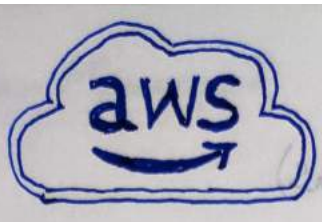
INDEX

NAME: S. Ashfaq Ahamed STD.: _____ SEC.: _____ ROLL NO.: _____ SUB.: _____

7/4/2021



What is Cloud Computing?

→ On-demand delivery of compute power, database storage, applications, etc.

→ Pay-as-you-use.

→ Access as many as possible resources instantly.

→ Simple way to access servers, storage, databases and set of application service.

Characteristics of Cloud Computing

* On-demand self service
* Broad network access
* Multi-tenacy and resource pooling
* Rapid elasticity and scalability
* Measured service

Problems Solved

* Flexibility
* Cost-effectiveness
* Scalability
* Elasticity
* High availability
* Fault tolerance

Advantages of Cloud Computing

→ Trade Capital expense (CAPEX) for Operational expense (OPEX)

→ Benefit from massive economics of scale.

→ Stop guessing capacity.

→ Increase speed and agility.

→ Stop spending money running and maintaining data centers.

→ Go global in minutes.

Types of Cloud Computing

i) Infrastructure as a Service (IaaS)

* Provide building blocks for cloud IT
* Provides networking, computers, data storage space
* Easy parallel with traditional on-premises IT

ii) Platform as a Service (PaaS)

* Removes the need for company to manage underlying infrastructure
* Focus on the deployment and management of your applications.

iii) Software as a Service (SaaS)

* Complete product that is run and managed by the service provider.

IAM - Identity and Access Management

- Users & Groups
- Global service
- Root account → Created by default shouldn't be shared or used.
- Users → Can be grouped and shared.
- User can belong to multiple groups
- Users or Groups can be assigned JSON documents called policies
- In AWS, apply least privilege principle → Don't give more permissions than the nee

AWS CLI

→ Protected by access keys.

→ Users manage their own access keys

→ Just like passwords. Don't share them.

Access Key ID ~ Username

Secret access key ~ Password

EC2 - Elastic Compute Cloud

→ Most popular AWS offering
→ Infrastructure as a Service
→ Secure. Reliable. Resizable compute capacity
→ Web-scale cloud computing easier for developers.
→ It mainly consists of
  * Renting virtual machines - EC2
  * Storing data on virtual drives - EBS
  * Distributing load across machines - ELB
  * Scaling the services using an auto-scaling group (ASG)

Sizing & Configuration

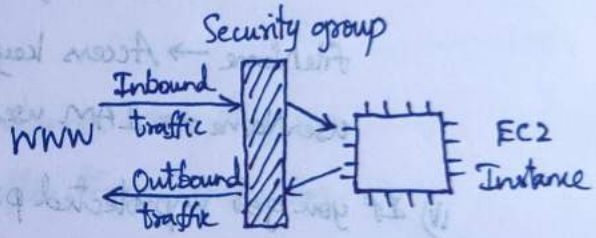
- Operating system: Linux or Windows
- How much compute power & cores (CPU)
- How much random access memory (RAM)
- How much storage space:
  i) Network-attached (EBS & EFS)
  ii) Hardware (EC2 Instance store)
- Network card: speed of the card. Public IP address
- Firewall rules: security group
- Bootstrap script (configure at first launch): EC2 User Data

EC2 Instance Types

i) General purpose → Diversity of workloads such as web servers or code repositories.
ii) Compute Optimized → Great for compute-intensive tasks, high performance processor
iii) Memory Optimized → Fast performance for large data sets in memory.
iv) Accelerated Computing
v) Storage Optimized → Great for storage-intensive tasks, read write access large data
vi) Instance Features
vii) Measuring Instance Performance

Security Groups

→ Fundamentals of network security.
→ Controls how traffic is allowed
→ It only contains allow rules
→ Can reference by IP or security group



Functions of SGs

- Access to Ports
- Authorised IP ranges - IPV4 and IPv6
- Control of Inbound network (from other to the instance)
- Control of Outbound network (from the instance to other)
- Eg: HTTP, SSH

Ports to know

- 3389 = RDP (Remote Desktop Protocol) - Log into Windows instance
- 22 = SSH (Secure Shell) - Log into a Linux instance.
- 21 = FTP (File Transport Protocol) - Upload files into a file share
- 22 = SFTP (Secure File Transport Protocol) - Upload files using SSH
- 80 = HTTP (Hypertext Transfer Protocol) - Access unsecured websites
- 443 = HTTPS (" " Secure) - Access secured websites

SSH - Secure Shell

→ Supports on Linux, Mac and Windows (versions ≥ 10)
→ One of the most important function
→ Allows to control a remote machine all using the command line.

Windows 10 & Above

i) Go to Windows Powershell
ii) Type ssh and see if it responds back.
iii) ssh -i C:\~~\fileName.pem userName@public IP address

-i → Identity command

fileName → Access key

userName → IAM user role name

iv) If you get unprotected private key file:

- Go to file → Right click → Go to properties → Security → Advanced
- Modify Owner and remove other owners.
- Remove or disable inheritance

v) Go to Cmd and paste step 3 and it will work flawlessly.

EC2 Instance Connect

- Click on the instance and the click Connect on top.
- Add userName under EC2 Instance connect.
  → ec2-user ~ default
- It is based on Amazon Linux 2 so uses Linux commands.
- It also includes inbuilt AWS commands

EC2 Instances Purchasing Options

i) On-Demand instances - short workload, predictable pricing

ii) Reserved : Minimum 1 year

a) Reserved instances - long workloads.

b) Convertible Reserved instances - long workloads with flexible instances.

c) Scheduled Reserved instances - Eg: Every weekends

iii) Spot instances - short workloads, cheap, can lose instances (less reliable)

iv) Dedicated Hosts - Book an entire physical server, control instance placement

EBS - Elastic Block Store Volume

→ Network drive you can attach to your instances while they run.

→ Allows instances to persist data even after their termination.

→ They can be only mounted to one instance at a time (at the CCP level)

→ They are bound to a specific availability zone

→ Analogy: Think of them as a "network USB stick"

→ It's a network drive not a physical drive so there might be latency.

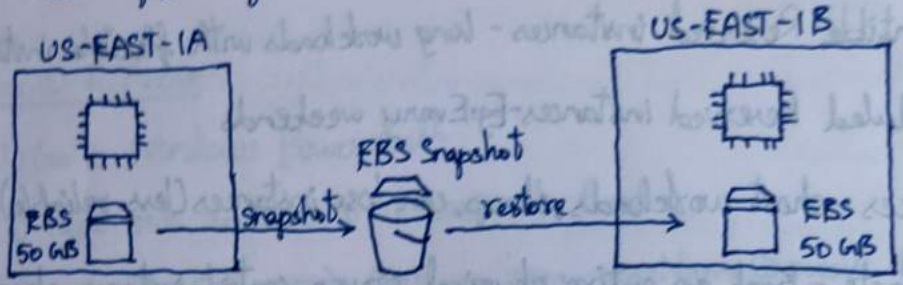
→ Can be detached from an EC2 instance and attached to another one quickly.

→ Have a provisioned capacity [size in GBs and IOPS (I/O per second)]

→ We can have an EBS unattached and attach it on demand.

EBS Snapshots

* Make a backup (snapshot) of your EBS volume at a point of time.
* Not necessary to detach volume to do snapshot but recommended.
* Can copy snapshots across AZ or region.

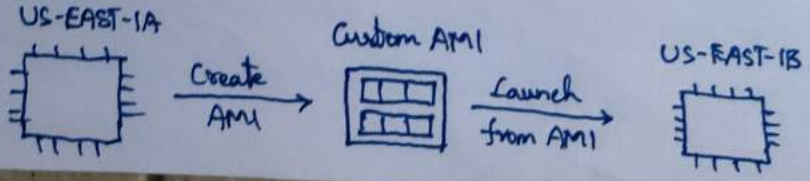


AMI - Amazon Machine Image

→ Customization of an EC2 instance
→ Can add your own software, configuration, operating system, monitoring.
→ Faster boot/configuration time because all your software is pre-packaged.
→ AMI are built for a specific region (and can be copied across regions)
→ You can launch EC2 instances from:
  a) Public AMI : Amazon Linux 2
  b) Your own AMI : Make and maintain yourself
  c) AWS Marketplace AMI : Sold by someone else not AWS

AMI Process (From an EC2 instance)

* Start an EC2 instance and customize it
* Stop the instance (for data integrity)
* Build an AMI - this will create EBS snapshots
* Launch instances from other AMIs



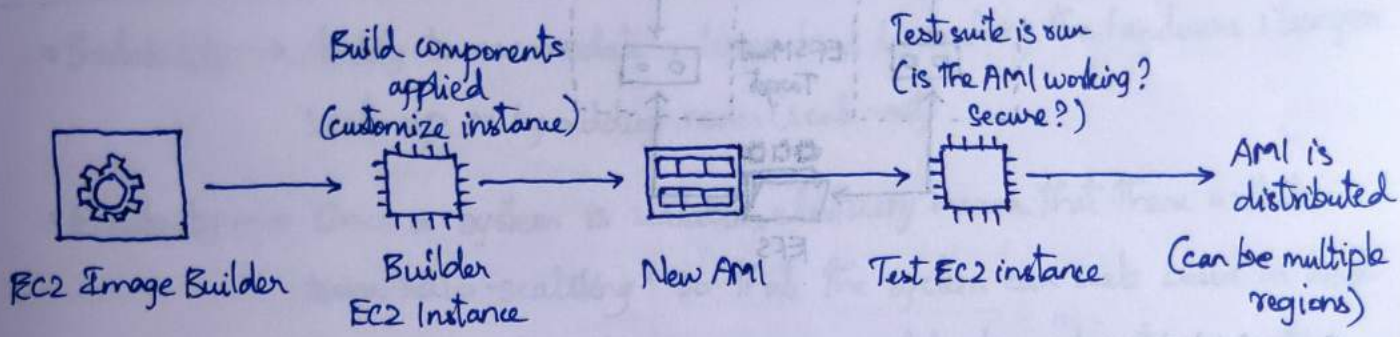
EC2 Image Builder

→ Used to automate the creation of Virtual Machines or container images.

→ Automate the creation, maintain, validate and test EC2 AMIs

→ Can be run on a schedule (weekly, whenever packages are updated, etc...)

→ Free service (only pay for the underlying resources)



EC2 Instance Store

→ EBS volumes are network drives with good but limited performance.

→ In need of high-performance hardware disk - use EC2 Instance store.

→ Better I/O performance.

→ EC2 Instance store lose their storage if they're stopped (ephemeral)

→ Not a good choice for large long term data.

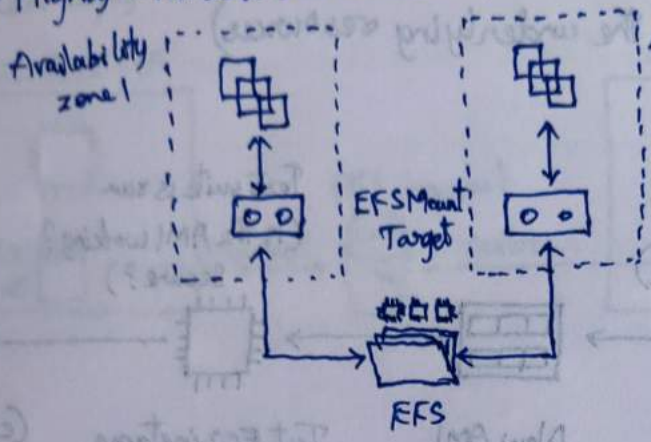
→ Good for buffer/cache/scratch data/temporary content.

→ Risk of data loss if hardware fails

→ Backups and replication are your responsibility.

EFS - Elastic File System

- Managed NFS (Network file system) that can be mounted on 100s of EC2
- EFS works with Linux EC2 instances in multi-AZ
- Highly available, scalable, expensive (3x gp2), pay per use, no capacity plann



Scalability

→ Application/system can handle greater loads by adapting.

a) Vertical scalability

b) Horizontal scalability (elasticity)

a) Vertical scalability - Scale up/down

- Increase the size of the instance
- For eg: t2.micro ⟹ t2.large
- Very common for non distributed systems such as database.
- There is a limit to how much you can vertically scale (hardware limit)

b) Horizontal scalability - Scale in/out

- Increasing the number of instances
- Adding another t2.micro and so on
- Very common for web applications/modern applications.
- Its easy to horizontally scale. Thanks to Amazon EC2

High Availability

→ Usually goes hand in hand with horizontal scaling.

→ Means running your application/system in at least 2 Availability Zones.

→ Goal is to survive a data center loss (disaster)

Scalability vs Elasticity vs Agility

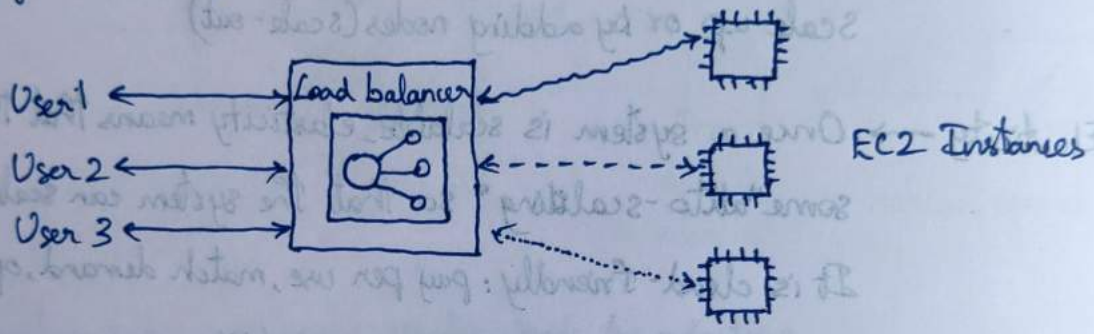
* Scalability → Ability to accomodate a larger load by making the hardware stronger scale-up or by adding nodes (scale-out)

* Elasticity → Once a system is scalable, elasticity means that there will be some "auto-scaling" so that the system can scale based on load. It is cloud-friendly: pay per use, match demand, optimize costs.

* Agility → Not related to scalability or elasticity
New IT resources are just a click away which means that you reduce the time to make those resources available to your developers from weeks to just minutes.

Elastic Load Balancing - ELB

→ Load balancers are servers that forward internet traffic to multiple servers
→ Spread load across multiple downstream instances (EC2 instances)
→ Expose a single point of access (DNS) to your application
→ Seamlessly handle failures and do regular health checks to your instances
→ Provide SSL termination (HTTPS) for your websites.
→ High availability across zones.



* An ELB is a managed load balancer. AWS is responsible for the following:
  i) It takes care of upgrades, maintenance, high availability.
  ii) It provides only a few configuration knobs.

* It costs less to setup your own load balancer but it will be a lot more effort on your end (maintenance, integrations)

* 3 kinds of load balancers offered by AWS:
  a) Application Load Balancer (HTTP/HTTPS only) - Layer 7
  b) Network Load Balancer (ultra-high performance, allows for TCP) - Layer 4
  c) Classic Load Balancer (slowly retiring) - Layer 4 & 7 [outdated]

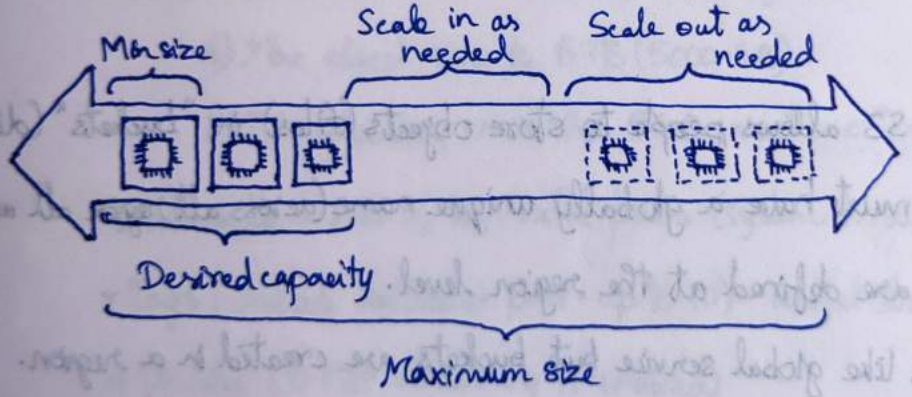
* New kind is Gateway Load Balancer

Auto Scaling Group - ASG

* In real-life, the load on your websites and application can change.

Eg. Traffic difference between day/night.

* In the cloud, you can create and get rid of servers very quickly.
* The goal of ASG is to:
  i) Scale out (add EC2 instances) to match increased load.
  ii) Scale in (remove EC2 instances) to match decreased load.
  iii) Ensure we have minimum and maximum number of machines running.
  iv) Automatically register/deregister new instances to a load balancer.
  v) Replace unhealthy instances.
* Huge cost savings: only run at an optimal capacity



Scaling Strategies

* Manual Scaling → Update the size of an ASG manually.
* Dynamic Scaling → Respond to changing demand
  a) Simple/Step scaling
     If CPU > 70% add 2 units
  b) Target Tracking scaling
     Average CPU ≥ 40%.
* Scheduled Scaling → Anticipate on known usage patterns.
* Predictive Scaling → Predict future traffic ahead of time

Amazon S3

→ One of the main building block of AWS
→ It's advertised as "infinitely scaling" storage
→ Many websites use Amazon S3 as backbone
→ Many AWS services use Amazon S3 as an integration as well

S3 Use Cases

- Backup & Storage
- Disaster recovery
- Archive
- Hybrid Cloud storage
- Application hosting
- Media hosting
- Data lakes and big data analytics
- Software delivery
- Static website

S3 - Buckets

- Amazon S3 allows people to store objects (files) in "buckets" (directories)
- Buckets must have a globally unique name (across all region all accounts)
- Buckets are defined at the region level.
- S3 looks like global service but buckets are created in a region.
- Naming convention
    i) No uppercase
    ii) No underscore
    iii) 3-63 characters long
    iv) Not an IP
    v) Must start with lowercase letter or number

S3 - Objects

- Objects (files) have a key.
- The "key" is the FULL path:
  a) s3://bucket-name/file.txt (Here file.txt → key)
  b) s3://bucket-name/folder-name/file.txt
- The key is composed of prefix + object name
  folder-name ⟹ prefix
  file.txt ⟹ object
- There is no concept of "directories" within buckets (although the UI will trick you to think otherwise)
- Just keys with very long names that contain slashes ("/")
- Object values are the content of the body:
  a) Max object size is 5 TB (5000 GB)
  b) If uploading more than 5 GB, must use "multi-part upload"
- Metadata - List of key/value pairs (system or user metadata)
- Tags (Unicode key/value pair - up to 10) - useful for security/lifecycle
- Version ID (If versioning is enabled)

S3 Security

- User based
  - IAM policies - which API calls should be allowed for a specific user
- Resource based
  - Bucket policy → bucket wide rules from S3 console - allows cross account
  - Object Access Control List (ACL) - finer grain
  - Bucket Access Control List (ACL) - less common
- Encryption - encrypt objects in Amazon S3 using encryption keys.

Note: An IAM principal can access an S3 object if

a) the user IAM permissions allow it OR the resource policy allows it.

b) AND there's no explicit deny.

S3 Bucket Policies

i) JSON based policies

- Resources: buckets and objects
- Actions: set of API to allow or deny
- Effect: allow/deny
- Principal: the account/user to apply the policy to

ii) Use S3 Bucket for policy to:

- Grant public access to the bucket.
- Force objects to be encrypted at upload.
- Grant access to another account (Cross account)

S3 Websites

* S3 can host static websites and have them accessible on the www
* The website URL will be:
    a) <bucket-name>.s3-website-<AWS-region>.amazonaws.com
    OR
    b) <bucket-name>.s3-website.<AWS-region>.amazonaws.com
* If you get a 403 (Forbidden) error, make sure the bucket policy allows public reads.

S3 Versioning

→ You can version your files in Amazon S3
→ It is enabled at the bucket level
→ Same key overwrite will increment the version: 1, 2, 3...
→ It is best practice to version your buckets
    * Protect against unintended deletes (ability to restore a version)
    * Easy roll back to previous version.

S3 Access Logs

* For Audit purpose, we want to log all access to S3 buckets.
* Any request made to S3, from any account, authorized or denied will be logged into another S3 bucket.
* That data can be analysed using data analysis tools.
* Very helpful to come down to root cause of an issue or view suspicious patterns.

S3 Replication

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

CRR → Compliance, Lower latency access, replication across accounts

SRR → Log aggregation, live replication between production and test accounts.

S3 Storage Class

i) S3 Standard - General Purpose
ii) S3 Standard - Infrequent Access (IA)
iii) S3 One Zone - Infrequent Access
iv) S3 Intelligent Tiering
v) Glacier
vi) Glacier Deep Archive
vii) S3 Reduced Redundancy Storage

S3 Durability

→ High durability (99.999999999) % across multiple AZ
→ For 10,000,000 objects, incur a loss of single object once every 10,000 years.
→ Same for all storage classes.

S3 Availability

→ Measures how readily a service is
→ S3 Standard has 99.99% availability - unavailable for 53 minutes a year
→ Varies depending on storage class

i) S3 Standard

- 99.99% availability → 99.99%
- Used for frequently accessed data
- Low latency and high throughput
- Sustain 2 concurrent facility failures
- Big data analytics, Gaming...

ii) S3 Standard - Infrequent Access (IA)

- Less frequent but rapid access
- 99.9% availability
- Lower cost with retrieval fee
- Sustain 2 concurrent failures
- Backups, Disaster recovery

iii) S3 Intelligent - Tiering

- 99.9% availability
- Low latency and high throughput
- Cost optimized automatically
  - a) Frequent access
  - b) Infrequent access.

iv) S3 One Zone - Infrequent Access

- Same as IA but in single AZ
- 99.5% availability
- Lower cost than S3-IA (20%)
- Secondary backup copies
  Storing data as we recreate

v) Glacier & Glacier Deep Archieve

- Low cost object storage for backup
- Data retained for longer term (years)

Glacier - Cheap	Glacier - Deep
a) Expedited 1-5 minutes	Cheapest
b) Standard 3-5 hours	a) Standard 12 hours
c) Bulk 5-12 hours	b) Bulk - 48 hours

S3 Object Lock

* Adopt a WORM (Write Once Read Many) model
* Block an object version deletion for a specified amount of time

Glacier Vault Lock

* Adopt a WORM (Write Once Read Many) model
* Lock the policy for future edits (can no longer be changed)

AWS Snow Family

→ Highly secure, portable devices to collect and process data at the edge and migrate data into and out of AWS

→ Data Migration
- i) Snowcone
- ii) Snowball Edge
- iii) Snowmobile

⟹ Edge Computing
- i) Snowcone
- ii) Snowball edge

Why Data Migration with AWS Snow Family?

Data	Time to transfer		
	100 Mbps	1 Gbps	10 Gbps
10 TB	12 days	30 hours	3 hours
100 TB	124 days	12 days	30 hours
1 PB	3 years	124 days	12 days

Challenges:
- Limited connectivity
- Limited bandwidth
- High network cost
- Shared bandwidth
- Connection stability

→ Offline devices sent to our place by post to perform data migrations.

→ If it takes more than a week to transfer over network, use Snowball devices.

Snowball Edge

* Physical data transport solution: move TBs or PBs of data in or out of AWS
* Alternative to moving data over the network (and paying network fees)
* Pay per data transfer job.
* Provide block storage and Amazon S3-compatible object storage.

a) Snowball Edge Storage Optimized
   → 80 TB of HDD capacity for block volume.

b) Snowball Edge Compute Optimized
   → 42 TB of HDD capacity for block volume

* Use cases: Large data cloud migration, DC decommission, disaster recovery.

Snowcone

* Small, portable computing, anywhere, rugged and secure, withstand any environ
* Light weight (4.5 pounds, 2.1 kg)
* Device used for edge computing, storage and data transfer
* 8 TBs of usable storage.
* Use Snowcone where Snowball does not fit (space-constrained environment)
* Must provide your own battery/cables.
* Can be sent back to AWS offline or connect it to internet and use AWS DataSync to send data.

AWS Snowmobile

- It is an actual truck.
- Transfer exabytes of data (1 EB = 1000 PB = 1,000,000 TBs)
- Each Snowmobile has 100 PB of capacity (use multiple in parallel)
- High security, temperature controlled, GPS, 24/7 video surveillance.
- Better than Snowball if you transfer more than 10 PB

What is Edge Computing?

- Process data while it's being created on an edge location
- Edge location can be a truck on the road, a ship on the sea, mining station
- These locations may have
  - → Limited/no internet access
  - → Limited/no easy access to computing power
- We setup a Snowball Edge/Snowcone device to do edge computing
- Use cases of Edge computing:
  - → Preprocess data
  - → Machine learning at the Edge
  - → Transcoding media streams
- Eventually (if need be) we can ship back the device to AWS (for data transfer)

Snowcone (Smaller)

- 2 CPU, 4 GB of memory
- Wired or wireless access
- USB-C power using a cord
- Optional battery

Snowball Edge - Compute

- 52 vCPUs, 208 GiB of RAM
- Optional GPU
  - for video processing
  - Machine learning
- 42 TB usable storage

Snowball Edge - Storage

- 40 vCPUs, 80 GiB of RAM
- Object storage clustering

Hybrid Cloud for Storage

→ Part of your infrastructure is on-premises

→ Part of your infrastructure is on the cloud

→ This can be due to:

* Long term cloud migrations.
* Security requirements
* Compliance requirements.
* IT strategy.

→ S3 is a proprietary storage technology (unlike EFS/NFS), so how do you expose the S3 data on-premise? AWS Storage Gateway!

Storage Cloud Native Options

AWS Storage Gateway

→ Bridge between on-premises data and cloud data in S3

→ Hybrid storage service to allow on-premises to seamlessly use the AWS Cloud.

→ Use cases: disaster recovery, backup & restore, tiered storage.

→ Types of storage gateway:

a) File Gateway

b) Volume Gateway

c) Tape Gateway

Database Intro

→ Storing data on disk (EFS, EBS, EC2, S3) can have its limits.

→ If we want to store in a structured way we need to use database.

- To be able to build indexes to efficiently query/search through the data
- Can define relationship between your datasets.

→ Databases are optimised for a purpose and come with different features, shapes and constraints.

Relational Databases

- Links between the tables and columns is referenced.
- Can use SQL language to perform queries/lookups

NoSQL Databases

- It is non relational database, there are no links.
- Purpose built for specific data models and have flexible schemas for modern apps

→ Benefits

- Flexibility : easy to evolve data model.
- Scalability : designed to scale-out by using distributed clusters.
- High Performance : optimized for specific data model
- Highly functional : types optimized for the data model.

→ Examples

- Key-value
- Document
- Graph
- in-memory
- search databases

NoSQL data example: JSON

- JSON → JavaScript Object Notation
- Common form of data for NoSQL model.
- Data can be nested.
- Fields can change over time
- Support for new arrays, etc...

```
{
  "name": "John",
  "age": 30,
  "cars": [
      "Ford",
      "BMW".
  ],
"address": {
      "street": "Dream Road"
  }
}
```

NOTE: Database can be run on EC2, but we must handle backup, patching, scaling so limitations.

AWS RDS Overview

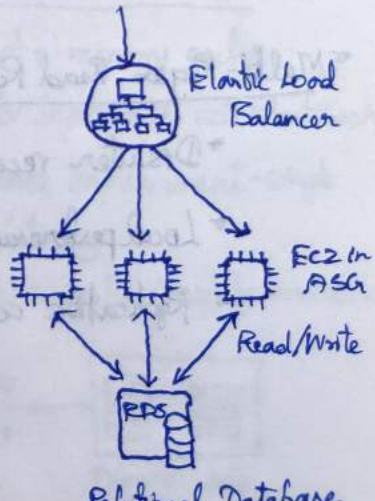
- RDS stands for Relational Database Service.
- It's a managed DB service that use SQL as a query language.
- It allows to create database in cloud that are managed by AWS

  Postgres, MySQL, MariaDB, Oracle, Microsoft SQL Server, Aurora

Advantages over deploying DB on EC2

- Automated provisioning, OS patching.
- Continuous backups and restore to specific timestamp (Point in Time restore)
- Monitoring dashboards
- Read replicas for improved read performance.
- Multi AZ setup for DR (Disaster Recovery)
- Maintenance windows for upgrades
- Scaling capability (vertical and horizontal)
- Storage backed by EBS (gp2 or io1)

BUT you can't SSH into your instances.



Amazon Aurora

→ Proprietary technology from AWS (not open sourced)

→ PostgreSQL and MySQL are both supported as Aurora DB

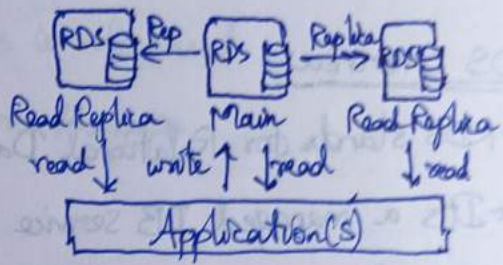
→ It is 'AWS Cloud optimized' and claims 5x performance improvement over MySQL on RDS, over 3x the performance of Postgres on RDS

→ Storage automatically grows in increments of 10GB up to 64 TB

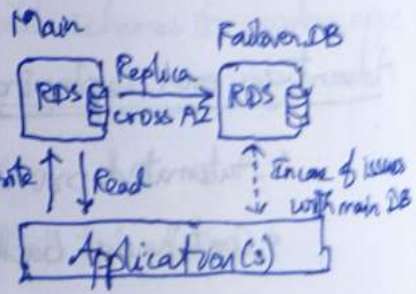
→ Costs more than RDS (20% more) but it is more efficient.

RDS Deployments

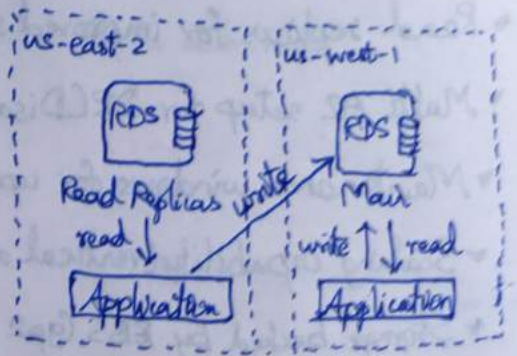
- **Read Replicas**
  - Scale the read workload of your DB
  - Can create upto 5 Read Replicas
  - Data is written to the main DB



- **Multi-AZ**
  - Failover in case of AZ outage
  - Data is only read/written to the main database
  - Read/written on Failover DB happens if main inaccessible
  - Can have only 1 other AZ as failover

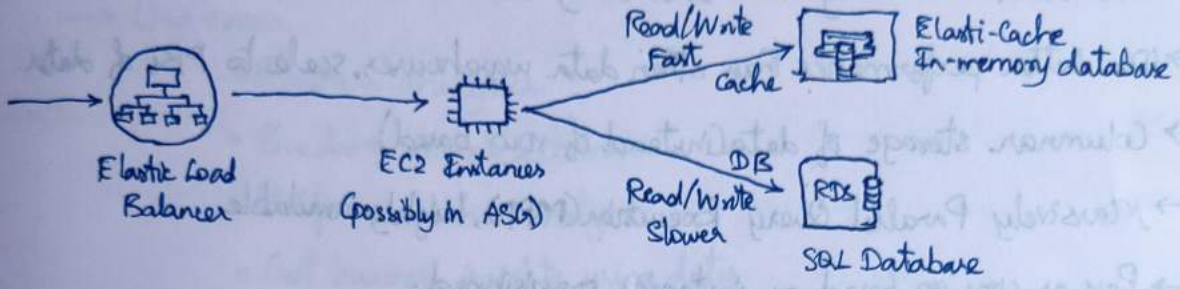


- **Multi-Region Read Replicas**
  - Disaster recovery incase of region issue
  - Local performance for global reads
  - Replication cost



Amazon ElastiCache

→ To get managed Redis or Memcached
→ Caches are in memory database with high performance, low latency
→ Helps reduce load off databases for read intensive workloads.



Dynamo DB

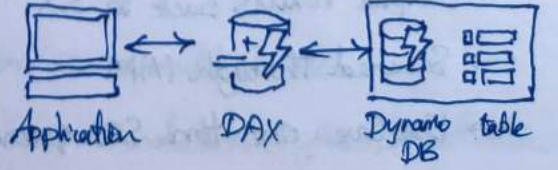
→ Fully Managed Highly available with replication across 3 AZ
→ NoSQL database - not a relational database
→ Scales to massive workloads, distributed "serverless" database.
→ Millions of requests per second, trillions of row, 100s of TB of storage.
→ Fast and consistent in performance.
→ Single-digit millisecond latency - low latency retrieval.
→ Integrated with IAM for security, authorization and administration.
→ Low cost and auto-scaling capabilities.

Type of Data

• key/value database

Dynamo-DB Accelerator (DAX)

• Fully managed in-memory cache
• Especially for DynamoDB not ElastiCache
• 10x performance improvement - single digit microsecond latency
• Secure, highly scalable & highly available



Redshift

→ Based on PostgreSQL, but it is not used for OLTP (Online Transaction Processing)
→ It is OLAP - Online Analytical Processing (analytics and data warehousing)
→ Load data once every hour, not every second - It is not continuous load data
→ 10x better performance than other data warehouses, scale to PBs of data
→ Columnar storage of data (instead of row based)
→ Massively Parallel Query Execution (MPP), highly available
→ Pay as you go based on instances provisioned.
→ Has a SQL interface for performing the queries.
→ BI tools such as AWS Quicksight or Tableau integrate with it.

Amazon EMR

→ Stands for "Elastic MapReduce"
→ Helps creating Hadoop clusters (Big Data) to analyze and process vast amount
→ Clusters can be made of hundreds of EC2 instances.
→ Also supports Apache Spark, HBase, Presto, Flink...
→ EMR takes care of all the provisioning and configuration.
→ Auto scaling and integrated with Spot instances.
→ Use cases: Data processing, machine learning, web indexing, big data

Athena

→ Fully serverless database with SQL capabilities.
→ Used to query data in S3 - Pay per query
→ Output results back to S3
→ Secured through IAM
→ Use Case: one-time SQL queries, serverless queries on S3, log analytics

Amazon QuickSight

→ Serverless machine learning-powered business intelligence service to create interactive dashboards.

→ Fast, automatically scalable, embeddable, with per-session pricing.

→ Use cases:

- Business analytics
- Building visualizations.
- Perform ad-hoc analysis.
- Get business insights using data

→ Integrated with RDS, Aurora, Athena, Redshift, S3...

DocumentDB

→ Same as Aurora for PostgreSQL/MySQL

→ It is for MongoDB (which is a NoSQL database)

→ Used to store, query and index JSON data

→ Similar "deployment concepts" as Aurora

→ Fully Managed, highly available with replication across 3 AZ

→ Storage grows automatically in increments of 10 GB up to 64TB

→ Automatically scales to workloads with millions of requests per seconds.

Amazon Neptune

→ Fully managed graph database.

→ A popular graph dataset would be a social network

→ Highly available across 3 AZ, with upto 15 read replicas.

→ Build and run applications working with highly connected datasets - optimized for these complex and hard queries.

→ Can store up to billions of relations across multiple AZs

→ Eg: Wikipedia, fraud detection, recommendation engines, social networking.

Amazon QLDB

→ It stands for "Quantum Ledger Database"

→ A ledger is a book recording financial transactions.

→ Fully managed, Serverless, Highly available. Replication across 3 AZ

→ Used to review history of all the changes made to your application data

→ Immutable system: no entry can be removed or modified, cryptographically verified

→ 2-3x better performance than common ledger blockchain frameworks, manipulate data using SQL

→ Difference with Amazon Managed Blockchain: no decentralization component, in accordance with financial regulation rules.

Amazon Managed Blockchain

→ Blockchain makes it possible to build applications where multiple parties can execute transactions without the need for a trusted, central authority.

→ It is a managed service to:

- Join public blockchain networks
- Or create your own scalable private network

→ Compatible with the frameworks Hyperledger Fabric & Ethereum

Database Migration Service (DMS)

→ Quickly and securely migrate databases to AWS, resilient, self-healing.

→ The source database remains available during the migration.

→ Homogeneous migration

- Eg: Oracle to Oracle

Heterogeneous migration

- Eg: Microsoft SQL Server to Aurora

AWS Glue

→ Managed extract, transform and load (ETL) service.

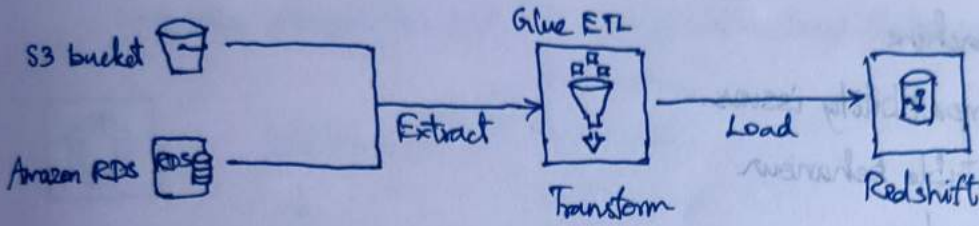
→ Useful to prepare and transform data for analytics

→ Fully serverless service

Glue Data Catalog
- Catalog of datasets

↓

EMR, Athena, Redshift



Database Summary in AWS

- Relational Database - OLTP - RDS & Aurora (SQL)
- In-memory Database - ElastiCache
- Key/value Database - DynamoDB (serverless) & DAX (cache for DynamoDB)
- Warehouse - OLAP - Redshift (SQL)
- Hadoop cluster - EMR
- Athena - Query data on Amazon S3 (serverless & SQL)
- QuickSight - Dashboards on your data (serverless)
- DocumentDB - "Aurora for MongoDB" (JSON - NoSQL database)
- Amazon QLDB - Financial Transaction Ledger
- Amazon Managed Blockchain - Hyperledger Fabric & Ethereum blockchains
- Glue - Managed ETL and Data Catalog service
- Database migration - DMS
- Neptune - Graph database

Docker

→ Software development platform to deploy apps.

→ Apps are packaged in containers that can be run on any OS

→ Apps run the same, regardless of where they're run

- Any machine
- No compatibility issues.
- Predictable behaviour
- Less work
- Easier to maintain and deploy
- Works with any language and OS, any technology

→ Scale containers up and down very quickly (seconds)

Where Docker images are stored?

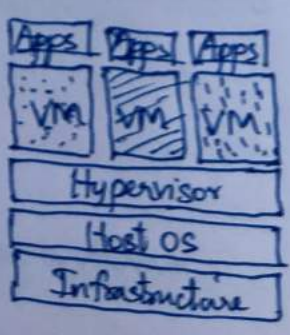
- Docker images are stored in Docker repositories
- Public Docker Hub: https://hub.docker.com/
- Find base images for many technology or OS:
  a) Ubuntu
  b) MySQL
  c) NodeJS
  d) Java, etc..
- Private: Amazon ECR (Elastic Container Registry)

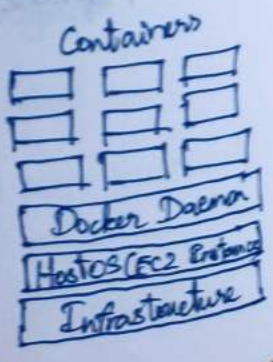
Virtual Machines vs Docker

→ 'Sort of' virtualization technology but not exactly

→ Resources are shared with host

many containers on one server





Elastic Container Service (ECS)

→ Launch docker containers on AWS

→ You must provision and maintain the infrastructure (the EC2 instances)

→ AWS takes care of starting / stopping containers

→ Has integration with the Application Load Balancer.

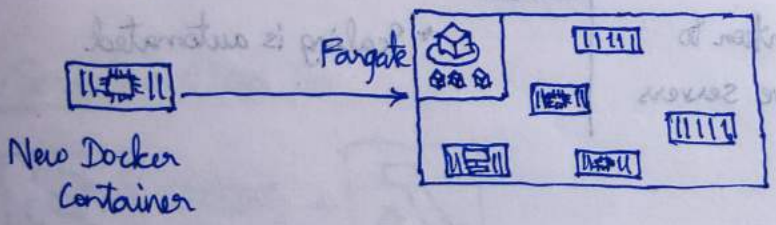
Fargate

→ Also launch Docker container on AWS

→ But do not provision the infrastructure (no EC2 instance to manage) - simpler!

→ Serverless offering

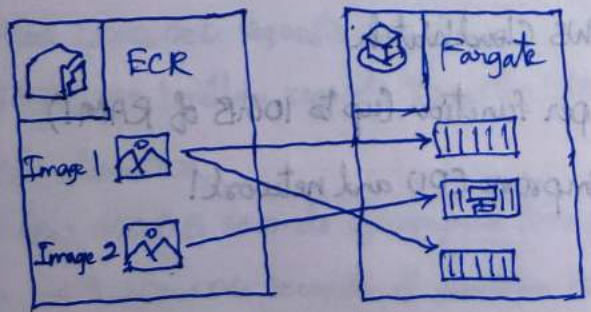
→ AWS just runs containers for you based on the CPU/RAM you need.



Elastic Container Registry (ECR)

→ Private Docker registry on AWS

→ Store your docker images to run them by ECS or Fargate



Serverless - Introduction

→ New paradigm in which developers don't have to manage servers anymore

→ They just deploy code i.e Functions!

→ Initially Serverless == FaaS (Function as a Service)

→ Serverless was pioneered by AWS Lambda but now also includes anything that's managed: 'databases', 'messaging', 'storage', etc...

→ Serverless does not mean there are no servers
- it means you don't have to manage/provision/see them.

→ Eg: Amazon S3, DynamoDB, Fargate, Lambda

AWS Lambda - Why?

Amazon EC2

- Virtual servers on cloud
- Limited by CPU & RAM
- Continuously running
- Scaling intervention to add/remove servers

Lambda

- Virtual functions - no servers to manage
- Limited by time - short executions
- Run on-demand
- Scaling is automated

Benefits of AWS Lambda

→ Easy Pricing - Pay per request and compute time.

→ Integrated with the whole AWS suite of services.

→ Event-Driven: functions get invoked by AWS when needed.

→ Integrated with many programming languages.

→ Easy monitoring through AWS CloudWatch

→ Easy to get more resources per function (up to 10GB of RAM!)

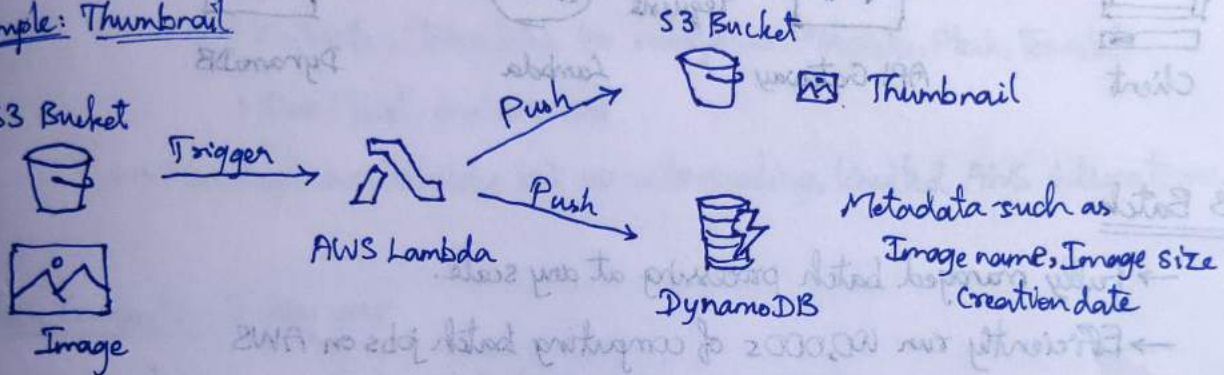
→ Increasing RAM will also improve CPU and network!

AWS Lambda Language Support

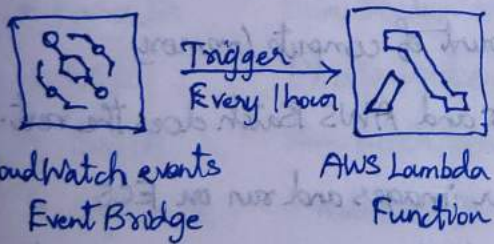
- Node.js (JavaScript)
- Python
- Java (Java 8 compatible)
- C# (.NET Core)
- Lambda Container Image
- Golang
- C#/Powershell
- Ruby
- Custom Runtime API (Community supported. Eg: Rust)

- The container image must implement the Lambda Runtime API
- ECS/Fargate is preferred for running arbitary docker images.

Example: Thumbnail



Example: Serverless CRON Job



AWS Lambda Pricing

→ Pay Per calls
- First 1,000,000 requests are free
- $0.20 per 1 million requests thereafter ($0.0000002 per request)

→ Pay Per duration
- 400,000 GB seconds of compute time per month if FREE on 1 GB RAM
- == 3,200,000 seconds if function is 128 MB RAM
- After that $1.00 for 6,000,00 GB-seconds

API Gateway

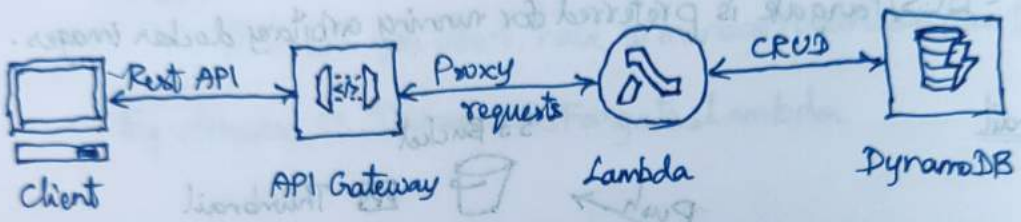
→ Example: Building a serverless API

→ Fully managed service for developers to easily create publish, maintain, monitor and secure APIs

→ Serverless and scalable

→ Supports RESTful APIs and WebSocket APIs

→ Support for security, user authentication, API throttling, API keys, monitoring



AWS Batch

→ Fully managed batch processing at any scale.

→ Efficiently run 100,000s of computing batch jobs on AWS

→ A "batch" job is a job with a start and an end (opposed to continuous)

→ Batch will dynamically launch EC2 instances or Spot instances

→ AWS batch provisions the right amount of compute/memory

→ You submit or schedule batch jobs and AWS Batch does the rest.

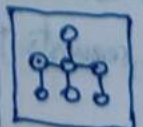
→ Batch jobs are defined as Docker images and run on ECS

→ Helpful for cost optimizations and focusing less on the infrastructure

Batch vs Lambda

Lambda

- Time limit
- Limited runtimes
- Limited temporary diskspace
- Serverless



Batch

- No time limit
- Any runtime as long as Docker image
- Rely on EBS/instance store for diskspace
- Relies on EC2 (can be managed by AWS)

Amazon Lightsail

→ Kind of a stand-alone service.

→ Virtual servers, storage, databases and networking in one place.

→ Low & predictable pricing.

→ Much simpler alternative to using EC2, RDS, ELB, EBS, Route 53...

→ Great for people with little cloud experience.

→ Can setup notifications, monitoring of your Lightsail resources

→ Use Cases:

- Simple web applications (has templates for LAMP, Nginx, MEAN, Node.js...)
- Websites (templates for Wordpress, Magneto, Plesk, Joomla)
- Dev / Test environment

→ Has high availability but no auto-scaling, limited AWS integrations.

Other Compute - Summary

- Docker : Container technology to run applications
- ECS : Run Docker containers on EC2 instances
- Fargate : Run Docker containers without provisioning the infrastructure
- ECR : Private docker images repository
- Batch : Run batch jobs on AWS across managed EC2 instances
- Lightsail : Predictable & low pricing for simple application & DB stacks
- API Gateway : Expose lambda functions as HTTP API

DEPLOYING and MANAGING INFRASTRUCTURE at SCALE

Cloud Formation - Go to US East-1 for creation

→ Its a declarative way of outlining your AWS Infrastructure, for any resources

→ For. eg:
- I want a security group.
- I want two EC2 instances using this security group.
- I want an S3 bucket
- I want a load balancer (ELB) in front of these machines

→ CloudFormation creates those for you, in the right order, with the exact configuration

Benefits

→ Infrastructure as Code
- No resources are manually created, which is excellent for control.
- Changes to the infrastructure are reviewed through code.

→ Cost
- Each resources within the stack is tagged with an identifier so you can easily see how much a stack costs you
- You can estimate the costs of your resources
- Savings strategy: In Dev you could automation deletion of templates at 5 PM and recreated at 8 AM safely.

→ Productivity
- Ability to destroy and create an infrastructure on the fly
- Automated generation of diagram for your templates
- Declarative programming (no need to figure out ordering and orchestration)

→ Don't re-invent the wheel
- Leverage existing templates on the wheel
- Leverage the documentation

→ Support (almost) all AWS resources.

CDK Overview

→ AWS Cloud Development kit (CDK)

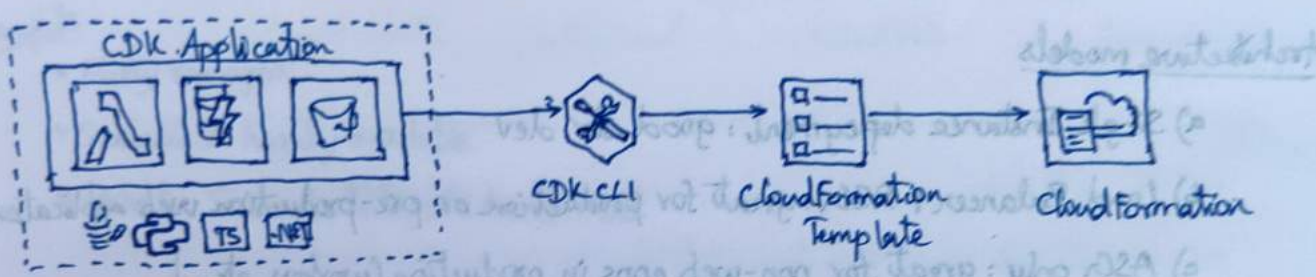
→ Define your cloud infrastructure using a familiar language:

- Javascript, Typescript, Python, Java and .NET

→ The code is "compiled" into a CloudFormation template (JSON/YAML)

→ You can therefore deploy infrastructure and application runtime code together.

- Great for Lambda functions
- Great for Docker containers in ECS/EKS



Beanstalk Overview - Why?

→ Don't want to be managing infrastructure, just focusing on deploying code.

→ Don't want to be configuring all the databases, load balancers, etc...

→ You wanna make sure whatever you're doing scales.

→ You wanna the code to run consistently across different applications and environments

AWS Elastic Beanstalk

- It's a developer centric view of deploying an application on AWS
- It uses the same components we have seen before: EC2, ASG, ELB, RDS
- But it's all in one view that's easy to make sense of
- We still have full control over the configuration.

Beanstalk = Platform as a Service (PaaS)

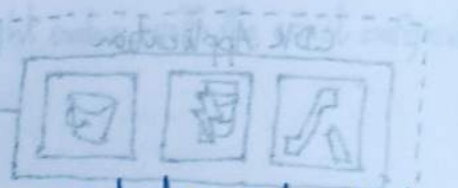
- Beanstalk is free but you have to pay for the underlying instances.

BeanStalk - Managed Service

- Instance configuration / OS is handled by Beanstalk
- Deployment strategy is configurable but performed by Elastic Beanstalk
- Capacity provisioning
- Load balancing and auto-scaling.
- Application health monitoring and responsiveness

⟹ Just the application code is the responsibility of the developer.

Architecture models



a) Single Instance deployment : good for dev

b) Load Balancer + ASG : great for production or pre-production web applications

c) ASG only : great for non-web apps in production (workers, etc...)

Health Monitoring

→ It has a full monitoring suite available.

→ Health agents on each EC2 instance that push metrics to CloudWatch.

→ Checks for app health, publishes health events.

AWS CodeDeploy

→ To deploy our application automatically.

→ It's a Hybrid service

- Works with EC2 instances
- Works with On-prem servers.

→ Servers/Instances must be provisioned and configured ahead of time with the CodeDeploy Agent.

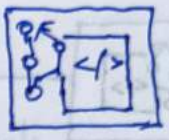
AWS Code Commit

- Before pushing the application code to the servers, it needs to be stored somewhere.
- Developers usually store code in a repository, using the Git technology.
- A famous public offering is GitHub ≃ AWS product is Code Commit

→ Source control service that hosts Git-based repositories.

→ Makes it easy to collaborate with others on code.

→ The code changes are automatically versioned.



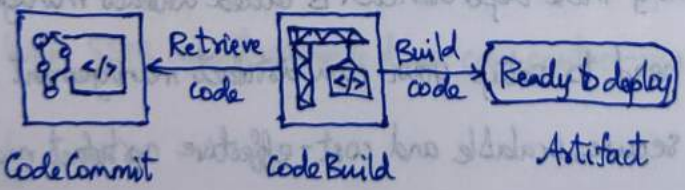
Benefits

- Fully managed
- Scalable & highly available
- Private, Secured, Integrated with AWS

AWS CodeBuild

→ Code building service on the cloud

→ Compiles source code, run tests and produces packages that are ready to be deployed (by CodeDeploy for example)



Benefits

- Fully managed, serverless
- Continuously scalable & highly available
- Secure
- Pay-as-you-go pricing - only pay for the build time.

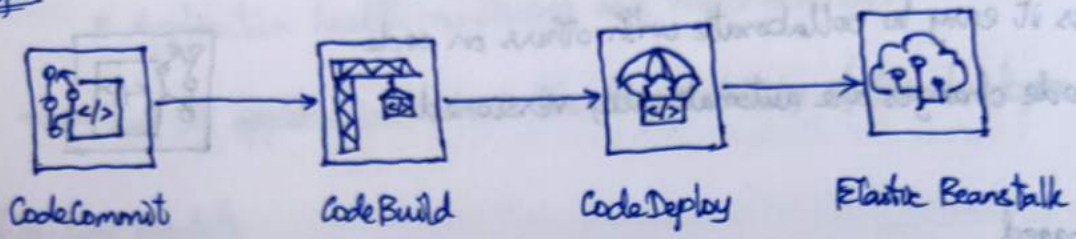
AWS CodePipeline

→ Orchestrate the different steps to have the code automatically pushed to Production

→ Code ⟹ Build ⟹ Test ⟹ Provision ⟹ Deploy

→ Basis for CICD (Continuous Integration & Continuous Delivery)

Example:



Benefits

- Fully managed.
- Compatible with CloudFormation, GitHub, 3rd-party services
- Fast delivery & rapid updates.

AWS CodeArtifact

- Software packages depend on each other to be built (also called code dependencies)
- Storing and retrieving these dependencies is called artifact management
- Traditionally you need to setup your own artifact management system.
- CodeArtifact is a secure, scalable and cost-effective artifact management.
- Works with common dependency management tools such as Maven, Gradle, npm,
- Developers and CodeBuild can then retrieve dependencies straight for CodeArtifact

AWS CodeStar

→ Unified UI to easily manage software development activities in one place

→ "Quick way" to get started to correctly setup CodeCommit, CodePipeline, ...

→ Can edit the code "in-the-cloud" using AWS Cloud9

AWS Cloud9

→ Cloud IDE for writing, running and debugging code.

→ Classic IDE (IntelliJ, Visual Studio Code) are downloaded on a computer before used.

→ Cloud IDE can be used within a web browser, no setup is necessary.

→ It allows for code collaboration in real-time (pair programming)

Systems Manager (SSM)

→ Helps you manage your EC2 and On-premises systems at scale.

→ It's an Hybrid AWS service

→ Get operational insights about the state of your infrastructure

→ Suite of 10+ products

→ Most important features are:

- Patching automation for enhanced compliance.
- Run commands across an entire fleet of servers.
- Store parameter configuration with the SSM Parameter Store

→ Works for both Windows and Linux OS

→ We need to install SSM agent onto the systems (EC2, On-Prem VM) we control.

→ With SSM agent connected we can control-run commands, patch & configure servers.

AWS Ops Works

- Chef & Puppet (both not created by AWS) helps in performing server configuration or repetitive actions.
- They work great with EC2 & On-Premises VM
- AWS Ops Works ⟹ Managed Chef & Puppet
- It's an alternative to AWS SSM
- Only provision standard AWS resources: EC2 Instances, Load Balancers, EBS volume

Leveraging the AWS Global Infrastructure

Why make a global application?

→ Decreased Latency
- Latency is the time takes for a network packet to reach a server.
- It takes time for a packet from Asia to reach the US
- Deploy your applications closer to your users to decrease latency

→ Disaster Recovery (DR)
- If an AWS region goes down (earthquake, storms, power shutdown)
- You can fail-over to another region and have your application still working.
- DR plan is important to increase the availability of your application

→ Attack protection: distributed global infrastructure is harder to attack.

Global App & Region

- Global application is an application deployed in multiple geographies.
- On AWS this could be Regions and/or Edge Locations

→ Regions: For deploying applications and infrastructure

→ Availability Zones: Made of multiple data centers.

→ Edge Locations: For content delivery as close as possible to users. (Points of Presence)
- More at https://infrastructure.aws/

a) Global DNS: Route 53
- Great for route users to closest deployment
- Great for disaster recovery strategies.

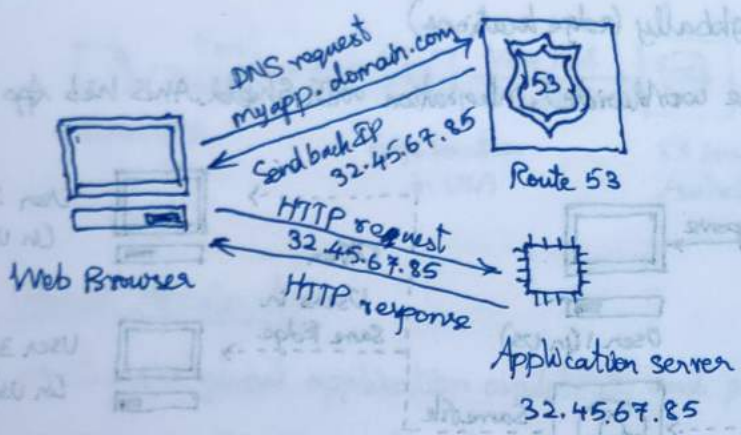
c) S3 Transfer Acceleration
- Accelerate global upload & downloads

b) Global CDN: CloudFront
- Replicate apps to AWS Edge Locations
- Cache common requests - improved experience

d) AWS Global Accelerator
- Improve global application availability

Amazon Route 53

- It is a Managed DNS (Domain Name System)
- DNS is a collection of rules and records which helps clients understand how to reach a server through URLs

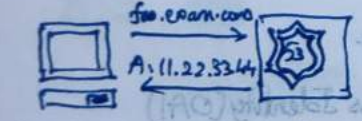


Routing Policies

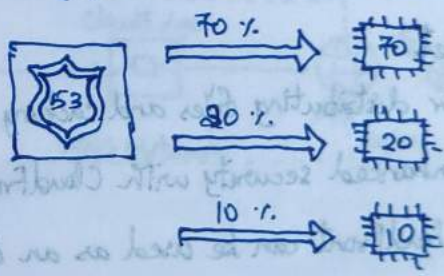
→ Need to know how to do routing based on different policy.

Simple Routing Policy

No health check



Weighted Routing Policy

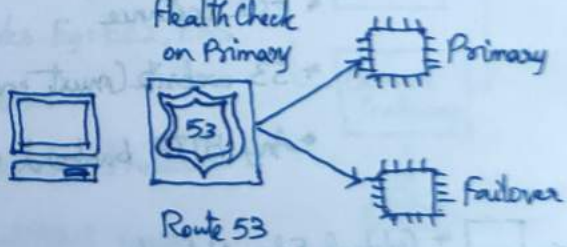


Latency Routing Policy

- Route the users to the nearest region or AZ
- To minimalize latency

Failover Routing Policy

Disaster Recovery



AWS CloudFront

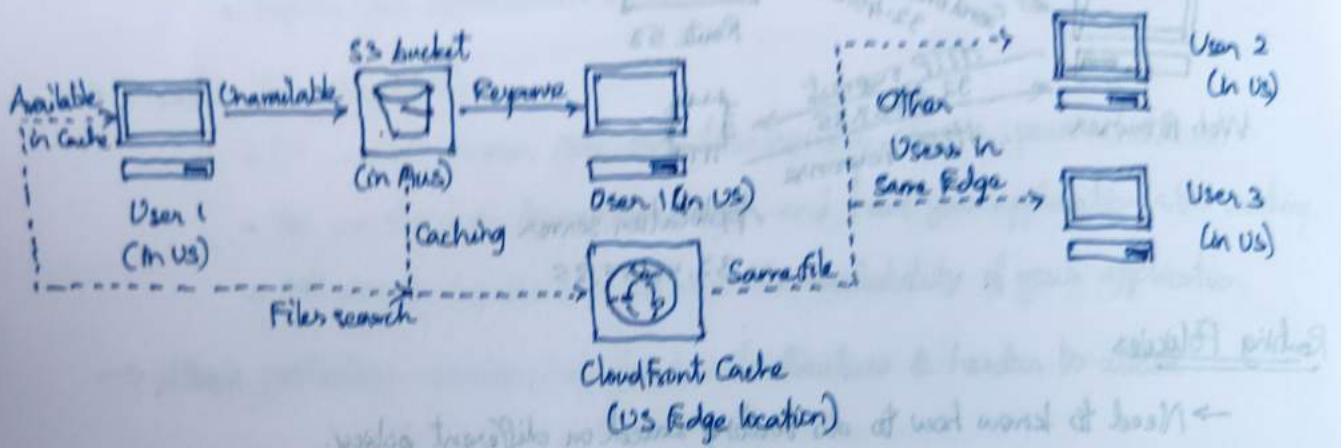
→ Content Delivery Network (CDN)

→ Improves read performance, content is cached at the edge location

→ Improves user experience

→ 216 Point of Presence globally (edge locations)

→ DDoS Protection (because worldwide), Integration with Shield, AWS Web Application Firewall



CloudFront - Origins

→ S3 bucket

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

→ Custom Origin (HTTP)

- Application Load Balancer
- EC2 Instance
- S3 website (must enable the bucket as a static S3 website)
- Any HTTP backend you want

Comparison

CloudFront

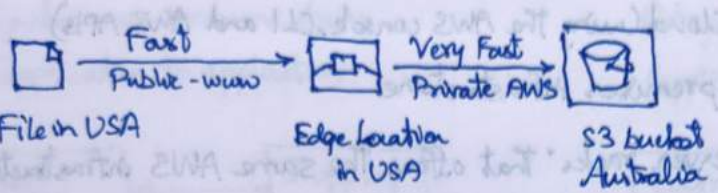
- Global Edge Network
- Files are cached for a TTL (maybe a day)
- Great for static content that must be available everywhere

S3 Cross Region Replication

- Must be setup for each region for replication need
- Files are updated in near real-time
- Read-only
- Great for dynamic content that needs to be available at low-latency

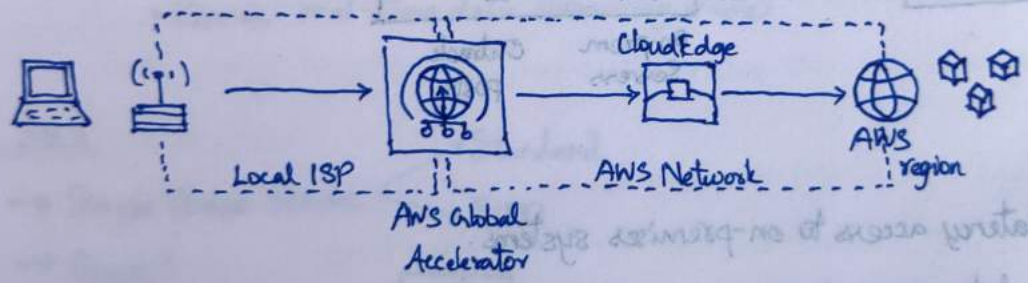
S3 Transfer Acceleration

- S3 buckets are linked only to one region. Sometimes we need to transfer from anywhere
  → Increase transfer speed by transferring file to an AWS edge location which will forward the data to the S3 bucket in the target region



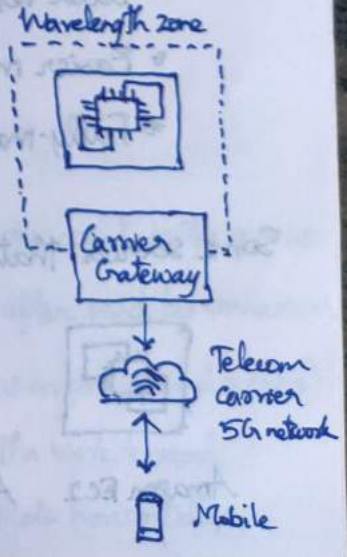
AWS Global Accelerator

→ Improve global application availability and performance using the AWS global network.
→ Leverage the AWS internal network to optimize the route (60% improvement)
→ 2 Anycast IP are created for your application and traffic is sent through Edge
→ The Edge locations send the traffic to your application. No caching is done



AWS WaveLength

→ Wavelength Zones are infrastructure deployments, embedded within the telecommunication providers's datacenters at the edge of 5G network
→ Brings AWS services to the edge of the 5G networks. Eg: EC2, EBS
→ Ultra-low latency applications Through 5G networks
→ Traffic doesn't leave Communicaton Service Provider's (CSP) network
→ High bandwidth and secure connection to parent AWS region
→ No additional charges or service agreements.
→ Use cases: Smart cities, ML-assisted diagnostics, Connected Vehicles, AR/VR Real-time Gaming



AWS Outposts

→ Hybrid Cloud: businesses that keep an on-premises infrastructure alongside a cloud infrastructure

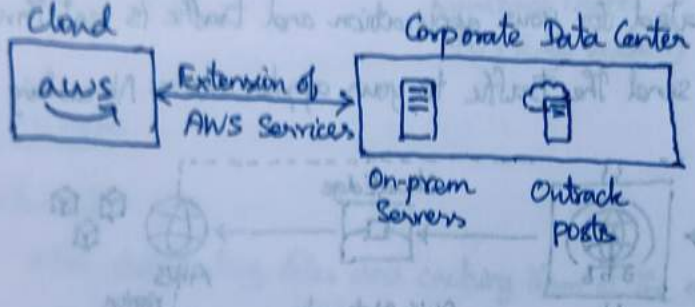
→ Therefore the two ways of dealing with IT systems:

- One for the AWS Cloud (using the AWS console, CLI and AWS APIs)
- One for their on-premises infrastructure

→ AWS Outposts are "server racks" that offers the same AWS infrastructure, services, APIs & tools to build your own applications on-premises just as in the cloud.

→ AWS will setup and manage "Outposts Racks" within your on-premises infrastructure and you can start leveraging AWS services on-premises.

→ You are responsible for the Outposts Rack physical security.



Benefits

- Low-latency access to on-premises systems.
- Local data processing
- Data residency
- Easier migration from on-premises to the cloud
- Fully-managed service

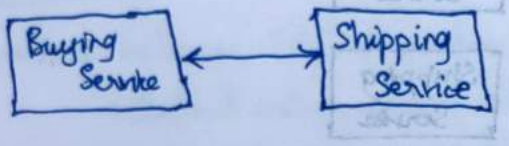
Some services that work on Outposts

Amazon EC2 | Amazon EBS | Amazon S3 | Amazon EKS | Amazon ECS | Amazon RDS | Amazon EMR

Cloud Integrations

- When we start deploying multiple applications, they will inevitably need to communicate with one another.
- There are two patterns of application communication

a) Synchronous communications (application to application)



b) Asynchronous / Event Based (application to queue to application)

→ Synchronous between applications can be problematic if there are sudden spikes of traffic.

→ Eg: What if you suddenly need to encode 1000 videos but usually it is 10?

→ It's better to decouple your applications:

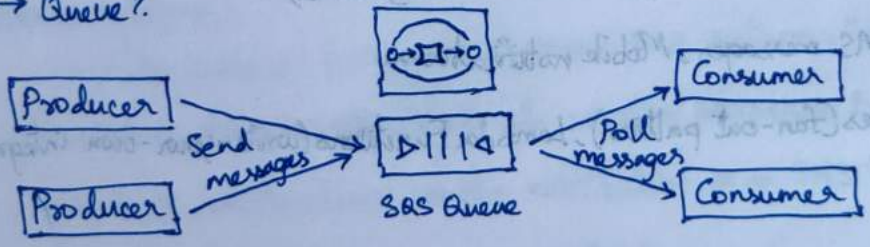
- SQS: queue model
- SNS: pub/sub model
- Kinesis: real-time data streaming model

They can scale independently for our applications

Amazon SQS

→ Simple Queue Service → Standard / FIFO

→ Queue?



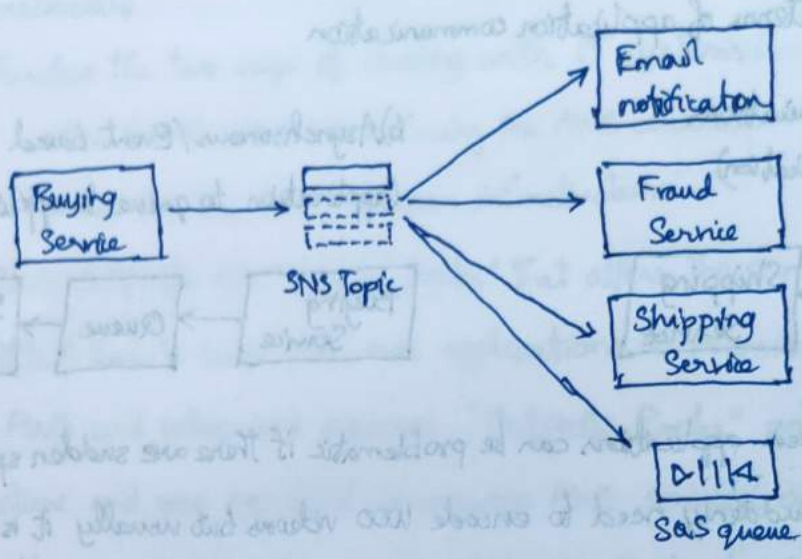
- Oldest AWS offering (over 10 years old)
- Fully managed (~serverless), use to decouple apps
- Scales from 1 to 10000s messages per second
- Default retention of messages: 4 days
  maximum: 14 days

- No limit to messages count in the queue
- Messages deleted after read by consumers
- Low latency (<10 ms on publish and receive)
- Consumers share the work to read messages and scale horizontally

Amazon SNS

- What if you want to send one message to many receivers

Pub/Sub



→ The "event publishers" only sends message to one SNS topic.

→ As many "event subscribers" as we want to listen to the SNS topic notifications

→ Each subscriber to the topic will get all the messages.

→ Up to 10,000,000 subscriptions per topic, 100,000 topics limit

SNS Subscribers

- HTTP/HTTPS (with delivery retries - how many times)
- Emails, SMS messages, Mobile notifications.
- SQS queues (fan-out pattern), Lambda Functions (write-your-own integration)

Amazon Kinesis

→ Managed service to collect, process and analyze real-time big data streaming.

a) Kinesis Data Streams: low latency to ingest data at scale from hundreds of sources.

b) Kinesis Data Firehose: load streams into S3, Redshift, ElasticSearch, etc...

c) Kinesis Data Analytics: real-time analytics on streams using SQL

d) Kinesis Video Streams: monitor real-time video streams for analytics or ML

Cloud Monitoring

- Cloud Watch provides metrics for every services in AWS
- Metric is a variable to monitor (CPU Utilization, NetworkIn...)
- Metrics have timestamps.
- Can create CloudWatch dashboard of metrics.

Important Metrics

→ EC2 instances: CPU Utilization, Status, Checks, Network (not RAM)
- Default metric every 5 minutes.
- Option for Detailed Monitoring ($$$): metrics every 1 minute

→ EBS volumes: Disk Read/Writes

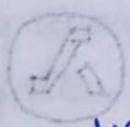
→ S3 buckets: BucketSizeBytes, NumberOfObjects, AllRequests

→ Billing: Total estimated charge (only in us-east-1)

→ Service Limits: how much you've been using a service API

→ Custom metrics: push your own metrics.

Amazon CloudWatch Alarms



- Alarms are used to trigger notifications for any metric.
- Alarm actions:
  → Auto Scaling: increase or decrease EC2 instances "desired" count
  → EC2 Actions: stop, terminate, reboot or recover an EC2 instance.
  → SNS notifications: send a notification to an SNS topic.
- Various options (sampling, %, max, min, etc...)
- Can choose the period on which to evaluate an alarm.
- Example: creates a billing alarm on the CloudWatch Billing metric
- Alarm States: Ok, INSUFFICIENT_DATA, ALARM

CloudWatch Logs

→ It can collect log from
- Elastic Beanstalk: collection of logs from application.
- ECS: collection from containers.
- AWS Lambda: collection from function logs.
- CloudTrail based on filter.
- CloudWatch log agents: on EC2 machines or on-premises servers.
- Route53: Log DNS queries.

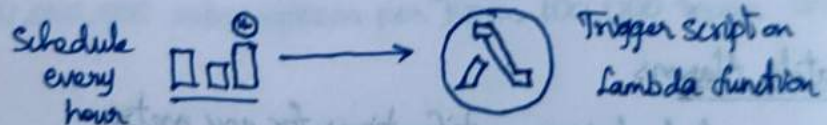
→ Enables real-time monitoring of logs.

→ Adjustable CloudWatch Logs retention.

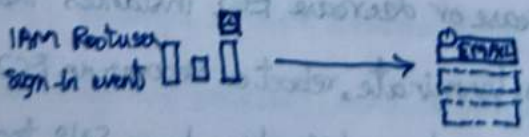
Amazon CloudWatch Events

→ Trigger Lambda functions, send SQS/SNS messages

- Schedule Cron jobs (scheduled scripts)



- Event Pattern: Event rules to react to a service doing something



Amazon EventBridge

- EventBridge is the next evolution of CloudWatch Events
- Default event bus: generated by AWS services
- Partner event bus: receive events from SaaS or applications (ZenDesk, AuthO...)
- Custom event bus: for your own applications
- Schema registry: model event schema

CloudTrail

- Provides governance, compliance and audit for your AWS account.
- CloudTrail is enabled by default.
- Get an history of events / API calls made within your AWS account by:
  - → Console
  - → SDK
  - → CLI
  - → AWS Services.
- Can put logs into CloudWatch or S3
- A trail can be applied to all regions (default) or a single Region.
- If a resource is deleted in AWS, investigate CloudTrail first.

CloudTrail Events

→ Management events

- Operations that are performed on resources in your AWS account.
- Examples:
  - Configuring security (IAM AttachRolePolicy)
  - Configuring rules for routing data (Amazon EC2 CreateSubnet)
  - Setting up logging (AWS CloudTrail CreateTrail)
- By default, trails are configured to log management events.
- Can separate Read Events (that don't modify resources) from Write Events

→ Data Events

- By default data events are not logged (because high volume operations)
- Amazon S3 object-level activity (ex: GetObject, DeleteObject, PutObject)
- AWS Lambda function execution activity (the Invoke API)

→ CloudTrail Insight Events

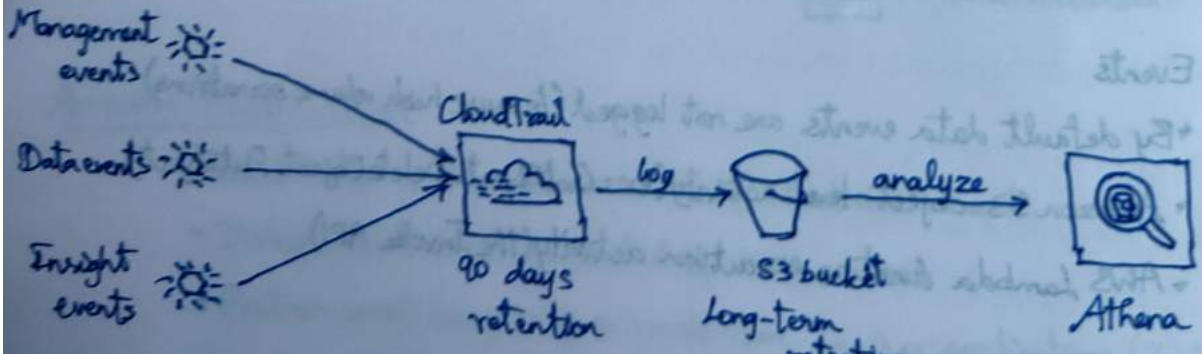
CloudTrail Insights

- Enable to detect unusual activity in your account.
  - Inaccurate resource provisioning.
  - Hitting service limits
  - Bursts of AWS IAM actions
  - Gaps in periodic maintenance activity.
- It analyses normal management events to create a baseline.
- And then continuously analyzes write events to detect unusual patterns.

- Anomalies appear in the CloudTrail console
- Event is sent to Amazon S3
- An EventBridge is generated (for automation needs)

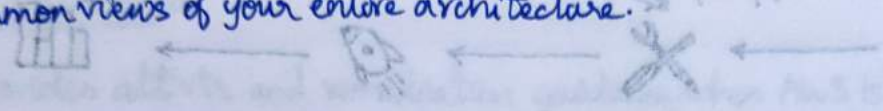
CloudTrail Events Retention

- Events are stored for 90 days in CloudTrail
- To keep events beyond this period, log them to S3 and use Athena



AWS X-Ray - Why?

- Debugging in Production, the good old way:
  - Test locally
  - Add log statements everywhere
  - Re-deploy in production
- Log formats differ across applications and log analysis is hard
- Debugging: one big monolith "easy", distributed services "hard"
- No common views of your entire architecture.

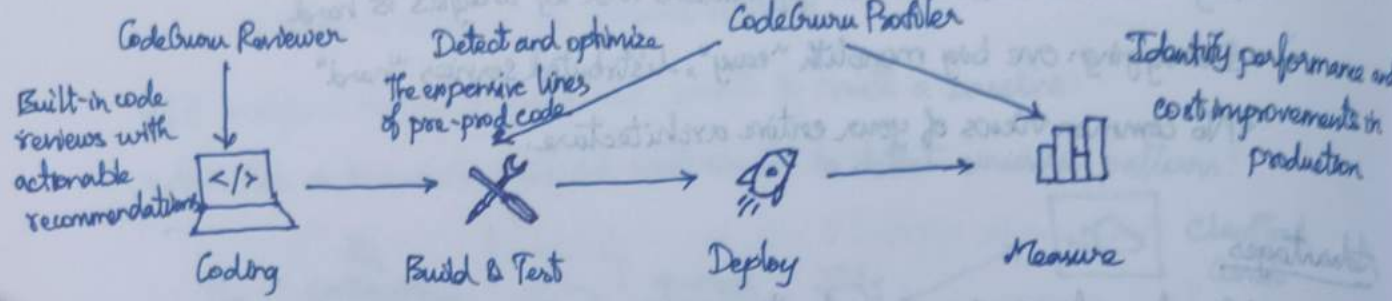


Advantages

- Visual analysis of our applications.
- Troubleshooting performance (bottlenecks)
- Understand dependencies in a microservice architecture.
- Pinpoint service issues.
- Review request behavior.
- Find errors and exceptions.
- Are we meeting time SLA?
- Where am I throttled?
- Identify users that are impacted.

Amazon CodeGuru

- An ML-powered service for automated code reviews and application performance
- Provides two functionalities:
  - CodeGuru Reviewer: automated code reviews for static code analysis (development)
  - CodeGuru Profiler: visibility/recommendations about application performance in runtime



CodeGuru Reviewer

- Identify critical issues, security vulnerabilities and hard-to-find bugs.
- Uses machine learning and automated reasoning.
- Hard-learned lessons across millions of code reviews on 1000s of open source and Amazon repositories.
- Supports Java and Python. Integrates with GitHub, BitBucket and AWS CodeCommit
- Example: common coding best practices, resource leaks, security detection, input validation

CodeGuru Profiler

- Helps understand the runtime behavior of your application.
- Features:
  - Identify and remove code inefficiencies
  - Improve application performance (e.g., reduce CPU utilization)
  - Decrease compute costs
  - Provide heap summary and anomaly detection.
- Support applications running on AWS or on-premise
- Minimal overhead on application.

AWS Status

Service Health Dashboard

- Show all regions, all services health.
- Show historical information for each day.
- Has an RSS feed you can subscribe to

  https://status.aws.amazon.com/

Personal Health Dashboard

- It provides alerts and remediation guidance when AWS is experiencing events that may impact you. Global service https://phd.aws.amazon.com/
- It gives you a personalized view into the performance and availability of the AWS services underlying your AWS resources.
- The dashboard displays relevant and timely information to help you manage events in progress and provides proactive notification to help you plan for scheduled activities.

Monitoring Summary

→ CloudWatch : monitor the performance of AWS services and billing metrics.
→ CloudTrail : audit API calls made within your AWS account
→ CloudTrail Insights : automated analysis of your CloudTrail Events
→ X-Ray : trace requests made through your distributed applications.
→ Service Health : status of all AWS services across all regions.
→ Personal Health : AWS events that impact your infrastructure
→ CodeGuru : automated code reviews and application performance recommendation

VPC & Networking

VPC and Subnets Primer

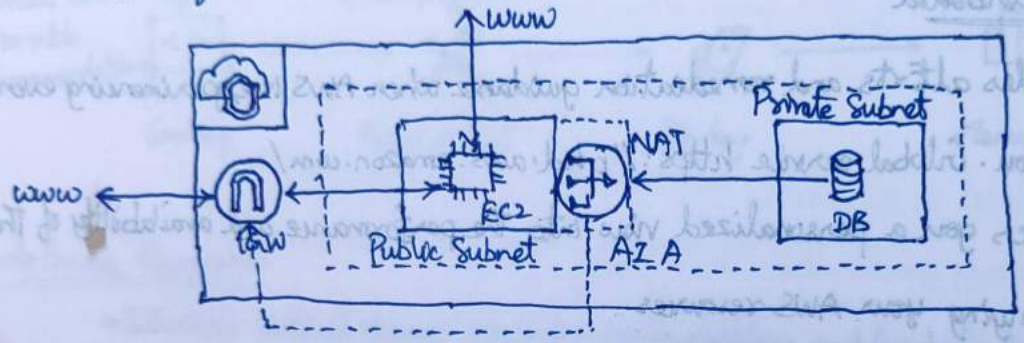
→ VPC - Virtual Private Cloud: private network to deploy your resources (regional resource)

→ Subnets : allow you to partition your network inside your VPC (AZ resource)

i) Public subnet : accessible from the internet

ii) Private subnet : not accessible from the internet

→ To define access to the internet and between subnets, we use Route Tables



Internet Gateway

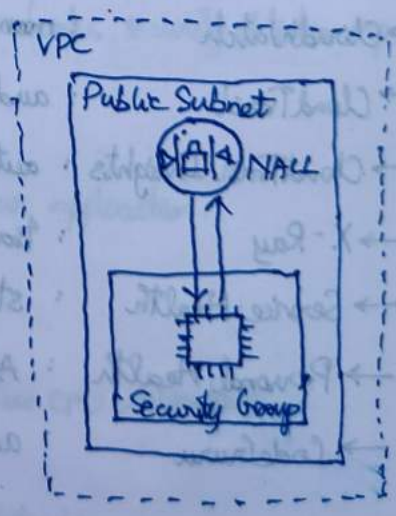
- Helps our VPC instances connect to internet
- Public subnets have a route to the internet gateway

NAT Gateway (AWS-Managed)

* NAT Instances (self-managed) allow private subnets to access internet while being private

Network ACL (NACL)

- Firewall controls traffic from and to subnet.
- Can have ALLOW and DENY rules
- Are attached at the subnet level
- Rules only include IP addresses.



Security Groups

* Firewall controlls traffic from an ENI / an EC2
- Can have only ALLOW rules
- Rules include IP addresses & other security groups

VPC Flow Logs

→ Capture information about IP traffic going into your interfaces:

- VPC Flow logs.
- Subnet Flow logs.
- Elastic Network Interface Flow logs.

→ Helps to monitor and troubleshoot connectivity issues. Example:

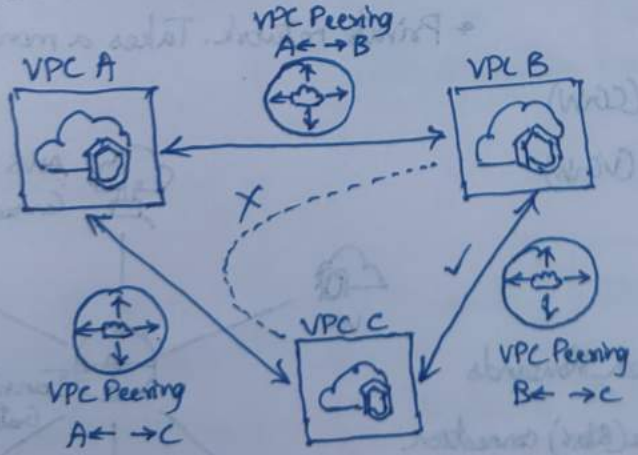
- Subnets to internet
- Subnets to subnets
- Internet to subnets

→ Capture network information from AWS managed interfaces too: ELB, RDS, Aurora...

→ VPC Flow logs data can go to S3/CloudWatch logs.

VPC Peering

- Connect two VPC, privately using AWS network.
- Make them behave as if they were in the same network
- Must not have overlapping CIDR (IP address range)
- VPC Peering connection is not transitive (must be established for each VPC that need to communicate with one another)

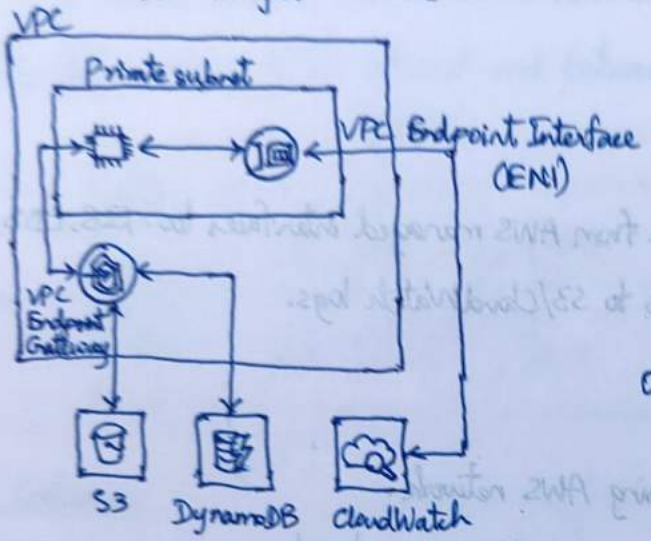


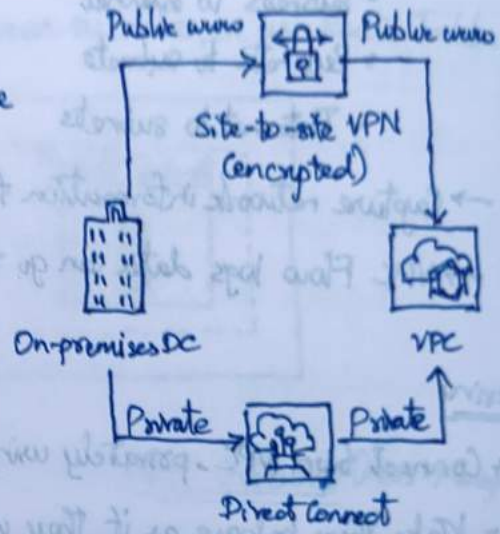
VPC Endpoints

→ Endpoints allow you to connect to AWS Services using a private network instead of public www network.

→ It gives enhanced security and lower latency to access AWS Services.

- VPC Endpoint Gateway : S3 & DynamoDB
- VPC Endpoint Interface : The rest.





Site to Site VPN

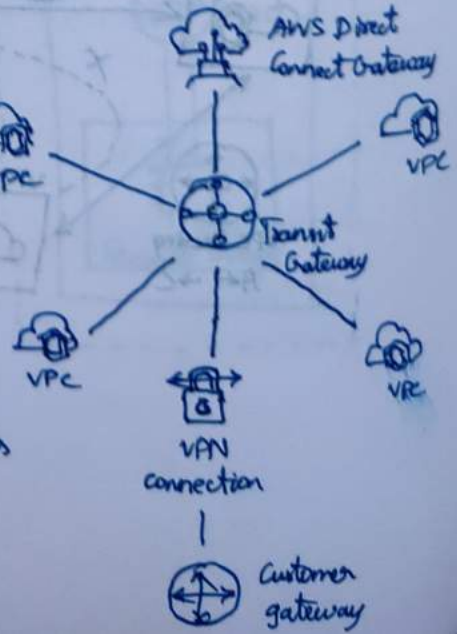
- Connect an on-premises VPN to AWS
- The connection is automatically encrypted.
- Goes over the public internet
- On-premises : Customer Gateway (CGW)
- AWS : Virtual Private Gateway (VGW)

Direct Connect (DX)

- Physical connection between on-prem and AWS
- Connection is private, secure and fast
- Private network. Takes a month to establish

Transit Gateway

- For having transitive peering between thousands of VPC and on-premises, hub & spoke (star) connection
- One single gateway to provide this functionality
- Works with Direct Connect Gateway, VPN connections



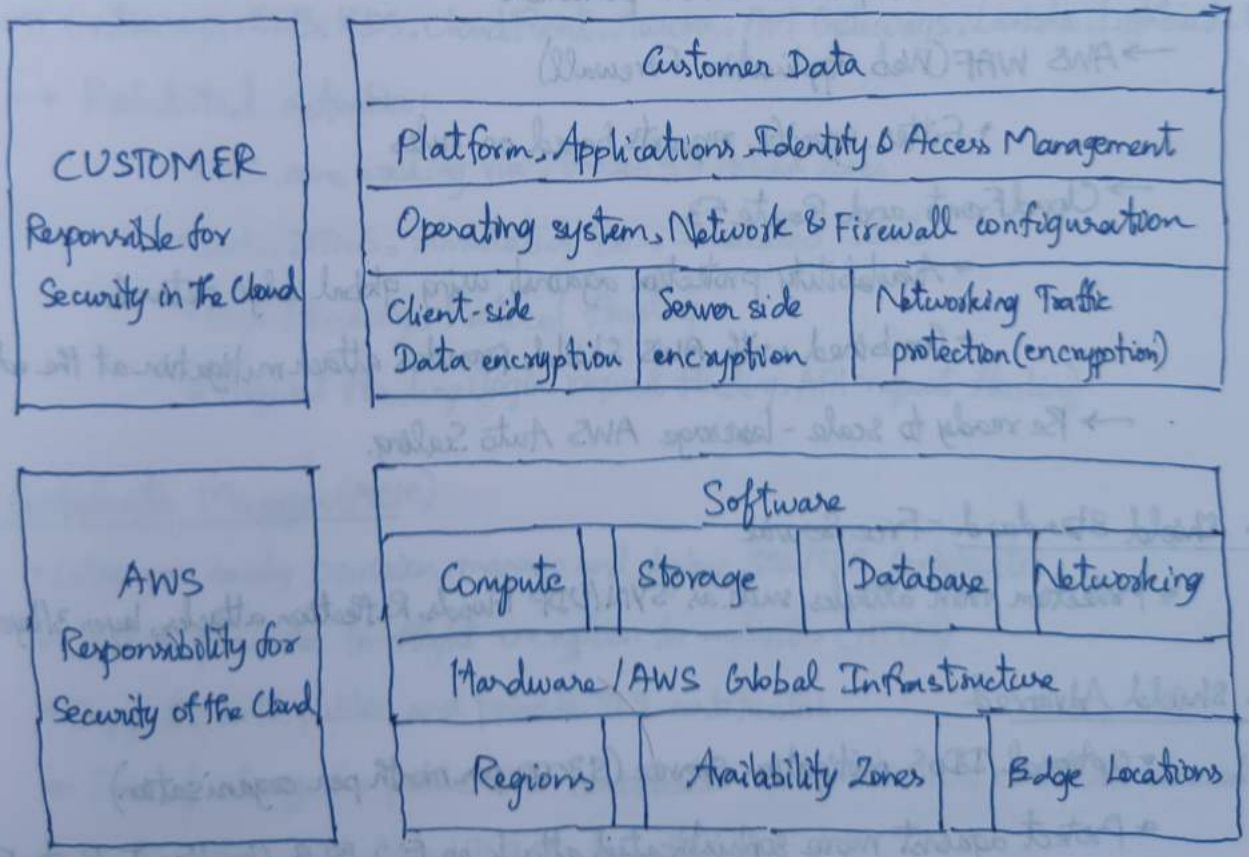
SECURITY & COMPLIANCE

AWS Shared Responsibility Model

→ AWS Responsibility - Security of the Cloud
- Protecting infrastructure (hardware, software, facilities and networking) that runs
- Managed services like S3, DynamoDB, RDS, etc...

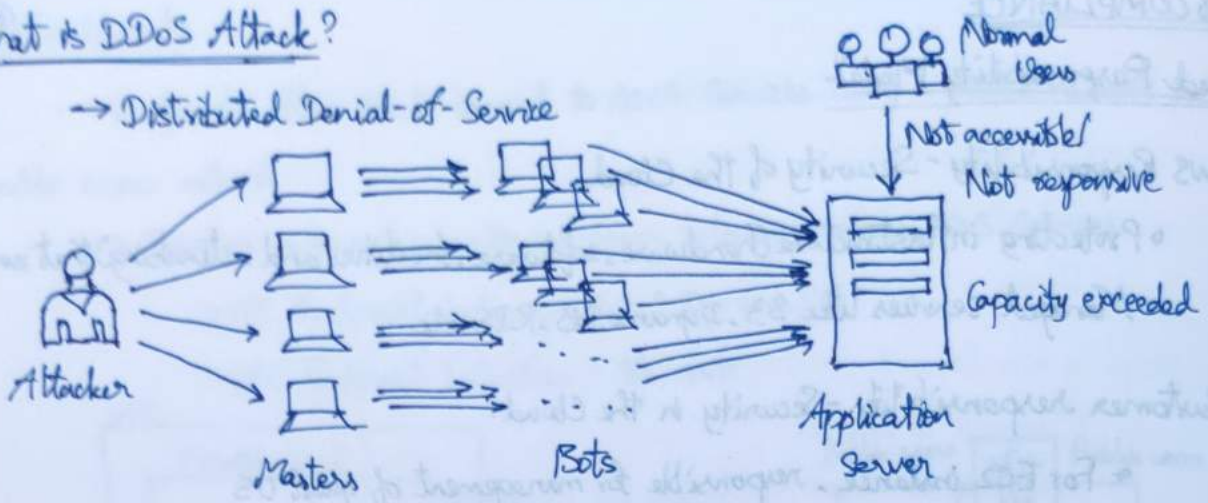
→ Customer responsibility - Security in the Cloud.
- For EC2 instance, responsible for management of guest OS (including security patches and updates) firewall & network configuration, IAM
- Encrypting ~~IAM~~ application data

→ Shared controls
- Patch management, Configuration management, Awareness & Training



What is DDoS Attack?

→ Distributed Denial-of-Service



DDoS Protection on AWS

→ AWS Shield Standard
- Protect against DDoS attack for all customers at no additional cost.

→ AWS Shield Advanced
- 24/7 premium DDoS protection

→ AWS WAF (Web Application Firewall)
- Filter specific requests based on rules

→ CloudFront and Route 53
- Availability protection against using global edge network
- Combined with AWS Shield, provides attack mitigation at the edge

→ Be ready to scale - leverage AWS Auto Scaling.

AWS Shield Standard - Free Service
- Protection from attacks such as SYN/UDP Floods, Reflection attacks, layer 3/layer 4 attacks

AWS Shield Advanced
- Optional DDoS mitigation service ($3000 per month per organisation)
- Protect against more sophisticated attack on EC2, ELB, CloudFront, Route 53
- 24/7 access to AWS DDoS response team (DRP)
- Protect against higher fees during usage spike due to DDoS

AWS WAF

- Protects from web exploits (Layer 7) | Layer 7 - HTTP : Layer 4 - TCP
- Deploy on Application Load Balancer, API Gateway, CloudFront

→ Define Web ACL (Web Access Control List)

- Rules can include IP addresses, HTTP headers, HTTP body or URI strings.
- Protects from common attack - SQL injection and Cross-Site Scripting (XSS)
- Size constraints, geo-match (block countries)
- Rate-based rules (to count occurrences of events) - for DDoS protection.

Penetration Testing

- When you're trying to attack your own infrastructure to test your security.
- We can carryout the testing without prior approval for 8 services:

EC2, NAT Gateways, ELB, RDS, CloudFront, Aurora, API Gateways, Lambda, Lightsail, EBS

→ Prohibited activities:

- DNS zone walking via Amazon 53 Hosted Zones
- DoS, DDoS, Simulated DoS, Simulated DDoS
- Post flooding, Protocol flooding
- Request flooding (login request flooding, API request flooding)

AWS Certificate Manager (ACM)

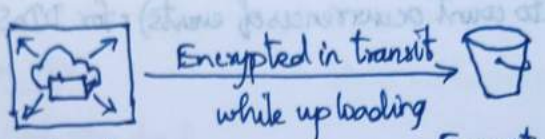
- Let's you easily provision, manage and deploy SSL/TLS Certificates.
- Used to provide in-flight encryption for websites (HTTPS)
- Supports both public and private TLS certificates.
- Free of charge for public TLS certificates. Automatic TLS certificate renewal.
- Integration with (load TLS certificates on)
  - Elastic Load balancers.
  - CloudFront Distributions.
  - APIs on API Gateway

Data at rest vs Data in transit

→ At rest: data stored or archived on a device
- On a hard disk, on a RDS instance, in S3 Glacier Deep Archive

→ In-transit (in motion): data being moved from one location to another.
- Transfer from on-premises to AWS, EC2 to DynamoDB etc..
- Means data transferred on the network

→ We want to encrypt data in both states to protect it. For this we leverage encryption keys



Encrypted at rest on EFS

Encrypted at rest on S3

AWS KMS (Key Management Service)

→ AWS manages the encryption keys for us.

→ Encryption Opt-in:
- EBS volumes : encrypt volumes
- S3 buckets : Server-side encryption of objects
- Redshift database : encryption of data.
- RDS database : encryption of data.
- EFS drives : encryption of data.

→ Encryption Automatically enabled:
- CloudTrail logs
- S3 Glacier
- Storage gateway

CloudHSM

- AWS provisions encryption hardware.
- Dedicated Hardware (HSM → Hardware Security Module)
- You manage your own encryption keys (not AWS)
- HSM device is tamper resistant. FIPS 140-2 Level 3 compliance.

Types of CMK (Customer Managed Keys)

→ Customer Managed CMK
- Create, manage and used by the customers, can enable or disable.
- Possibility of rotation policy (new key generated every year, old key preserved)
- Possibility of bring-your-own-key.

→ AWS Managed CMK
- Created, managed and used on the customer's behalf by AWS.
- Used by AWS services (aws/s3, aws/ebs, aws/redshift)

→ AWS owned CMK
- Collection of CMK that an AWS service owns and manages to use in multiple accounts.
- AWS can use those to protect resources in your account (can't view the keys)

→ CloudHSM Keys (custom keystore)
- Keys generated from your own CloudHSM hardware device.
- Cryptographic operations are performed within the CloudHSM cluster.

AWS Secrets Manager

- Newer service meant for storing secrets.
- Capability to force rotation of secrets every X days.
- Automate generation of secrets on rotation (uses Lambda)
- Integration with Amazon RDS (MySQL, PostgreSQL, Aurora)
- Secrets are encrypted using KMS

AWS Artifact (Not really a Service)

- Portal that provides on-demand access to AWS compliance documentation.
- Artifact Reports - Allows you to download AWS security and compliance documents from third-party auditors, like AWS ISO certifications, Payment Card Industry (PCI) and System and Organization Control (SOC) reports.
- Artifact Agreements - Allows you to review, accept and track the status of AWS agreements such as Business Associate Addendum (BAA) or the Health Insurance Portability and Accountability Act (HIPAA) for an individual account or an organization.

Amazon GuardDuty

- Intelligent threat discovery to protect AWS account.
- Uses Machine learning algorithms, anomaly detection, 3rd party data.
- One click to enable (30 days trial), no need to install software.

→ Input data includes:

- CloudTrail logs : unusual API calls, unauthorized deployments.
- VPC Flow logs : unusual internal traffic, unusual IP address
- DNS Logs : EC2 instances sending decoded data within DNS queries

→ Can setup CloudWatch Event rules to be notified in case of findings.

Amazon Inspector

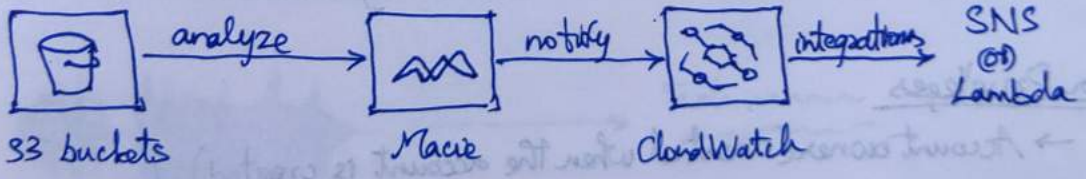
- Automated security Assessments for EC2 instances.
- Analyze the running OS against known vulnerabilities.
- Analyze against unintended network accessibility
- AWS Inspector agent must be installed on OS in EC2 instances.
- After the assessment you get a report with a list of vulnerabilities.

AWS Config

- Helps with auditing and recording compliance of your AWS resources.
- Helps record configurations and changes over time.
- Possibility of storing the configuration data into S3 (analyzed by Athena)
- Questions that can be solved by AWS Config:
  - Is there unrestricted SSH access to my security groups?
  - Do my buckets have any public access?
  - How has my ALB configuration changed over time?
- You can receive alerts (SNS notifications) for any changes.
- AWS Config is a per-region service.
- Can be aggregated across regions and accounts.

Amazon Macie

- Fully managed data security and data privacy service that uses ML and pattern matching to discover and protect your sensitive data in AWS.
- Macie helps identify and alert you to sensitive data, such as PII (personal data)



Amazon Detective

- GuardDuty, Macie and Security Hub are used to identify potential security issues.
- It analyzes, investigates and quickly identifies the root cause of security issues or suspicious activities (using ML and graphs)
- Automatically collects and processes events from VPC Flow Logs, CloudTrail, GuardDuty and creates a unified view.
- Produces visualizations with details and context to get to the root cause.

AWS Security Hub

→ Central security tool to manage security across several AWS accounts and automate security checks.

→ Integrated dashboards showing current security and to quickly take actions.

→ Automatically aggregates alerts in predefined or personal findings formats from various AWS services & AWS partner tools:

- GuardDuty
- Inspector
- Macie
- IAM Access Analyzer
- AWS Systems Manager.
- AWS Firewall Manager.
- AWS Partner Network Solutions.

→ Must first enable the AWS Config service.

AWS Abuse

→ Report suspected AWS resources used for abusive or illegal purposes.

- Spam
- Port scanning
- DoS and DDoS attacks
- Intrusion attempts
- Hosting objectionable or copyrighted content.
- Distributing malware.

Root User Privileges

→ Account owner (created when the account is created)

→ Complete access to all AWS services and resources. Lock root user access keys.

→ Do not use root user for everyday tasks, even administrative tasks.

→ Actions performed only by the root user:

- Change account settings.
- View certain tax invoices.
- Close your AWS account.
- Restore IAM user permissions
- Sign up for GovCloud
- Change or cancel your AWS Support plan
- Register as a seller in the Reserved Instance Marketplace
- Configure an Amazon S3 bucket to enable MFA
- Edit or Delete an Amazon S3 bucket policy that includes an invalid VPC ID or VPC endpoint ID

MACHINE LEARNING

Amazon Rekognition

- Find objects, people, text, scenes in images and videos using ML
- Facial analysis and facial search to do user verification, people counting.
- Create a database of "familiar faces" or compare against celebrities.

→ Use cases:

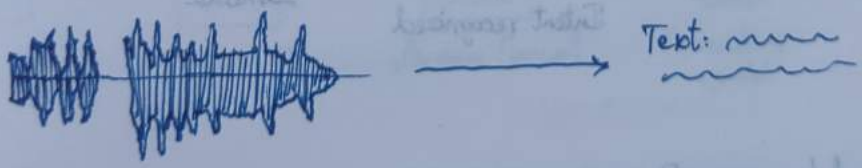
- Labeling
- Content moderation
- Text Detection
- Celebrity Recognition
- Face detection and Analysis
- Face Search and Verification
- Pathing (ex: for sports game analysis)

Amazon Transcribe

- Automatically convert speech to text.
- Uses a deep learning process called automatic speech recognition (ASR)

→ Use Cases:

- Transcribe customer service calls.
- Automate closed captioning and subtitling.
- Generate metadata for media assets to create a fully searchable archive.



Amazon Polly

- Turn text into lifelike speech using deep learning.
- Allowing you to create applications that talk.

Amazon Translate

- Natural and accurate language translation.
- It allows to localize content - such as websites and applications.
- For international users, to easily translate large volumes of text efficiently.

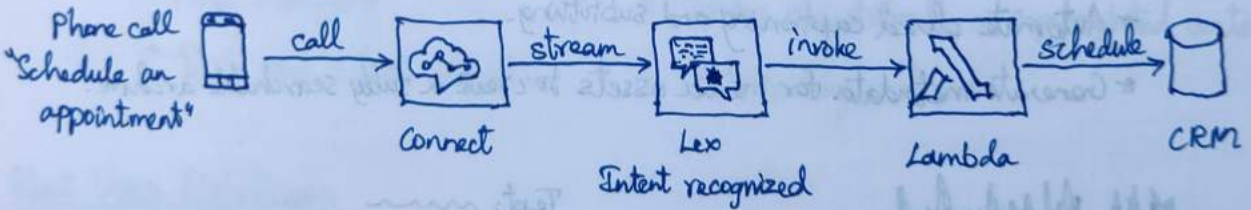
Amazon Lex & Connect

LEX

- Same technology that powers Alexa
- Automatic Speech Recognition (ASR) to convert speech to text.
- Natural language understanding to recognize the intent of text/callers.
- Helps build chatbots, call center bots.

CONNECT

- Receive calls, create contact flows, cloud-based virtual contact center.
- Can receive with other CRM systems or AWS.
- No upfront payments, 80% cheaper than traditional contact center solutions.



Amazon Comprehend

- For Natural Language Processing - NLP. Fully managed and serverless service.

→ Uses machine learning to find insights and relationships in text.

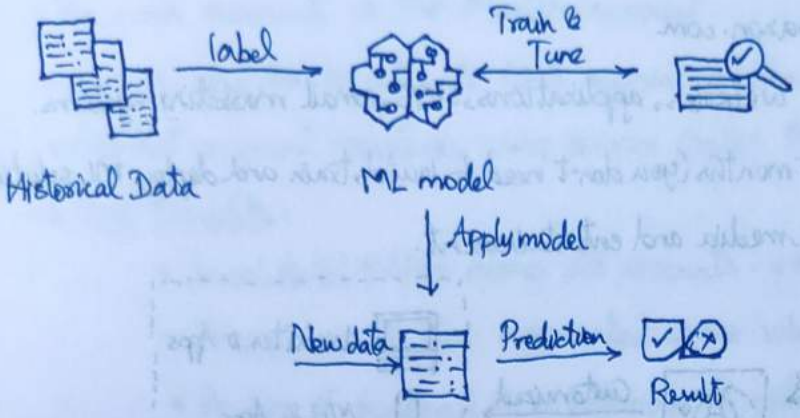
- Language of the text.
- Extracts key phrases, places, people, brands or events.
- Understand how positive or negative a text is
- Analyzes text using tokenization and parts of speech.
- Automatically organizes a collection of text files by topic.

→ Sample Use Case:

- Analyze customer interactions (emails) to find positive or negative experience.

SageMaker

- Fully managed service for developers / data scientists to build ML models.
- Typically difficult to do all the processes in one place + provision servers.

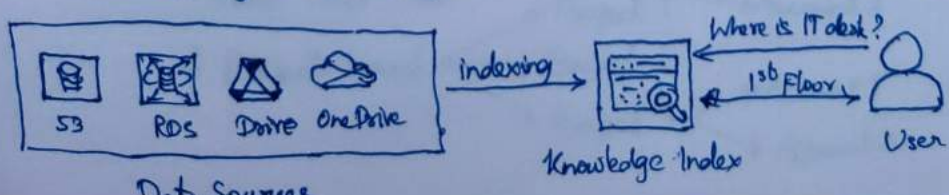


Amazon Forecast

- Fully managed service that uses ML to deliver highly accurate forecasts.
- Example: predict the future sales of a raincoat.
- 50% more accurate than looking at the data itself.
- Reduce forecasting time from months to hours.
- Use Cases: Product Demand Planning, Financial Planning, Resource Planning.

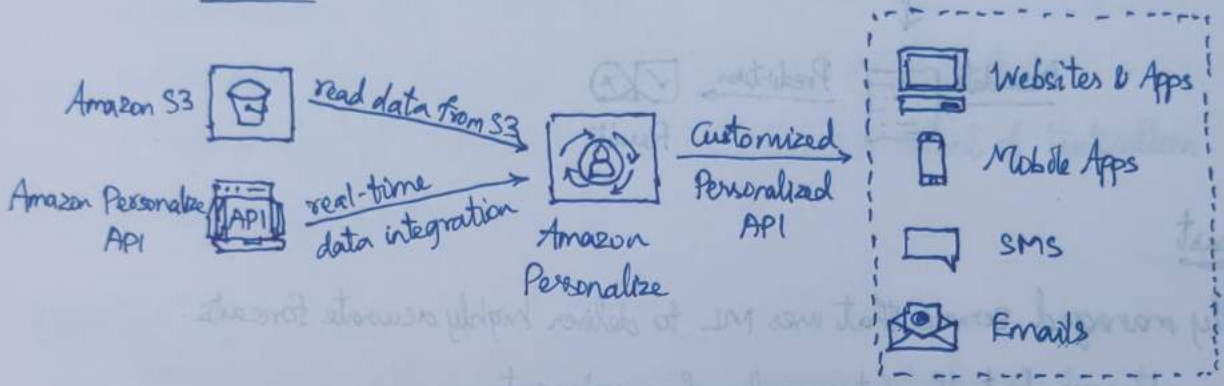
Amazon Kendra

- Fully managed document search service powered by Machine Learning.
- Extract answers from within a document (text, pdf, HTML, PowerPoint, MS Word)
- Natural language search capabilities.
- Learn from user interactions / feedback to promote preferred results (Incremental Learning)
- Ability to manually fine-tune search results (importance of data, freshness, custom..)



Amazon Personalize

- Fully managed ML-service to build apps with real-time personalized recommendations.
- Example: personalized product recommendations/re-ranking, customized direct marketing.
- Same technology by Amazon.com
- Integrates into existing websites, applications, SMS, email marketing systems.
- Implement in days, not months (you don't need to build, train and deploy ML solutions)
- Use cases: retail stores, media and entertainment.



AWS Machine Learning - Summary

- Rekognition : face detection, labeling, celebrity recognition.
- Transcribe : audio-to-text (ex: subtitles)
- Polly : text to audio
- Translate : translations
- Lex : build conversational bots - chatbots.
- Connect : cloud contact center
- Comprehend : natural language processing
- SageMaker : machine learning for every developer and data scientist
- Forecast : build highly accurate forecasts.
- Kendra : ML-powered search engine
- Personalize : real-time personalized recommendations.

ACCOUNT MANAGEMENT, BILLING & SUPPORT

AWS Organizations

- Global service - Allows to manage multiple AWS accounts.
- The main account is the Master account.
- API is available to automate AWS account creation.
- Restrict account privileges using Service Control Policies (SCP)

→ Cost Benefits:

- Consolidated Billing across all accounts - single payment method.
- Pricing benefits from aggregated usage (volume discount for EC2, S3...)
- Pooling of Reserved EC2 instances for optimal savings.

Multi Account Strategies

- Create accounts per department, per cost center, per dev/test/prod, based on regulatory restrictions (using SCP) for better resource isolation (ex: VPC) to have separate per-account service limits, isolated account for logging.
- Multi Account vs One Account Multi VPC
- Use tagging standards for billing purposes.
- Enable CloudTrail on all accounts, send logs to central S3 account.
- Send CloudWatch Logs to central logging account.

Organizational Units (OU)

How can you organize your accounts?

a) Business Unit → Sales, Retail

b) Environment Lifecycle

c) Project based → Project 1 → Account 1, Account 2; Project 2; Project 3 → Account 1, Account 2

Service Control Policies (SCP)

- White-list or Blacklist IAM actions.
- Applied at the OU or Account level.
- Does not apply to the Master Account.
- Applied to all the users and roles of the Account including Root.
- Does not affect service-linked roles. This roles enable other AWS services to integrate with AWS Organizations and can't be restricted by SCPs.
- SCP must have an explicit Allow (does not allow anything by default)

→ Use Cases:

- Restrict access to certain services (for ex: can't use EMR)
- Enforce PCI compliance by explicitly disabling services.

AWS Control Tower

- Easy way to set-up and govern a secure and compliant multi-account AWS environment based on best practices.
- Runs on top of AWS Organizations: Automatically sets up AWS Organizations to organize accounts and implement SCPs (Service Control Policies)

→ Benefits:

- Automate the set up of your environments in a few clicks.
- Automate on-going policy management using guardrails.
- Detect policy violations and remediate them.
- Monitor compliance through an interactive dashboard.

Pricing Models in AWS

- AWS has 4 pricing models:

a) Pay as you go
- Pay for what you use, remain agile, responsive, meet scale demands.

b) Save when you reserve
- Minimize risks, predictably manage buckets, comply with long-term requirements
- Reservations are available (Eg: EC2 Reserved Instances, DynamoDB Reserved Capacity)

c) Pay less by using more
- Volume-based discounts.

d) Pay less as AWS grows

Free services

- IAM
- VPC
- Consolidated Billing
- Elastic Beanstalk
- CloudFormation
- Auto Scaling Groups

} You do not pay for the resources created.

Savings Plan - AWS Cost Explorer

- Commit a certain $ amount per hour for 1 or 3 years.
- Easiest way to setup long-term commitments on AWS.

→ EC2 Savings Plan
- Upto 72% discount compared to On-Demand.
- Commit to usage of individual instance families in a region (e.g. C5 or M5)
- Regardless of AZ, size (m5.xl to m5.4xl), OS or tenancy
- All upfront (huge discount), partial upfront, no upfront.

→ Compute Savings Plan
- Up to 66% discount compared to On-Demand.
- Regardless of Family, Region, size, OS, tenancy, compute options.

Compute Optimizer

- Reduce costs and improve performance by recommending optimal AWS resources.
- Helps you choose optimal configurations and right-size your workloads (over/under provisioned)
- Uses Machine Learning to analyze your resources's configurations and their utilization.

Cloudwatch metrics.

- Lower your costs by up to 25%.
- Recommendations can be exported to S3.

→ Supported resources:

- EC2 instances.
- EC2 Auto Scaling Groups.
- EBS volumes.
- Lambda functions.

Billing and Costing Tools

→ Estimating costs in the cloud

- TCO Calculator
- Simple Monthly Calculator / Pricing Calculator

→ Tracking costs in the cloud

- Billing dashboard
- Cost Allocation Tags
- Cost and Usage Reports.
- Cost Explorer.

→ Monitoring against costs plans:

- Billing Alarms
- Budgets

AWS TCO Calculators

[Deprecated]

- Total Cost of Ownership (TCO) helps by reducing the need to invest in large capital expenditures and providing a pay-as-you-go model.
- Allows you to estimate the cost savings when using AWS and provide a detailed set of reports that can be used in executive presentations.
- Compare the cost of your applications in an on-premises or traditional hosting environment to AWS: Server, Storage, Network, IT Labor.

AWS Pricing Calculator (previously Simply Monthly Calculator)

- Estimate the cost of your architecture solution.

Cost Allocation Tags

- To track AWS costs on a detailed level.
- AWS generated tags
  - Automatically applied to the resource you create.
  - Starts with prefix aws (eg:- aws: createdBy)
- User-defined tags
  - Defined by the user
  - Starts with Prefix user:

→ Tags are used for organizing resources:

- EC2 : instances, images, load balancers, security groups...
- RDS, VPC resources, Route 53, IAM users, etc..
- Resources created by CloudFormation are all tagged the same way.

→ Free naming, common tags are: Name, Environment, Team...

→ Tags can be used to create Resource Groups

- Create, maintain, and view a collection of resources that share common tags.
- Manage these tags using the Tag Editor

Cost and Usage Reports

- Dive deeper into your AWS costs and usage
- It contains the most comprehensive set of AWS cost and usage data available, including additional metadata about AWS services, pricing and reservations. (eg: Amazon EC2 Reserved Instances (RIs))
- It lists the AWS usage for each service category used by an account and its IAM users in hourly or daily line items, as well as any tags that you have activated for cost allocation.
- Can be integrated with Athena, Redshift or QuickSight.

Cost Explorer

- Visualize, understand and manage your AWS costs and usage over time
- Create custom reports that analyze cost and usage data.
- Analyze your data at a high level: total costs and usage across all accounts.
- Or Monthly, hourly, resource level granularity.
- Choose an optimal Savings Plan (to lower prices on your bill)
- Forecast usage up to 12 months based on previous usage.

Billing Alarms

- Stored in CloudWatch us-east-1 but overall worldwide AWS costs shown.
- It's actual cost, not projected costs. Sends an alarm if threshold is exceeded.

AWS Budgets

- Create alarm and send alarms when costs exceeds the budget.
- Types of budgets: Usage, Cost, Reservation.
→ For reserved instances (RI): Track utilization, supports EC2, ElastiCache, Redshift.
- Up to 5 SNS notifications per budget.
- Can filter by: Service, Linked account, Tag, purchase option, AZ, Region, API operation, etc.
- Same options as AWS Cost Explorer!
- 2 budgets are free. Then $0.02/day/budget.

AWS Trusted Advisor

- No need to install anything - high level AWS account assessment
- Analyze your AWS accounts and provides recommendation

a) Cost optimization - low utilization EC2, idle load balancers, under utilized EBS volumes.
b) Performance - High utilization EC2 instances, CloudFront CDN optimization.
c) Security - MFA enabled on root account, IAM key rotation, exposed Access keys.
d) Fault Tolerance - EBS snapshots age, Availability Zone Balance, ELB configuration.
e) Service Limits - Info whether or not reaching the service limit. Increase before reaching limit.

Two Tiers

i) Core Check and recommendations - all customers.
   - Can enable weekly email notification from the console.

ii) Full Trusted Advisor - For Business & Enterprise support plans.
   - Ability to set CloudWatch alarms when reaching limits.
   - Programmatic Access using AWS Support API

Support Plans

a) Basic Support Plan

- Customer Services & Communities.
  - 24x7 access to customer service, documentation, whitepapers, and support forums.
- AWS Trusted Advisor
  - Access to the 7 core Trusted Advisor checks and guidance to provision your resources following best practices to increase performance and improve security.
- AWS Personal Health Dashboard
  - A personalized view of the health of AWS services and alerts when your resources are impacted.

b) Developer Support Plan

→ All basic support plan +

- Business hours email access to Cloud Support Associates.
- Unlimited cases / 1 primary contact.

→ Case severity / response times:

- General guidance : < 24 business hours.
- System impaired : < 12 business hours.

c) Business Support Plan

- Intended to be used if you have production workloads.
- Trusted Advisor - Full set of checks + API access.
- 24×7 phone, email and chat access to Cloud Support Engineers.
- Unlimited cases / unlimited contacts.
- Access to infrastructure Event Management for additional fee

→ Case severity / response times:

- General guidance : < 24 business hours.
- System impaired : < 12 business hours.
- Production system impaired : < 4 hours.
- Production system down : < 1 hour

d) Enterprise Support Plan

- Intended to be used if you have mission critical workloads.
- All of Business Support Plan +
- Access to a Technical Account Manager (TAM)
- Concierge Support Team (for billing and account best practices)
- Infrastructure Event Management, Well-Architected & Operations Reviews

→ Case severity / response times: (same as Business plan)

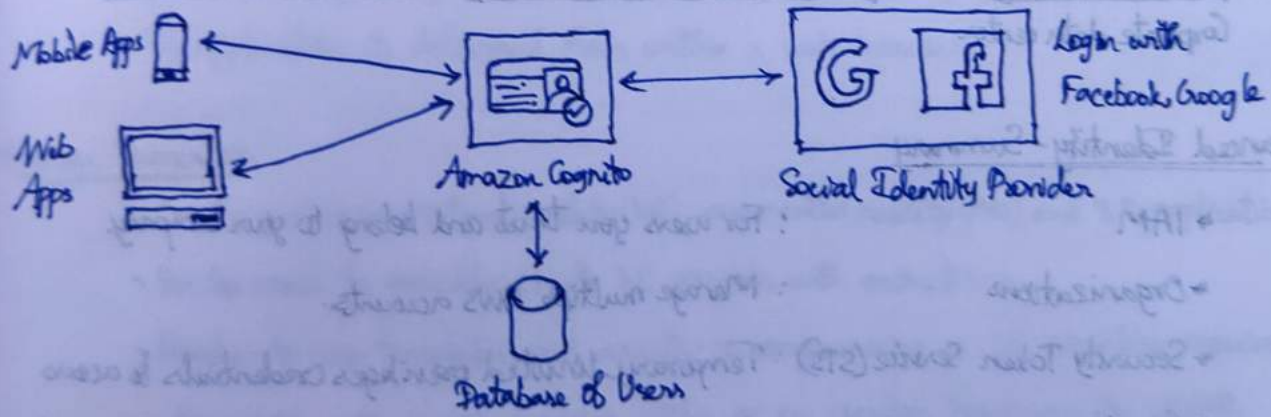
ADVANCED IDENTITY

AWS Security Token Service (STS)

- Enables to create temporary, limited privileges credentials to access your AWS resources.
- Short-term credentials: you configure expiration period.
- Background service but its center of AWS.
- Use cases:
  - Identity federation - manage user identities in external systems, and provide them with STS token to access AWS resources.
  - IAM Roles for cross/same account access.
  - IAM Roles for Amazon EC2 : provide temporary credentials for EC2 instances

Cognito

- Identity for your Web and Mobile application users (potentially millions)
- Instead of creating them an IAM user, you create a user in Cognito



AWS Directory Services

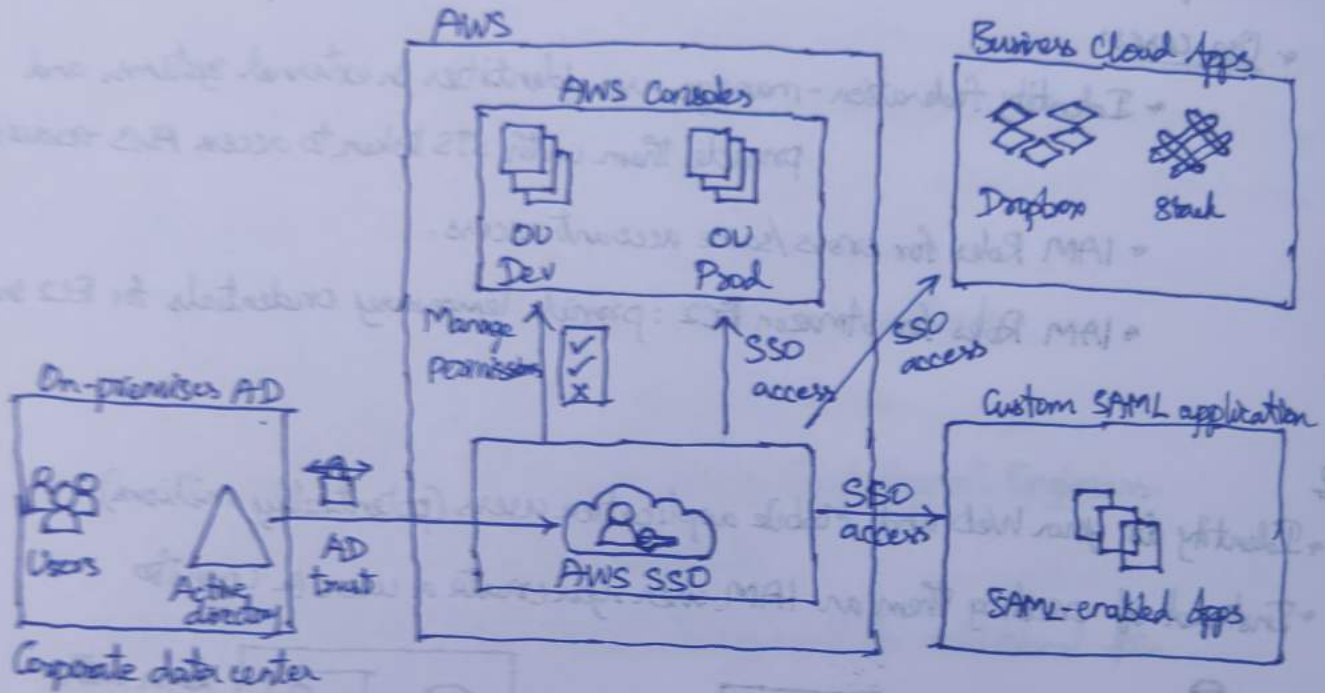
→ AWS Managed Microsoft AD
- Create your own AD in AWS, manage users locally, supports MFA
- Establish "trust" connections with your on-premise AD

→ AD Connector
- Directory Gateway (proxy) to redirect to on-premise AD
- Users are managed on the on-premise AD

→ Simple AD
- AD-compatible managed directory on AWS
- Cannot be joined with on-premise AD

AWS Single Sign-On (SSO)

- Centrally manage one Sign-on to access multiple accounts and 3rd party applications.
- Integrated with AWS Organizations.
- Supports SAML 2.0 markup.
- Integration with on-premise Active Directory.



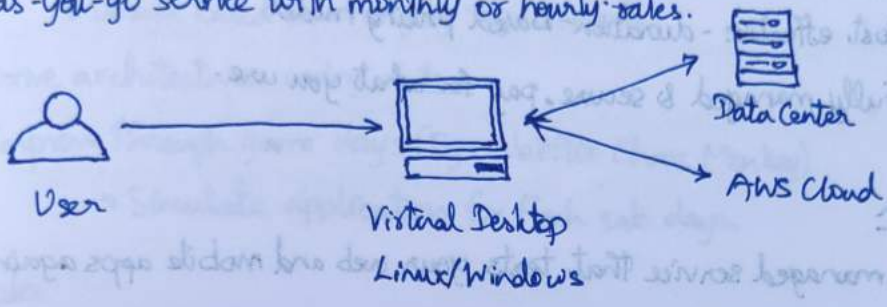
Advanced Identity - Summary

- IAM : For users you trust and belong to your company.
- Organizations : Manage multiple AWS accounts.
- Security Token Service (STS) : Temporary, limited previleges credentials to access
- Cognito : Create a database of users for mobile & web applications.
- Directory Services : Integrate Microsoft Active Directory in AWS
- Single Sign-On (SSO) : One login for multiple AWS accounts & applications.

Other AWS Services

Amazon WorkSpaces

- Managed Desktop as a Service (DaaS) solution to easily provision Windows or Linux desktops.
- Great to eliminate management of on-premise VDI (Virtual Desktop Infrastructure)
- Fast and quickly scalable to thousands of users.
- Secured data - integrates with KMS
- Pay-as-you-go service with monthly or hourly rates.



Amazon AppStream 2.0

- Desktop Application Streaming Service.
- Deliver to any computer, without acquiring, provisioning infrastructure.
- The application is delivered from within a web browser.

Amazon Sumerian

- Create and run virtual reality (VR), augmented reality (AR) and 3D applications.
- Can be used to quickly create 3D models with animations.
- Ready-to-use templates and assets - no programming or 3D expertise required.
- Accessible via a web-browser URLs or on popular hardware for VR/AR

AWS IoT Core

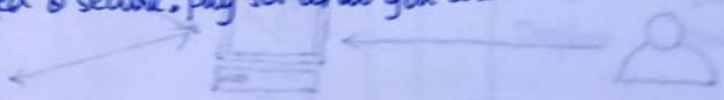
- Internet of Things - the network of internet-connected devices that are able to collect and transfer data. It allows you to easily connect IOT devices to the AWS Cloud.
- Serverless, secured scalable to billions of devices and trillions of messages.
- Your applications can communicate with your devices even when they aren't connected.
- Integrates with a lot of AWS services (Lambda, S3, SageMaker, etc...)
- Build IoT applications that gather, process, analyze and act on data.

Amazon Elastic Transcoder

- It is used to convert media files stored in S3 into media files in the formats required by consumer playback devices (phones etc..)

→ Benefits:

- Easy to use
- Highly scalable - can handle large volumes of media files and large file sizes
- Cost effective - duration-based pricing model.
- Fully managed & secure, pay for what you use.



AWS Device Farm

- Fully managed service that tests your web and mobile apps against desktop browsers, real mobile devices and tablets
- Run tests concurrently on multiple devices (speed up execution)
- Ability to configure device settings (GPS, language, Wi-Fi, Bluetooth)

AWS Backup

- Fully managed service to centrally manage and automate backup across AWS services
- On-demand and scheduled backups.
- Supports PITR (Point-in-time Recovery)
- Retention Periods, Lifecycle Management, Backup Policies
- Cross-Region Backup. Also Cross-Account Backup

CloudEndure

- Disaster recovery - Quickly and easily recover your physical, virtual and cloud-based servers into AWS.
- Eg: Protect most critical databases (including Oracle, MySQL), protect your data from ransomware attacks.
- Continuous block-level replication for your servers.

AWS Architecting & Ecosystem

Well-Architected Framework

- Stop guessing your capacity needs.
- Test systems at production scale.
- Automate to make architectural experimentation easier.
- Allow for evolutionary architectures
  - Design based on changing requirements.
- Drive architectures using data.
- Improve through game days (Eg: Netflix Chaos Monkey)
  - → Simulate application for flash sale days.

Design Principles

→ Scalability: vertical & horizontal.

→ Disposable Resources: servers should be disposable & easily configured.

→ Automation: Serverless, Infrastructure as a Service, Auto Scaling.

→ Loose Coupling
- Monolith are applications that do more and more over time, become bigger.
- Break it down into smaller, loosely coupled components.
- A change or a failure in one component should not cascade to other components.

→ Services, not Servers
- Don't use just EC2
- Use managed services, databases, serverless, etc...

5 Pillars

a) Operational Excellence

b) Security.

c) Reliability

d) Performance Efficiency.

e) Cost Optimization.

→ They are not something to balance, or trade-offs, they're a synergy

a) Operational Excellence

- Includes the ability to run and monitor systems to deliver business value and to continually improve supporting processes and procedures.

→ Design Principles

- Perform operations as code - Infrastructure as code.
- Annotate documentation - Automate the creation of annotated documentation
- Make frequent, small, reversible changes - Incase of failure, you can revert it
- Refine operations procedures frequently - ensure team is familiar
- Anticipate failure. Learn from all operational failures.

Eg:

Prepare	Operate	Evolve
• AWS CloudFormation	• AWS CloudFormation	• AWS CloudFormation
• AWS Config	• AWS Config	• AWS CodeBuild
	• AWS CloudTrail	• AWS CodeCommit
	• AWS CloudWatch	• AWS CodeDeploy
	• AWS X-Ray	• AWS CodePipeline

b) Security

- Includes the ability to protect information, systems and assets while delivering business value through risk assessments and mitigation strategies.

→ Design Principles

- Implement a strong identity foundation - principle of least privilege IAM
- Enable traceability - Integrate logs and metrics to automatically take action.
- Apply security at all layers - Like Edge, VPC, subnet, Load Balancer, OS and [illegible]
- Automate security best practices.
- Protect data in transit and at rest - Encryption, tokenization and access control
- Keep people away from data - Reduce the need for direct access
- Prepare for security events - Run incident response simulations and use tools with automation to increase your speed for detection, investigation and recovery.

Eg:

IAM	Controls	Protection	Data	Incident Response
• AWS-STS	• Config	• CloudFront	• KMS	• IAM
• MFA-Token	• CloudTrail	• VPC	• S3	• CloudFormation
• Organizations	• CloudWatch	• Shield	• ELB	• CloudWatch
		• WAF	• EBS	
		• Inspector	• RDS	

c) Reliability

• Ability of a system to recover from infrastructure or service disruptions, dynamically acquire computing resources to meet demand and mitigate disruptions such as misconfigurations or transient network issues.

→ Design Principles

• Test recovery procedures - Use automation to simulate different failures or recreate

• Automatically recover from failure - Anticipate and remediate failures before they occur

• Scale horizontally to increase aggregate system availability - distribute requests across multiple, smaller resources to ensure that they don't share a common point of failure

• Stop guessing capacity - Maintain the optimal level to satisfy demand - Autoscaling

• Manage change in automation - Use automation to make changes to infrastructure

Eg:

Foundations	Change Management	Failure Management
• IAM	• Auto Scaling	• Backups
• Amazon VPC	• CloudWatch	• CloudFormation
• Service Quotas	• CloudTrail	• S3
• Trusted Advisor	• Config	• S3 Glacier
		• Route 53

d) Performance Efficiency

- Includes the ability to use computing resources efficiently to meet system requirements and to maintain that efficiency as demand changes and technologies evolve.

→ Design Principles

- Democratize advanced technologies - Advance technologies become services.
- Go global in minutes - Easy deployments in multiple regions.
- Use serverless architectures - Avoid burden of managing servers.
- Experiment more often - Easy to carry out comparative testing.
- Mechanical sympathy - Be aware of all AWS services.

Eg:

Selection	Review	Monitoring	Tradeoffs
+ Auto Scaling	+ CloudFormation	+ CloudWatch	+ RDS
+ Lambda	+ News Blog	+ Lambda	+ ElastiCache
+ EBS			+ Snowball
+ S3			+ CloudFront
+ RDS			

e) Cost Optimization

- Includes the ability to run systems to deliver business value at the lowest pri

→ Design Principles

- Adopt a consumption mode - Pay only for what you use.
- Measure overall efficiency - Use CloudWatch
- Stop spending money on data center operations - AWS does the infrastructure part and enables customer to focus on organization projects.
- Analyze and attribute expenditure - Accurate identification of system usage and costs, helps measure return on investment (ROI) - Make sure to use tags.
- Use managed and application level services to reduce cost of ownership As managed services operate at cloud scale, they can offer a lower cost per transaction or ser

Eg: Expenditure Awareness
- AWS Budgets
- Cost and Usage Reports
- Cost Explorer
- Reserved Instance Reporting

Cost-effective Resources
- Spot instance
- Reserved instance
- S3 Glacier

Matching Supply and Demand
- Auto Scaling
- Lambda

Optimizing Over Time
- Trusted Advisor
- Cost and Usage Reports

AWS Well-Architected Tool

- Free tool to review your architectures against the 5 pillars Well-Architected Framework and adopt architectural best practices.
  - How does it work?
    - Select your workload and answer questions.
    - Review your answers against the 5 pillars.
    - Obtain advice: get videos and documentations, generate a report

AWS Marketplace

- Digital catalog with thousands of software listings from independent software developers.
- If you buy through the AWS Marketplace it goes into your AWS Bill.

→ Example:
- Custom AMI (custom OS, firewalls, technical solutions)
- CloudFormation templates
- Software as a Service
- Containers

- You can sell your own solutions on the AWS Marketplace.