)
* Three different behaviors:
  + - 1. Allow all requests except the ones you specify
      2. Block all requests except the ones you specify
      3. Count the requests that match the properties you specify
* Can define conditions by using characteristics of web requests such as
  + IP addresses
  + Country of origin
  + Values in request header
  + Strings in request or string that match regex pattern
  + Length of request
  + Presence of SQL (SqL injection)
  + Presence of script (cross-site scripting)
* Use WAF or network AcLs to block malicious IP addresses

AWS Shield gives DDOS protection

# Logging

## CloudWatch

Summary

* **Monitoring** and management service built for developers, sys-ops, site reliability engineers, and IT managers.
* Can monitor most of AWS as well as your apps that run on AWS
* **Performance monitoring**
* Monitoring service like:

1. Compute
   1. EC2 instances: monitors every 5 mins by default but can have 1 min intervals by turning on detailed monitoring
   2. Autoscaling Groups
   3. Elastic Load Balancers
   4. Route53 Health Checks
2. Storage & content delivery
   1. EBS volumes
   2. Storage Gateways
   3. CloudFront

* EC2, DynamoDB, RDS and any log files you generate
* Main focus in monitoring/automation/debugging
* Management Service

Cloudwatch Dashboards

* Create awesome dashboards to see what is happening with your AWS environment

Cloudwatch Logs

* Log Expiration policies - **Default** is set to **never expire**, but can set retention between **20 years** and **one day**
* Logs Agent - Supports IAM policy, is how you monitor logs on EC2

**Cloudwatch Events**

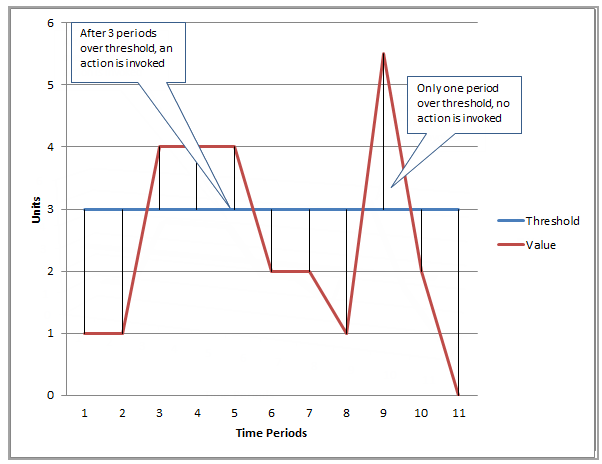
* **Describes changes in AWS Resources**
* Can set up rules to configure notifications, i.e. **email on CodeCommit**
* Difference between this and Cloudtrail is - this is near real-time and you can set rules that perform automated actions. CloudTrail just delivers logs of API calls

CloudWatch Metrics

* Host level metrics consists of:
  + CPU
  + Network
  + Disk
  + Status Check
* Standard Resolution Metrics - Default: CPU, Network, Disk, Status Check
  + **RAM Utilization is a custom metric**
* High -Resolution Metrics (cost more):
  + Stores it with **resolution** of **one second**
  + Can retrieve/read metric with periods of 1, 5, 10, 30, 60+
* Custom Metric - minimum granularity can have is **1 minute,** minimum **resolution is 1 second**
* Get metric data using GetMetricStatistics API
* Dimensions - In custom metrics you can classify what it is using dimensions {key, value}
* Can’t delete metrics

Cloudwatch Alarms

* Triggers notifications
* Standard Resolution Metrics - 60+
* High Resolution Metrics - **10** or **30** seconds
* Can take actions such as sending email, terminating EC2 instance, or ASG policy
* Pick Metric -> Pick Evaluation period (5 min - 60 min) -> Pick value to measure (avg/max) ->Set a threshold



## CloudTrail

Summary

* Monitor your AWS deployments in the cloud by getting a history of AWS **API** calls for your account, including API calls made via the AWS Management **Console**, the AWS **SDKs**,**CLI**, and higher-level **AWS services**
* Identify Users, Accounts, IPs, Timestamps
* Define S3 bucket for storage
* Can define workflows that execute when security events are detected
* Troubleshoot security and operational issues
* **Auditing** changes to services
* Enables governance, compliance
* Monitor **Authorization** attempts

Can view up to past 90 days just by turning it on, by default logs are stored indefinitely.

By default encrypted with S3 SSE, can also enable MFA for deletion

No cost for management events, but data events do cost. Data events are operations performed on the resource itself

## DynamoDB

Summary:

* Serverless
* Stored on SSD
* Conditions on IAM Policy to restrict data that isn’t your own using dynamodb: Leading Keys
* Stored across 3 geographically distinct data centres
* Eventually consistent reads (default)
* Strongly consistent reads (less than one second)
* Allows storing binary objects but there is a limit of 400KB

LSI (Local Secondary Indexes)

* Has to be created as you’re creating table
* Has same primary key but can have different sort key

GSI (Global Secondary Indexes)

* Use these when you want to query by something that isn’t the partition key
* Keys in a GSI don’t need to be unique
* The tradeoff to projecting every attribute into an index is the cost to store and write
* If you need to access most of the non-key attributes often project the entire base table, but storage cost will double

Limits:

* 256 tables per region

You can store session state data on both DynamoDB and ElastiCache

## Non-relational vs. relational databases

* OLTP(online transaction processing): example of pulling specific data(1 record)
* OLAP(online Analytics Processing): more complicated example querying data to find net profits (multiple records/ big data set)

`

## RDS - Relational Database Service

Summary: It is Online Transaction Processing **(OLTP**). i.e A relational database.

* Runs on VM’s (cannot SSH into)
* Cannot log into these OS and patching of RDS responsibility of AWS
* I/O may be briefly suspended while the backup process initializes (typically under a few seconds), and you may experience a brief period of elevated latency.
* The easiest way to find if an error has occurred. Error node in the response from the Amazon RDS API
* You can create Amazon RDS for SQL Server database instances with up to 16TB of storage. The 16TB storage limit is available when using the Provisioned IOPS and General Purpose (SSD) storage types.
* Data transferred between Availability Zones for replication of Multi-AZ deployments is free
* When applying changes to the backup window, you can choose either to have the changes done during the next scheduled maintenance window or immediately.

**Types**

* SQL Server
* Oracle (gross)
* MySQL Server
  + Default endpoint port 3306
* PostgreSQL
* Aurora (Amazon’s SQL database technology)
* MariaDB

**Back Ups**

* **Automated-** enabled by default. can recover db any time in your retention period 1 - 35 (default 7) days. These backups are stored on S3, you automatically get a bucket = to your db size
* **Database Snapshots** - Done manually, stored even after RDS instance is deleted. (think of someone taking a photo with a polaroid camera)
* Restored db will be new RDS instance with new DNS endpoint.

**Encryption**

* Encryption at rest is **supported using AWS Key Management**.
* Encryption at rest is **supported for MySQL, Oracle, SQL Server, PostgreSQL, MariaDB & Aurora**
* Everything will be encrypted from snapshots to backups to db.
* Can’t encrypt existing db, **must take snapshot then encrypt that**. Then deploy a new RDS instance

Two Key Features: Multi-AZ & Read replicas

**Multi-AZ**

* **For Disaster Recovery ONLY**
* **Allows you to have an exact copy of your prod database in another AZ.**
* Available for all SQL dbs in AWS (Built into Aurora by default)
* Any changes to db, its “backup” will also get these changes
* If for any reason main db goes down it auto fails over to backup. DNS addr updates automatically
* Minimal downtime, a minute or so
* Can force a failover from one AZ to another by rebooting RDS instance
* Available for SQL server, Oracle, MySQL Server, PostgreSQL, MariaDB

**Read Replicas**

* **Meant for performance improvement**, allows read only replicas to be created
* When db is written to, then it replicates across its replicas.
* Benefit is you can have some of your EC2 instances point to the replicas for just reading
* This is scaling out your database (Up to 5)
* You can get read replica of a read replica (mind-blown)
* Available for all RDS except SQL Server and Oracle
* Each Read Replica has its own dns endpoint
* Can be promoted to master but this will break the read replica
* Can have a read replica in a separate region
* Available for Oracle, MySQL Server, PostgreSQL, MariaDB, Aurora

**Monitoring**

* Enhanced monitoring: you can monitor the operating system of your DB instance in real time.
* RDS delivers the metrics from Enhanced Monitoring into your Amazon CloudWatch Logs account.

You never deal with IPs with RDS, always DNS. This is important for disaster recovery.

Potential question: How to improve performance: Use ElastiCache.

Potential question: Troubleshooting - You have an ec2 instance and you have an instance of RDS. They are in two different security groups by default. So the ec2 instance won’t be able to talk to RDS. You must go to the RDS security group and allow inbound connection on 3306 from your webserver security group.

Potential question: When to use Read Replica vs Multi A-Z

## Aurora

Summary: MySQL and PostgreSQL – compatible relational database engine that combines the speed and availability of high-end commercial databases with the simplicity and cost-effectiveness of open-source databases

* Aurora serverless provides a relatively simple, cost effective option for infrequent, intermittent, or unpredictable workloads.
* 2 copies of your data are contained in each AZ, with min of 3 AZ’s. 6 copies of your data
* You can share Aurora snapshots with other AWS accounts
* 3 types of replicas:
  + - 1. Aurora Replicas has automated failover
      2. MySQL replicas
      3. PostgreSQL replicas
* Automated backups by default
* Use Aurora serverless if you want simple, cost-effective option for infrequent, intermittent or unpredictable workloads
* Using endpoints, you can map each connection to the appropriate instance or group of instances based on your use case.
* Amazon Aurora typically involves a cluster of DB instances instead of a single instance. Each connection is handled by a specific DB instance.

Graphical user interface, application

Description automatically generated

## RedShift (OLAP)

Summary: used for business intelligey6tnce or data warehousing

* Online analytics processing
* Fast and powerful, fully managed, petabyte-scale data warehouse service in the cloud.
* Available in one AZ
* Back-ups enable by default with 1 day retention period max 35 days
* Redshift always attempts to maintain 3 copies of your data (the original and replica on the compute nodes and a backup in Amazon S3)
* Can asynchronously replicate your snapshots to S3 in another region for disaster recovery.
* Curveball, **Redshift** could be the answer to free up a strained database IF the reason for the db stress is because you keep running OLAP (analytics) transactions. Data warehousing.

## ElastiCache

Summary - Web service that allows an **in-memory cache** in the cloud. **Used for speeding up applications**. A way of caching frequent queries from database. Improve latency and throughput for read heavy applications.

* Use elastiCache to increase database and web application performance

Types:

* **Memcached** - If no redundancy is necessary use Memcached. Pure caching, no persistence.
  + **Simpler**
  + Caching is primary goal
  + Scale out (horizontal)
  + Not Multi-AZ
* **Redis** - open source in memory key value store.
  + **Use this if you want Multi-AZ redundancy**.
  + **Back-ups and restores**
  + Managed as a relational db, is stateful
  + Advanced datatypes (lists, hashes, sets)
  + Sorting and ranking datasets (leaderboards)
  + Persistence
  + Across multiple AWS availability zones
  + Image below got info wrong

**Graphical user interface

Description automatically generated**

## EMR - Elastic Map Reduce

**Summary:**

* A container service that allows you to run and scale Spark, **Hadoop**, HBase, Presto, Hive and other **BIG DATA** frameworks
* Input data in S3
* Uses Ec2 Instance(s) for processing
* Run petabyte scale analysis at less than half the cost of traditional on-prem solutions and over three times faster than standard Apache Spark.

Nodes:

* Consists of a master node, core node and optionally a task node
* You can configure a cluster to periodically archive the log files stored on the master node to Amazon S3. This ensures the log files are available after the cluster terminates, whether this is through normal shutdown or due to an error. Amazon EMR archives the log files to S3 at 5 min intervals. **(Can only be configured when creating the cluster for the first time)**

\*\* Any questions about big data think about EMR

## Database Migration Service (DMS)

**Summary:**

* Allows you to migrate databases from one source to AWS
* Source can be on-prem, AWS or another cloud provider

**Types:**

Homogenous migrations:

* Example Oracle to Oracle
* On-prem database > ec2 instance running DMS > RDS

Heterogeneous migrations:

* Microsoft SQL server to AWS Aurora
* Need AWS schema conversion tool
* On-prem database > EC2 instance running DMS & SCT > RDS

\*\* if the databases are the same you do not need SCT

## Caching Services

* Caching is a balancing act between up-to-date, accurate info and latency.

Services with caching capabilities:

* CloudFront
* API Gateway
* ElastiCache-memcached and redis
* DynamoDB Accelerator (DAX)

## Route 53

Summary

* DNS Web Service
* Functions: Domain registration, **DNS routing**, and health checking
* Replicates some ELB functionality

Domain Registration

* You need to provide names and contact information for the domain
* Registrar is Amazon Registrar or Gandi

DNS Routing

* Routing Policies:
  + **Simple -** For a single resource that performs a function for your domain
  + **Failover** - For **active-passive** failover
  + **Geolocation** - To route traffic based on **location of your user**
  + **Geoproximity** - Route traffic based on **location of your resources**, and shift resources from one location to another optionally
  + **Latency** - When you have resources in multiple regions and want to route traffic to region with best latency
  + **Multivalue answer** - Makes route 53 respond to queries with up to 8 healthy records…. Selected at random
  + **Weighted routing policy** - Route traffic to multiple resources in proportions that you specify

Other Info

Data in a region is not replicated outside of that region, we must do that ourselves.

Networking

## ENI (Elastic Network interface) EN (Enhanced Networking) EFA (Elastic Fabric Adapter)

**ENI: Elastic Network interface**

* essentially a virtual network card on ec2 instance
* automatically attached to instance
* for basic networking
* You can attach a network interface to an instance when it's running (**hot attach**), when it's stopped (**warm attach**), or when the instance is being launched (**cold attach**).
* **when you need a separate management network to your production network** or separate logging network **but at a low cost**. Use multiple ENI’s for each network
* a primary private IPv4 address range of your VPC, one or more secondary private ipv4 addresses from the IPV4 address range of your VPC.
* Used to Create a management network, create a low budget highly available solution

**EN (enhanced networking)**

* uses single root I/O virtualization (SR-IOV) to provide high performance networking capabilities on supported instance types
* Essentially just speeds up networking
* For when you need speeds between 10Gbps and 100Gbps. Anywhere you need reliable, high throughput.
* Enhanced networking can be enabled using:
  + - 1. **Elastic Network Adapter** (**ENA**), which supports network speeds up to 100 Gbps for supported instance types.
      2. Intel 82599 **Virtual Function** (**VF**)interface, which supports network speeds up to 10 Gbps for supported instance types. (Typically used on older instances)

\*\*\* ALWAYS choose ENA when asking how to speed up networking

**EFA (Elastic Fabric Adapter)**

* is a network device that you can attach to your EC2 instance to accelerate high performance computing (HPC) and machine learning applications.
* Provides lower and more consistent latency and higher throughput

\*\*\* ALWAYS choose ENA when asking how to speed up networking

\*\*\* Whenever there is a question about ENI vs. ENA vs EFA and they’re talking about HPC and machine learning, then you want to choose an Elastic Fabric Adapter.

## Amazon VPC (Virtual Private Cloud)

* Think of it as a virtual data center in the cloud
* Let’s you provision a logically isolated section of the AWS cloud where you can launch AWS resources in a virtual network that you define. You have complete control of your networking environment, including section of your own IP address range, creation of subnets and configuration of route tables and network gateways.
* Example: can create a public-facing subnet with your webservers that have access to the internet and place backend systems such as databases or application servers in a private-facing subnet with no internet access. You can leverage multiple layers of security groups and network access control lists, to help control access to Amazon EC2 instances in each subnet.
* Network ACLS are stateless (has no saved state) (they allow you to do “allow” rules and “deny” rules. Firewall of subnets
* Your VPC automatically comes with a modifiable default network ACL
* **Security Groups are stateful** (they store state) and act as the first line of defense. Firewall of EC2 instances
* Bastian host is an EC2 instance in a public subnet
* 1 subnet = 1 availability zone
* **Can only have 1 internet gateway attached per VPC**
* Keep main route table private and create another route table for public
* Is a security group a firewall?
* AWS releases your instance's public IP address when it is stopped
* If your instance does not have a public IPv4 address, you can associate an Elastic IP address with your instance to enable communication with the internet.
* **5 VPCs per AWS** account but can request an increase
* AWS customers are welcome to carry out security assessments or penetration tests against their AWS infrastructure without prior approval for 8 services only ec2, rds, cloudfront, aurora, api gateway, lambda, lightsail, elastic beanstalk
* In a custom VPC with new subnets in each AZ, there is a route within the route table that supports communication across all subnets/AZs. Additionally, it has a Default SG with an "allow" rule: all traffic, all protocols, all ports, from resource using this default security group.
* An Elastic IP address is a public IPv4 address, which is reachable from the internet.

Table

Description automatically generated

## What can we do with a VPC?

* Launch instances into a subnet of your choosing
* Assign custom IP address ranges in each subnet
* Configure route tables between subnets
* Create internet gateway and attach it to our VPC
* Much better security control over your AWS resources
* Instance security groups
* Subnet network access control lists (ACLS)

## Default vs Custom VPC

* Default vpc is user friendly allowing immediate deploy instances
* All subnets in default VPC have a route out to the internet
* Each EC2 instance has both a public and private IP address

## Setting up a VPC

1. Internet Gateway and Virtual Private Gateway goes to our Router in our VPC.
2. Our router then directs traffic to our different route tables
3. Route tables direct traffic through our network ACLs (first line of defense and act just like firewalls very much like security groups
4. Next is Security groups which are the first line of defense
5. Subnets public means they can access the internet and private means it has no access to the internet

**Example**

Graphical user interface

Description automatically generated

**Nat Instances**

* Nat instances are individual ec2 instances that
* Nat instances act as a bridge between our internet gateway and private subnets
* Have to disable source and destination check on the instance
* Must be in a public subnet and must be a route of the private subnet to the NAT instance, in order for this to work.
* The amount of traffic that NAT instances can support depends on the instance size. If you are bottlenecking, increase the instance size
* Can create high availability using autoscaling groups, multiple subnets in different AZ’s, and a script to automate failover.
* Behind a security group

**Nat Gateway**

* Nat gateways are highly available not just one ec2 instance that spreads across multiple AZs and are not dependent on a single instance.
* sends and retrieves traffic on behalf of instances in a private subnet. As a result, source/destination checks on the NAT instance must be disabled to allow the sending and receiving traffic for the private instances.
* Nat gateways are gateways that allows you to let your private subnets communicate out to the internet without being public
* Redundant inside the AZ
* Can only have one per AZ
* Preferred by enterprise
* Scales automatically
* No need to patch
* Auto assigned ip address
* If you only have one Nat gateway shared by resources in multiple AZ’s if that resource goes down all of the resources will lose internet access. Best to create a Nat gateway in every AZ and configure routing to ensure resources use the Nat gateway in the same AZ they are in

**Nat instances & Nat Gateways**

* You have to add the nat gateway or instance to your route table in order

**IP Ranges**

* 10.0.0.0 (10/8 prefix) is for private IP’s only
* 172.16.0.0 (172.16/12 prefix) for private only
* 192.168.0.0 (192.168/16 prefix) private only and see this mainly for home networks
* Cidr.xyz
* 10.0.0.0/16 is the largest subnet you can use with VPC and the largest used across corporate
* Private IPs are not really unique the same IP could be used for other companies
* /24 you can control the 4th number in the IP from 1-254
* /16 you can control the 3rd and 4th number from 1-254
* /8 allows you to control the most range 2nd, 3rd and 4th number from 1-254
* The /32 denotes one IP address and the /0 refers to the entire network.

**Ports**

* Take note that the SSH protocol uses TCP and port 22.

**VPC Peering**

* Allows you to connect one VPC with another via a direct network route using private IP addresses
* Instances behave as if they were on the same private network
* You can peer VPC’s with other AWS accounts
* No transitive peering.

**Network Access Control List (Network ACL’**

* Done in chronological order (starting with lowest numbered rule)
* Evaluated before security groups
* VPC automatically comes with network ACL and allows all inbound and outbound traffic
* You can create custom network ACLs. By default, each custom network ACL denies all inbound and outbound traffic until you add rules
* **Each subnet in your VPC must be associated with a network ACL.** If you don’t explicitly associate a subnet with a network ACL, the subnet is automatically associated with the default network ACL
* Block IP addresses
* Can associate a network ACL with multiple subnets but a subnet can be associated with only one network ACL. When you associate a network ACL with a subnet, the previous association is removed
* Separate inbound and outbound rules and each rule can either allow or deny traffic
* Network ACLs are stateless, responses to allow inbound traffic are subject to the rules for outbound traffic (and vice versa)
* Use to allow or deny certain IPs or ranges of Ips into your subnet using inbound and outbound rules.
* Can also deploy a host-based firewall

**Security groups**

* By default, a security group includes an outbound rule that allows all outbound traffic.

**VPC Flow Logs**

* A feature that enables you to capture information about the IP traffic going to and from network interfaces in your VPC.
* Stored using Cloudwatch Logs or S3
* **Created at 3 levels**
  + **VPC**
  + **Subnet**
  + **Network Interface Level**
* Cannot enable for peered VPCs unless they’re in the same account
* Once created cannot change not even IAM role
* Traffic not monitored
  + Traffic generated by instances when they contact Amazon DNS service
  + Traffic generated by a windows instance for Amazon windows license activation
  + **Traffic to and from 169.254.169.254 for instance metadata**
  + DHCP traffic
  + Traffic to the reserved IP address for the default VPC router

**Bastions**

* A bastion host is a special purpose computer on a network specially designed and configured to withstand attacks.
* A way to SSH or RDP into private instances in your private subnets and you just focus on hardening them.
* A NAT gateway or NAT instance is used to provide Internet traffic to EC2 instances in a private subnet.
* A bastion is used to securely administer EC2 instances (using SSH or RDP). Bastions are called jump boxes in Australia   
  Cannot use a NAT gateway as a bastion host.

**Highly availability with Bastion Hosts**

* Two hosts in two separate AZ’s. Use a network load balancer with static IP addresses and health checks to fail over from one host to the other
* Can’t use an application load balancer, as it is layer 7 and you need to use layer 4
* One host in one AZ behind an auto scaling group with health checks

**VPC endpoint**

* Enables you to privately connect your VPC to supported AWS services and VPC endpoint services powered by PrivateLink without requiring an internet gateway, NAT device, VPN connection or AWS Direct connect
* Virtual devices that are horizontally scaled, redundant and highly-available VPC components that allow communication between instances in your VPC and services without imposing availability risks or bandwidth constraints on your network traffic
* Two Types
  + Interface endpoints
  + Gateway endpoints: supported for Amazon S3 and DynamoDB

**Default VPC**

* When you create a VPC a **default Route Table, Network Access Control List (NACL) and a default security group.**
* It won’t create any subnets or a default internet gateway
* AZ’s are randomized. US-East-1a could be completely different in each account

Graphical user interface

Description automatically generated

**AWS Network Cost**

* **Use private IP addresses to save on cost** and then utilize the AWS backbone network
* To cut all net work costs**, group ec2 instances in the same AZ** **and use private IP’s** this is cost free but keep in mind single point of failure issues.

## AWS VPN CloudHub

* Hub-and-spoke model
* If you have multiple sites with it’s own VPN connection, you can use AWS VPN CloudHub to connect those sites together
* Low-cost, easy to manage
* It operates over the public internet, but all traffic between the customer gateway and the AWS VPN CloudHub is encrypted

## AWS VPN

* Extend your on-premises networks to the cloud and securely access them from anywhere
* establish secure connections between your on-premises networks, remote offices, client devices, and the AWS global network. AWS VPN is comprised of two services: AWS Site-to-Site VPN and AWS Client VPN.

## AWS PrivateLink

* Allows you to open your services in a VPC to another VPC using privateLink
* **If you see a question asking about peering VPC’s to tens, hundreds, or thousands of customer VPCS, think of AWS PrivateLink (most efficient)**
* Doesn’t require VPC peering: no route tables, Nat, IGWs, etc.
* **Requires a Network load balancer on the service VPC and an ENI on the customer VPC**

## AWS Transit Gateway

* Allows you to have **transitive peering** between thousands of VPC’s and on-premises data centers.
* Works on **a hub-and-spoke model**
* Works on a regional basis but you can have it across multiple regions
* You can use it across multiple AWS accounts using RAM (Resource Access manager)
* You can use rout tables to limit how VPC’s talk to one another
* Works with direct connect as well as VPN connections
* **Supports IP multicast (not supported by any other AWS service)**

## AWS CloudFront

* Content delivery network (CDN) a system of distributed servers (network) that deliver webpages and other web content to a user based on the geographic locations of the user, the origin of the webpage and a content delivery server.
* Edge locations downloads files and cache them locally
* CloudFront checks its cache for the requested files

Terminology

**Edge Locations:** the location where content is cached and can also be written.(closer to the end user) Separate to an AWS Region/AZ. Not just Read only you can write to them too Also used for reducing latency for S3 uploads. Objects are cached for the life of the TTL (Time to live) which is adjustable.

**Origin:** This is the origin of all the files that the CDN will distribute. Typically a s3 bucket, ec2 instance, an elastic load balancer or Route 53.

**Distribution:** This is the name given the CDN which consists of a collection of Edge Locations.

**Web distribution:** Typically used for websites

**RTMP**: used for Media Streaming and adobe

1 file = 1 signed url

1 cookie = multiple files

Signed URL/ cookies to secure content

**Signed URL:**

1. Can have different origins
2. Key-pair account wide and managed by root user
3. Can utilize caching features
4. Can filter by date, path, IP address, expiration and etc
5. Doesn’t access S3 directly
6. If origin ec2 use cloudfront

Summary

* Speeds up distribution of your static and dynamic web content
* Data, Videos, Applications, and APIs
* Objects are cached with a TTL
* Protocols supported: HTTP, HTTPS, RTMP (Real-Time Messaging Protocol)
* RTMP is used for video streaming via cloudfront
* Edge location is a thing, not the same as AZ, where your content is cached
* Can use signed URL/Cookie to restrict it
* Origin is the origin of all the files that CDN will distribute can be s3 bucket, ec2 instances, elastic load balancers or route53
* Can invalidate cached objects but you will be charged
* Can use signed URL/Cookie to restrict it

## Route 53

Summary:

* DNS Web Service
* DNS is used to convert human friendly domain names
* Functions: Domain registration, **DNS routing**, and health checking
* Replicates some ELB functionality’
* ELB’s do. Not have a pre-defined IPv4 address, you resolve to them using a DNS name
* **Alias record vs CName pick an alias record!** The CNAME record maps a name to another name. ... The ALIAS record maps a name to another name but can coexist with other records on that name.
* Common DNS types: SOA records, NS records, A records, CNames, MX records, PTR records
* Can take up to 3-days to register domains
* The limit is 20 for new customers as of March 2021. If you have an existing account and your default limit is 50 now, it will remain at 50.

Domain Registration

* You need to provide names and contact information for the domain
* Registrar is Amazon Registrar or Gandi

**DNS Routing**

* Routing Policies:
  + **Simple** –
    - For a single resource that performs a function for your domain
    - You only have one record with multiple IP’s (if you specify multiple values in a record, Route53 returns all values to the user in a random order)
  + **Failover** - For **active-passive** failover. Will route to primary route53 dns entry unless it fails the health check and routes over to the passive site
  + **Geolocation** - To route traffic based on **location of your user**
  + **Geoproximity** - Route traffic based on **location of your resources and users**, and shift resources from one location to another optionally. Must use route53 trafficflow
  + **Latency** - When you have resources in multiple regions and want to route traffic to region with best latency.
  + **Multivalue answer** - Makes route 53 respond to queries with up to 8 healthy records…. Selected at random. Basically simple routing with health checks.
  + **Weighted routing policy** - Route traffic to multiple resources in proportions that you specify 20% to us-east-1 and 80% to us-east-2

Health Checks

* You can set health checks on individual record sets
* If a record set fails a health check it will be removed from route53 until it passes the health check
* You can set SNS notifications for it a health check fails

Other Info

Data in a region is not replicated outside of that region, we must do that ourselves.

## Elastic Beanstalk

Summary

* Service for deploying, managing and scaling web applications and services
* Supports Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker on familiar servers such as Apache, Nginx, Passenger, and IIS
* Makes it so developer don’t have to worry about underlying resources
* Compute Service

Deployments

* **All at once** - default, Deploy the new version to all instances at the same time. All instances are out of service for a short time (**Downtime**)
* **Rolling** - Elastic Beanstalk splits the environment into batches and updates them one batch at a time. Some instances will serve old version and some new until deployment is complete
* **Rolling deployment with an additional batch** - Will launch a new batch of instances prior to taking any instances out of service. Use if you want to maintain full capacity during deployments. After deployment additional batch is terminated
* **Immutable** - Launches full set of new instances running new version alongside old instances. If EC2 instances don’t pass health checks, they get terminated and originals stay. If you want to allow instances passing with lower status you can modify the **Healthy threshold**. Optionally you can override with **ignore health check** option
* Blue/Green - Use if you want to manually do immutable

## ELB - Elastic Load Balancer

Summary – Balances network load across web servers.

* Have their own DNS name not given an IP

**Types:**

* **Application** – **intelligent routing**,Can see all the way up to Application Layer and cleverly route. Suited for HTTP/HTTPS/Websocket traffic. **support path-based and host-based routing.** Very advanced routing (Tesla Model X)
* **Network** - Most expensive, **best/extreme performance**. Suited for routing TCP traffic, can handle millions of requests per second, also allows static IPS. (Tesla Roadster)
* **Classic** - **Legacy**, can load balance at application layer or network layer. However, it’s application load balancing isn’t intelligent, uses X-Forwarding and sticky sessions

**Troubleshooting**

* **504 Gateway Timeout Erro**r – means the app is not responding within the idle timeout period. (Not the load balancer itself having an error) means you need to troubleshoot **webserver** or **db server**.
* 4XX Error - This is a Client-side error
* **Stickiness** will route a user to the first instance they went to. This prevents re- login

**Sticky Sessions**

* Classic Load Balancer routes each request independently to the registered EC2 Instance with the smallest load
* Allow you to bind a user’s sessions to a specific EC2 instance. This ensures that all requests from the user during the session are sent to the same instance
* Can be enabled for application load balancers as well but will be sent at the target group level

**Cross Zone Load Balancing**

* Sending traffic to each EC2 instance across AZ’s evenly

**Path Patterns**

* Create a listener with rules to forward requests based on the URL path.
* Allow you to direct traffic to different EC2 instances based on the URL contained in the request.

Must have at least two public subnets in VPC

User -> Load Balancer -> EC2 Instance

**On server if you need to know user’s public IPV4 address look in the X-Forwarded-For header.**

To get certs can use AWS Cert Manager or generate your own and upload them to Cert Manager or IAM

# Other

## AWS Organizations & Billing

* An account management service that enables you to consolidate multiple AWS accounts into an organization that you create and centrally manage.
* Root account for billing only
* Organizational Units separate account based on how you split the accounts and policies are put in-front of the account
* All accounts are linked but separate. In the end you get one bill for all accounts and that can help save on costs
* Advantages:
  + one bill per AWS account,
  + very easy to track and allocate costs
  + volume pricing discount

**Best Practices**

* Always enable multi-factor authentication on the root account
* Always use a strong and complex password on root account.
* Paying account should be used for billing purposes only. Don’t deploy recourses into the paying account
* Enable/disable AWS services using service control policies (SCP) either on OU or on individual accounts

## AWS Directory Service

Summary:

* Family of managed services
* Connect AWS resources with on-prem ActiveDirectory

**Active Directory (AD) compatible**

* Managed Microsoft AD aka (directory service for Microsoft Active Directory)
* AD Connector
* Simple AD

**Not Active Directory (AD) compatible**

* Cloud directory
* Cognito user pools

## AWS Resource Access Manager

Summary:

* Helps to securely share your resources across accounts, within your org or org units in AWS organizations and with IAM roles and users for supported resource types.
* Share transit gateways, subnets, AWS license manager license configurations, amazon route 53 resolver rules etc.

## AWS Single Sign-On

Summary:

* Centrally manage access to multiple AWS accounts or applications.

\*\*\*\* if you see SAML 2.0 in an exam question look for SSO

## AWS Direct Connect

* A cloud service solution that makes it easy to establish a dedicated network connection from your premises to AWS.
* Make private connection between your datacenter, office or colocation environment and AWS
* Useful for high-throughput workloads (ie lots of network traffic)
* If you need a stable and reliable secure connection
* Steps to creating a direct connect connection
  + - 1. Create a virtual interface in the direct connect console. This is a public virtual interface.
      2. Go to the VPC console then to VPC connections. Create a customer gateway
      3. Create a virtual private gateway
      4. Attach the Virtual private gateway to the desired VPC.
      5. Select VPN connection and create a new VPN connection
      6. Select the Virtual private gateway an the customer gateway
      7. Once the VPN is available, setup the VPN on the customer gateway or firewall

## AWS Global Accelerator

Summary: A service in which you create accelerators to improve availability and performance of your apps for local and global users.

* Directs traffic to optimal endpoints over the AWS global network
* You are assigned two static IP addresses (or alternatively you can bring your own)
* You can control traffic dials. This is done within the endpoint group
* A network zone services the static IP addresses for your accelerator from a unique IP subnet.

## Auto Scaling

Has three components

* + - 1. **Groups:** logical component. Webserver group or app group or database group etc.
      2. **Configuration Templates:** Groups **uses a launch template or a launch configuration as a configuration template for its ec2 instances**.
      3. **Scaling options:** Scaling options provides several ways for you to scale your auto scaling groups. For example, you can configure a group to scale based on the occurrence specified conditions (dynamic scaling) or on a schedule.
* **Maintain current instance levels at all times**
  + Maintain number of instances running at all times
  + Periodic health checks on instance and auto stop and recreate on unhealthy instances.
* **Scale manually**
  + Specify only the change in. the min, max and desired capacity of your auto scaling group
* **Scale based on a schedule:** schedule-based scaling
  + Scale based on time and date
* **Scale based on demand**
  + More advanced
  + Using scaling policies
* **Use predictive scaling**
  + Predicts based on previous performance
  + Proactive and reactive approaches
* **Create scaling policies** 
  + **Simple scaling:** 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 expire before responding to additional alarms.
  + **Target tracking:** you select a scaling metric and set a target value. Amazon EC2 Auto Scaling creates and manages the CloudWatch alarms that trigger the scaling policy and calculates the scaling adjustment based on the metric and the target value.
    - To calculate your backlog per instance, start with the ApproximateNumberOfMessages queue attribute to determine the length of the SQS queue (number of messages available for retrieval from the queue). Divide that number by the fleet's running capacity, which for an Auto Scaling group is the number of instances in the InService state, to get the backlog per instance.
  + **Suspend and resume scaling**: used to temporarily pause scaling activities triggered by your scaling policies and scheduled actions.
* **The default auto scaling policy is to terminate the oldest instance/closest to the next billing hour.**

Auto scaling groups

Summary

* Contains a collection of Amazon Ec2 instances that share similar characteristics
* If an instance is unhealthy it is terminated and replaced
* You can also autoscale other services using the Application Auto Scaling API
* Enabled by **CLOUDWATCH**

Application Auto-Scaling API

* ECS, EMR clusters, Appstream, DynamoDB Capacity, Aurora Replicas, SageMaker
* ASG vs EC2 Auto Scaling - Use EC2 auto scaling if you only want to autoscale EC2 instances.
* Predictive Scaling - Uses historic information about your usage to schedule future usage, uses ML

## Highly Available Architecture

Summary: you should always plan for failure

* Must be able to tolerate the failure of 1 AZ
* Always design for failure
* Scailing out is adding additional ec2 instances
* Scailing up is scaling up ec2 resources. For example going up from a tq.micro to a 6x extra large

## CloudFormation

Summary:

* Templates that allow you to provision cloud resources. collection of sample templates will help you get started with AWS CloudFormation and quickly build your own templates
* Can save as .json, .yml, .template, or .txt
* To makes changes to a running stack use a change set (allows to see what your changes will do)
* Stored in S3

## On-Premises Services with AWS

Services you can use on premises:

* Database migration service (DMS)
  + Move databases to and from AWS
  + Works with oracle, mySQL, dynamoDB etc.
  + Supports homogeneous and heterogeneous
* Server migration service (SMS)
  + Supports incremental replication of your on-prem servers to AWS.
  + Can be used as a backup tool, multi-site strategy (on-premises and off-premises), and a DR tool.
* AWS application discovery service
  + Helps enterprise customers plan migration projects by gathering on prem data
  + Install as a virtual appliance on VMware vCenter
  + Will then build a server utilization map and dependency map of your on-premises environment.
  + This data is retained in. encrypted format in application discovery service data store. You can export this data as a CSV file and use it to estimate the total cost of ownership (TCO) of running on AWS and to plan your migration to aws. The data also available on AWS migration hub
* VM import/export
  + Migrate existing apps in to EC2
* Download amazon Linux 2 as an ISO
  + Works with vmware, hyper-v etc

# SQS - Simple Queueing Service

Summary: pull

* Web service that gives you access to a message queue that can be used to store messages while waiting for a computer to process them.
* three main parts in a distributed messaging system: the components of your distributed system, your queue (distributed on Amazon SQS servers), and the messages in the queue.
* Message oriented queueing service that allows **decoupling of microservices**, distributed systems and serverless apps
* NO ORDER - SQS messages can be delivered multiple times in any order
* Design - you can have 2 priority queues for priority-based message one for higher and other for lower priority
* EC2 instances always poll for messages from the queue (**pull** from the queue and not push)
* Long polling can reduce costs, it **waits for a message to come in before returning**. **Maximum long polling timeout 20 seconds**. This reduces # of empty ReceiveMessageResponse(s) returned
* Short polling returns immediately (even if the message queue being polled is empty)
* Amazon SQS requires you to implement your own application-level tracking, especially if your application uses multiple queues.
* Two queue types:
  + Standard queues (default):
    - Nearly unlimited number of transactions per second.
    - Guaranteed to deliver message at least once
    - Might be delivered out of order and same message out of order
  + FIFO (first-in-first-out) queues:
    - FIFO queues end in .fifo - offer exactly once processing, support multiple producers but not multiple consumers
    - First-in-first-out delivery and exactly once processing
    - Order is strict
    - Message is delivered once and remains available until a consumer processes and deletes it.
    - Limited to 300 transactions per second (TPS) but have all the capabilities of standard queues.

Limits

* **NO LIMIT** for number of messages in a SQS queue
* MAX of **120,000** in-flight messages (FIFO queue 20,000)
* Message can contain **1-256KB of tex**t, billed at 64KB chunks (For messages bigger than this use SQS Extended Client)
* Single consumers can have **1 to 10** messages unto maximum of 256KB payload
* Messages in the Queue can be retained from **1 min** up to **14 days default is 4days**
* 3,000 messages per second with batching
* Messages can have any format

DLQ - Dead Letter Queues:

* Other queues can put messages here when they aren’t processed successfully
* Main purpose is handling failures

Billing:

* First 1 million requests are free, then $0.50 PER EVERY MILLION REQUESTS

VisibilityTimeout

* It is a period of time where it prevents other consumers for processing the message again
* Visibility timeout always start from when the application instance polled the message.
* Visibility timeout expires that means there is a failure somewhere since that message was polled but not processed and hence not deleted so other some other process will poll the message again and visibility timeout starts again.
* Visibility timeout by **default is 30 Seconds** min is **0 seconds**, up to **12 hour** maximum (ChangeMessageVisibility) / maximum visibility
* ChangeMessageVisibility - Changes the timeout of a message in a queue to a new value.

## SWF - Simple Workflow Service

* You should use this if you require external signals to be a part of your workflow, or you want to launch child processes that return a result to their parent. Otherwise use AWS STEP Functions
* A web service that makes it easy to coordinate work across distributed application components.
* SWF workflow executions can last up to 1 year
* Presents a task-oriented API, whereas Amazon SQS offers a message-oriented API
* Ensures that a task is assigned only once and is never duplicated. With amazon SQS, you need to handle duplicated messages and may also need to ensure that a message is processed only once.
* Keeps track of all the tasks and events in an application. With amazon SQS, you need to implement your own application-level tracking, especially if your application uses multiple queues.

**SWF Actors**

* **Workflow starters** – an app that can initiate (start) a workflow. Could be an e-commerce website following the placement of an order, or a mobile app searching for bus times.
* **Deciders** – control the flow of activity tasks in a workflow execution. If something has finished or failed in a workflow a decider decides what to do next.
* **Activity works** – carry out the activity tasks

## SNS - Simple Notification Service

Summary: push

* Pub/Sub messaging for microservices, distributed systems and serverless apps
* SNS can notify to Email, Text / SMS, SQS or any HTTP end point (and lambda functions).
* Lower cost that SES

Tips:

* Using AWS PrivateLink you can publish messages without traversing the internet
* CloudWatch or Autoscaling triggers SNS
* protocols: HTTP, HTTPS, EMAIL, EMAIL-JSON, SQS or Application - messages can be customized for each protocol
* SNS messages are stored redundantly to multiple AZs
* SNS Dataformat - JSON (Subject, Message, TopicArn, MessageId, unsubscribeURL etc..)

Cost:

* $0.50 per 1 million SNS request
* Different price for different recipient types
* to HTTP: $0.06 / 100,000 notifications deliveries
* to EMAIL: $2 / 100,000 notifications deliveries
* to SMS: $0.75 / 100 notifications deliveries

**Fanout** v Filtering

* Fanout - Occurs when a message is sent to a topic and then replicated and pushed to multiple endpoints
* Filtering - Subscribers can filter out messages they don’t want/need

## Elastic Transcoder

* Media transcoder in the cloud
* Converts media files from original source format in to different formats that will play on smartphones, tablets, PCs, etc.

## API Gateway

Summary

* Can import API defs using Swagger syntax
* You can use it for SOAP passthrough
* CORS enforced by the client
* Low cost and auto scales
* Can throttle to prevent attacks
* Can log to cloud watch
* If using Javascript/ajax that uses multiple domains with API gateway, ensure that you have enabled CORS on API Gateway.
* Cors enforced by the client.

Logs

* Must grant API Gateway IAM permissions to read and write to CloudWatch

Caching – **This increases performance. Get requests, controlled by TTL, 0-3600. Client can invalidate using Header** **Cache-Control: max-age=0**

## Kinesis

Summary

* Platform to send streaming data
* Collect, process, and analyze video and data streams in real time
* Batch records to save cost increase throughput
* Retention period for data record is 24 hours after creation up to 7 days
* Uses AWS KMS for encryption

Terminology:

* Shard - Sequence of data streams (1,000 MB/s)
* Kinesis Data Stream - A set of shards, each shard has a sequence of data records
* Data record - Seq #, partition key, data blob

Kinesis Streams

* Accepts data from data producers stores in shards
* Then data consumers (EC2) do some processing on the data
* Then send data to be stored somewhere
* Shards store streams

Kinesis Firehouse

* Same as Streams except there’s no shards, streams, or consumers. This is all automated. Analyzed using Lambda
* No retention, in and out to S3.

Kinesis Analytics

* Can run SQL queries off data from Firehouse or Streams
* Analyzes

Increase number of shards and allow partition key to take many attributes **in order to scale**

## Cognito

Summary:

* Amazon Cognito lets you add user sign-up, sign-in, and access control to your web and mobile apps quickly and easily
* Can use Google, FB, Amazon using AD and SAML

User Pools (Directory):

* Sign Up and Sign in directly, or through identity providers
* A directory for all your users, a UI for users to sign up and store profiles
* User tokens are returned to your app
* Token handling for your app is provided by SDK
* Allows manually decoding and verifying JWT tokens as well
* Successful auth generates **JWT**

Identity Pools (Federated Identities)

* **Roles** and permissions
* Control access to AWS resources to control access.
* Can give different **permissions** and even **temporary** AWS credentials
* Unique ID for users and authenticate them with web ID providers??

Fed Auth types

* **Setting up SAML 2.0-Based Federation by using a Web Identity Federation** is used primarily to let users login with Amazon, facebook, google etc
* **Setting up SAML 2.0-Based Federation by using a Microsoft active** directory federation service (AD FS)

Cognito Sync

* Enables cross-device syncing of application-related user data
* Uses **SNS** to do this

## Event Processing patterns

* Dead letter Queues
  + SNS: messages published that fail
  + SQS: messages that exceed request limit
  + Lambda: queue of failed executions
* S3 event notifications: receive notifications based on bucket events

# KMS - Key Management Service

Summary:

* Used to create and control encryption keys
* KMS allows you to centrally manage keys known as customer master keys (CMKs)
* CMKs are protected by HSM (Hardware Security Modules)
* KMS generates **data keys** which are **used to encrypt the data** and then are **encrypted themselves**. (Envelope encryption) Unencrypted data keys are never stored
* Logged in CloudTrail
* Encryption Keys are **REGIONAL**

Types of CMK’s

* Customer Managed CMK: Free, used by default if you pick encryption in most AWS services. Only that service can use them directly.
* **AWS Owned CMK:** used by AWS on a shared basis across many accounts; you typically won see these
* **Customer managed CMK**: Allows key rotation; controlled via key policies and can be enabled/disabled

|  |  |
| --- | --- |
| Symmetric | Asymmetric |
| Same key used for encryption and decryption | Mathematically related public/private key pair |
| AES-256 | RSA and elliptic-curve cryptography (ECC) |
| Never leaves AWS unencrypted | Private key never leaves AWS unencrypted |
| Must call the KMS APIs to use | Must call the KMS APIs to use private key |
| AWS services integrated with KMS use symmetric CMKs | Download the public key and use outside AWS |
| Encrypt, decrypt, and re-encrypt data | Used outside AWS by users who can’t call KMS APIs |
| Generate data keys, data keys, data key pairs, and random byte strings | AWS services integrated with KMS do not support asymmetric CMKs |
| Import your own key material |  |

Usage:

* KMS APIs - directly encrypt/decrypt data using master keys in KMS
* AWS Services can encrypt your data (sometimes as easy as checking a box)
* AWS Encryption SDK that can encrypt it
* MAX SIZE: **4KB,** can’t encrypt anything larger, must use another method

API Calls

* Use key-d for commands
* Aws kms encrypt, decrypt, re-encrypt, enable-key-rotation

You can import your own 256-bit symmetric keys

## CloudHSM

* Dedicated hardware security module (HSM)
* **FIPS 140-2 Level 3**
* Level2 is KMS
* Manage your own keys
* No access to the AWS-managed component
* Runs within your VPC
* Keys irretrievable
* **Regulatory compliance requirements**

## Parameter Store

* Component of AWS systems manager (SSM)
* Parameter Store
* Provides configuration and password management including passwords, database strings, and license stores
* Provided at no cost
* Similar to Secrets Manager
* Track versions

## Secrets Manager

* Similar to systems manager parameter store
* Charge per secret stored and per 10,000 API calls
* Automatically rotate secrets
* Apply the new key/password in RDS for you
* Generate random secrets

## AWS Shield

* Protect against distributed denial-of service (DDoS) attacks

|  |  |
| --- | --- |
| AWS Shield Standard | AWS Shield Advanced |
| Automatically enabled for all customers at no cost. | $3000 per month per org |
| Protects against common layer 3 and 4 attacks | Enhanced protection for EC2, ELB, cloudfront, global accelerator, route53 |
| SYN/UDP floods | Business and enterprise support customers get 24 x 7 access to the DDOs response team (DRT) |
| Reflection attacks | DDoS cost protection |

## WAF - Web Application Firewall

Summary:

* Let’s you monitor HTTPS requests to Cloudfront, ALB, or API Gateway.
* Control access to content
* Blocked traffic returns 403
* Configure filtering rules to allow/deny traffic;
* Protection for common web attacks
* Configure Web ACLs (Access Control Lists)

## AWS Firewall Manager

* Centrally configure and manage firewall rules across an AWS org
* WAF rules: ALB, API Gateway, CloudFront distributions
* AWS shield Advanced protections: ALB, ELB classic, EIP, CloudFront Distributions

## Lambda

Summary:

* FAAS
* Languages: Node, Java, Go, C#, Python
* Layers allow you to put unchanging code (libraries) across functions
* Triggers: S3, Dynamo, Kinesis, SNS, SES, SQS, Cognito, Cloudformation, Cloudwatch Logs/Events, CodeCommit, Scheduled Events, Config, Alexa, Lex, API Gateway, IOT Button, CloudFront, Kinesis Data Firehouse
* **Scales OUT not up**
* Can trigger other lambda functions, 1 event can = x functions if functions trigger other functions
* Architecture can get complicated, AWS X-ray allows you to debug what is happening.
* Can do things globally, you can use it to back up s3buckets to other s3buckets

**Lambda@Edge**

* CloudFront feature that allows you to Run code closer to users of your app. Which improves performance and reduces latency.
* Responds to a cloudfront event
* Suck to work with

If a lambda is invoked asynchronously and retries twice before they fail and send to DLQ

you don't have to provision or manage infrastructure in multiple locations around the world. You pay only for the compute time you consume - there is no charge when your code is not running.

## SAM - Serverless Application Model

Summary:

* AWS::Serverless Transform - a macro hosted by AWS CloudFormation
* Takes a SAM template and converts it to a compliant cfn template
* Requires no special permissions

Globals - Lets you defined properties that you can use in all Serverless Functions and APIs. **AWS::Serverless::Function** and **AWS::Serverless::Api**

Resources:

* AWS::Serverless::Function - Creates lambda, IAM execution role, and event source mappings (Triggers), Also creates an API but not as extensible as using AWS::Serverless::Api
* AWS::Serverless::Api - Creates collection of API Gateway resources/methods
* AWS::Serverless::Application - Embeds a serverless app from either AWS serverless app repo (the AWS provided projects) or an S3 bucket
* AWS::Serverless::SimpleTable - Creates a DynamoDB table with one attribute (primary key)
* AWS::Serverless::LayerVersion - The new lambda layers stuff

CLI Commands

* Sam package
* Sam deploys

Ex: template.yaml

<https://www.baeldung.com/aws-serverless>

## ECS - Elastic Container Service

Summary

* Managed container orchestration service
* Container management service that makes it easy to run, stop, and manage Docker containers on a cluster of Amazons EC2 instances.
* To prepare your application to run on Amazon ECS, you create a task definition
* Defines rules for CPU and memory requirements
* Free for real
* Monitors resource utilization
* Deploy, update, roll back

**Components**

* **Cluster:** logical collection of ECS resources- either ECS instances or fargate instances
* **Task Definition:** Defines your app. Similar to a dockerfile but for running containers in ECS. Can contain multiple containers
* **Container Definition:** Inside a task definition, it defines the individual containers a task uses. Controls CPU and memory allocation and port mappings.
* **Task:**  single running copy of any containers defined by task definition. One working copy of an application
* **Service:** allows task definitions to be scaled by adding tasks. Defines min and max values
* **Registry:** Storage for container images (ECR or Docker Hub). Used to download images to create containers
* ECR - When using ECR image in definition use full registry/repository:tag

*aws\_account\_id*.dkr.ecr.*region*.amazonaws.com/*my-web-app*:*latest*

## Fargate

* Serverless container engine
* Eliminates need to provision and manage servers
* Works with ECS and EKS
* Each workload runs in its own kernel
* Isolation and security
* Choose EC2 instead if: compliance requirements, require broader customization, Require GPUs

## EKS

* Elastic Kubernetes service
* K8s is open-source software that lets you deploy and manage containerized apps at scale.
* Containers are grouped in pods
* Like ECS, supports both EC2 and Fargate

## ECR

* Managed docker container registry
* Store, manage, and deploy images
* Integrated with ECS and EKS
* Works with on-prem deployments
* Highly available
* Integrated with IAM

## Amazon MQ

* Fully managed service for open-source message brokers

## AWS GuardDuty

* Threat detection service that uses machine learning to continuously monitor for malicious behavior.
* Looks for unusual api calls
* Unauthorized deployments
* Monitors cloudtrail logs, vpc flow logs, and dns logs
* Uses AI to learn what normal behavior looks like in your account and alert you of any abnormal or malicious behavior.
* Updates a database of known malicious domains using feeds from third parties.
* Findings appear in the GuardDuty dashboard. CloudWatch Events can be used to trigger a lambda function to address a threat.

## AWS Migration Hub

* A single place to track the progress of your app migration to AWS. It integrates with Server Migration Service (SMS) and Database Migration Service (DMS)

## AWS Transfer Family

* securely scales your recurring business-to-business file transfers to Amazon Simple Storage Service (S3) and Amazon Elastic File System (EFS) using SFTP, FTPS, and FTP protocols.

supported destination for event notifications

What can an EBS volume do when snapshotting the volume is in progress”

Amazon RDS automatically provisions and maintains a synchronous (can co-exist) standby replica in a different Availability Zone.

## Cost explorer

* visualize, understand, and manage your AWS costs and usage over time

## Aws budgets

* Improve planning and cost control with flexible budgeting and forecasting
* Set up custom budgets

## AWS inspector

* Amazon Inspector is an automated security assessment service that helps improve the security and compliance of applications deployed on AWS. Amazon Inspector automatically assesses applications for exposure, vulnerabilities, and deviations from best practices.

You can attach a network interface to an instance when it's running (hot attach), when it's stopped (warm attach), or when the instance is being launched (cold attach).

You can detach secondary network interfaces when the instance is running or stopped. However, you can't detach the primary network interface.

You can move a network interface from one instance to another, if the instances are in the same Availability Zone and VPC but in different subnets.

**SAA Concepts I struggled with(could be duplicated)**

## AWS VPN

* Extend your on-premises networks to the cloud and securely access them from anywhere
* establish secure connections between your on-premises networks, remote offices, client devices, and the AWS global network. AWS VPN is comprised of two services: AWS Site-to-Site VPN and AWS Client VPN.

## AWS SQS

* for delay queues, a message is hidden *when it is first added to queue*, whereas for visibility timeouts a message is hidden *only after it is consumed from the queue*.
* Amazon SQS doesn't automatically delete the message.
* Messages are retained from 1minute to 14 days

Delay queue

* Minimum delay in seconds is 0
* The maximum is 15 minutes

Visibility time out

* period of time during which Amazon SQS prevents other consumers from receiving and processing the message
* The default visibility timeout for a message is 30 seconds. The minimum is 0 seconds. The maximum is 12 hours.

Diagram

Description automatically generated

## AWS VPC NACL

* The default configuration of the default NACL is Allow, and the default configuration of a custom NACL is Deny.

## Route tables

* Contains a set of rules, called *routes*, that are used to determine where network traffic from your subnet or gateway is directed.
* Tells the data where to go
* Provides the connection to various resources in VPC

## Auto scaling Groups

**Cool down period**

* A scaling cooldown helps you prevent your Auto Scaling group from launching or terminating additional instances before the effects of previous activities are visible.
* Default cooldown period automatically applies to any activities for simple scaling policies and can be requested for manual scaling policies.
* Default value is **300 seconds**
* Cooldown does not begin until instances move out of terminating: wait state

## Route 53 Geolocation vs Geoproximity Routing Policy

* **Geolocation:** Use when you want to **route traffic based on the location of your users**. Example: all hits from Europe route to the Frankfurt region.
* **Geoproximity routing policy** – Use when you want to **route traffic based on the location of your resources and users.** You can also optionally choose to **route more traffic or less to a given resource by specifying a value, known as a bias**

## S3 storage classes

1. **S3 Standard** - 99.99% availability, 99. (9\*11) durability
2. **S3 Infrequently Accessed (IA)**- 99.9% availability, Lower fee than S3 but you are **charged a retrieval fee**, good for files **accessed every 1 - 6 months**
3. **S3 Intelligent Tiering** –Using machine learning to optimize costs by automatically moving data to the most cost-effective access tier, without performance impact or operational overhead
4. **S3 One Zone IA** - Same as IA but only stored in **one availability zone 99.5% availability cost is 20% less**
5. **Glacier** - VERY cheap, used for archiving, takes 3-5 hours to access. Retrieval time of 12hour is acceptable. Configurable retrieval time
6. **S3 Glacier Deep Archive** – Amazon S3’s lowest cost storage class where retrieval time is 12 hours.

## EBS Volumes

Calendar

Description automatically generated

## AWS Data Sync

* service that automates and accelerates moving data between on premises and AWS storage services.

## AWS Data Sync vs AWS Storage Gateway

* Data Sync primarily used to migrate existing data to Amazon S3. On the other hand, AWS Storage Gateway is more suitable if you still want to retain access to the migrated data and for ongoing updates from your on-premises file-based applications.

## Amazon Lifecycle Manager

* you can manage the lifecycle of your AWS resources. You create lifecycle policies, which are used to automate operations on the specified resources.

## AWS Trusted Advisor

* provides recommendations that help you follow AWS best practices. Trusted Advisor evaluates your account by using checks. These checks identify ways to optimize your AWS infrastructure, improve security and performance, reduce costs, and monitor service quotas.

## Event Bridge (CloudWatch event)

* serverless event bus. It allows you to pass events from a source to an endpoint. Essentially, it’s the glue that holds your serverless app together

Diagram

Description automatically generated with medium confidence

* pretty cool because you can trigger a lambda based on the state of an ec2 instance
* holds together a serverless application and Lambda functions. Any API call that happens in AWS can alert a lambda function or a variety of different endpoints that something has happened
* Very common use case is triggering lambda functions when an AWS API call happens
* Fastest way to respond to things happening in your environment.

## Amazon aurora Cluster vs. read endpoint

Cluster Endpoint

* connects to the current primary DB instance for that DB cluster.
* Only endpoint that can perform write operations such as DDL statements
* the endpoint you connect to when you first set up a cluster or when your cluster only contains a single DB instance.

Reader endpoint

* load-balancing support for read-only connections to the DB cluster.
* Use for reading only operations such as queries
* reduces the overhead on the primary instance. It also helps the cluster to scale the capacity to handle simultaneous SELECT queries, proportional to the number of Aurora Replicas in the cluster.
* Each Aurora DB cluster has one reader endpoint.

Custom endpoint

* represents a set of DB instances that you choose.
* When you connect to the endpoint, Aurora performs load balancing and chooses one of the instances in the group to handle the connection.

Instance endpoint

* connects to a specific DB instance within an Aurora cluster. Each DB instance in a DB cluster has its own unique instance endpoint

## OSI Model

Timeline

Description automatically generated