IAM (Indetity and access managment)

→ Pay-as-you-go-basis

→ creating users and allowing them to access particular servers

→ Third party application slack, to prevent eavesdropping

1. Principal
2. Authentication
3. Request (API-call behind)
4. Authorization
5. Actions
6. Resources

user

→ can be a person or an application

→ each IAM user is attached with only one AWS account

Groups

→ collection of IAM users

→ permission applied to group, applied to all users in the group

Policies

→ policy sets permissions and controls access

→ stored as JSON documents

There are 2 types

managed policy – multiple entites
inline policy – single entity

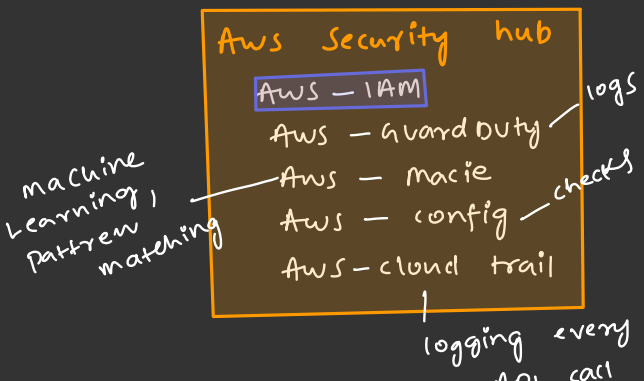
Roles

→ set of permissions that define what actions are allowed

→ role permissions are temporary credentials use resources but don't want to save the password or credential

Features

→ Shared access to aws account
→ granular permissions (least privelaged action)
→ MFA (multi-factor Authentication)
→ Identity Federation (facebook, google)
→ Free to create, use - no charge
→ PCIDSS accompliance
→ password policy (password changes etc --)



(STS- Secure TOken service)

Benefits

- control over data
- Fine grain identity
- continious monitoring for real time security information
- Reduce human config errors
- wide varrity of deeply integrated solutions
- regularly achieves third-party validation for thousands of global compliance
- help you meet security and compliance standards
- most secure global infrastructure
- ability to encrypt data move it and manage retention

→ STS generates Git credentials for IAm users (Temporary credentials - implies)

→ Aws partner to carry out audit on aws account — cross account IAM Role

→ Assume ROLE with SAML - AWS STS service

Billing

→ Pay-as-you-go

→ There will be a forecast of Budget graph based on our past resource consumption.

→ Cost explorer - analyze the cost and usage
  → set time interval and granularity
  → Filter/ group the data
  → Forecast future costs and usage
  → Save the progress

→ **Budgets** - track cost usage from simplest one to most complex use cases

→ stay informed with alerts and reports

→ customize budget as per needs

→ Granular budget time periods

There are 4 types here

cost budget - recieve alerts when the user-defined thresholds are met

usage budget - monitor one ore more specified usage types and alert when the threshold is met (for each service)

Savings plan budget - Track the utilization and recieve alerts when the percentage drops below the threshold

Reservation budget - Track the utilization with reservation and alerts when the percentage drops below the threshold

Alerts - email recipient
Amazon SNS alert (email, sms --)
Amazon chatbot alerts (chat channels like slack and chime)

Cost and usage reports (CUR)

→ accurate budget plans (presice)

→ Product code, usage type, operation

Benefits

→ Access comprehensive Aws and usage info

→ Track your Reserved Instance (RI) and Savings plan usage
→ Leverage integrations with other Analytic services
→ public and private pricing
→ cost category and cost allocation tags
→ we need to create or add existing S3 bucket. It will send reports to that bucket
→ Parquet - easy to extract data using SQL

Pricing calculator

→ view bills, and pay invoices in preffered currency
→ configure cost estimate that fits business or personal needs
→ we have to choose the resource type
→ By giving specifications and resource types we can get the estimated budget

Key managment service (KMS)

→ encryption for both in-transit and at-rest.
→ Two methods for encryption
  - client-side Encryption
  - server-side Encryption
    - → we encrypt our own data and manage the keys
    - → AWS encrypts and manages keys for you (S3, EBS, Redshift provide encryption using KMS behind the scenes)

→ Service that manages Encryption keys

→ It uses Hardware security module (HMS) to store only cmks

→ uses AWS cloud trial to log usage logs for audits and compliance needs

Customer master keys (CMK)

→ A cmk never leave the region

→ can only encrypt max of 4 KB (if 4KB↑ we use data keys)

→ we will ask the cmk to generate data key

→ we call KMS API for data keys (2 will be sent, plaintext data key, encrypted data key)

→ After encryption we will delete plaintext data key (if present easy to decrypt)

→ we can use openSSL or AWS Encryption SDK (outside AWS)

→ To decrypt I take encrypted data key and ask KMS API for decryption algorithm and get the decrypted text

Hardware security module (HSM) - within a vpc

→ computing device that processes cryptographic operations and provides secure storage for cryptographic keys

→ enables to generate and use own encryption keys (complaint with FIPS 140-2 level)

→ automates) administrative tasks like hardware provisioning, software-patching, high availability and back-ups

→ cloud HSM can scale quickly on demand with

no up-front costs

→ Aws cloud HSM interacts vpc instance with SSL

in clusters

→ each cluster is logical HSM

→ To interact HSM cluster, we need Aws cloud HSM client software

→ Operations on one HSM, other HSM's in the same cluster gets updated

→ Offload the SSL/TLS processing for web servers

→ protect the private keys for an issuing certificate Authority

→ enable transparent Data encryption for oracle data base

Benefits

→ easy for load balance and scale

→ Pay by the hour

Simple storage service (S3)

→ infinetly scaling storage

→ Object storage

Use cases

→ Backup and storage

→ Disaster recovery

→ Archive

→ Hybrid cloud storage

→ Application hosting

→ media hosting
→ Data lakes and big data analytics
→ software delivery
→ static website

Buckets

→ files are stored in buckets
→ defined at region-level not global → name
→ no uppercase, no underscore.

Key ← s3://my-bucket/my_folder/file.txt
Key ← Prefix / Object

S3- Security

→ user-based (IAM policies)
→ Resource-based
  → Bucket policies in bucket itself
  → Object access control list (ACL)
  → Bucket access control list (ACL)
→ encrypt object in S3 using encryption keys

S3- Bucket policy

→ JSON based policies
→ S3 bucket for policy

→ user access to S3- IAM permission
→ EC2 instance access- IAM Roles
→ cross account access
→ Public Access
→ To make the objects public we need to uncheck block access and add bucket policy to objects to make them public

S3 - static website hosting

→ we need to make the bucket public (for no issues)
→ enable static weblite hosting
→ if 403 error comes, (policy allow : public reads)

S3 - versioning

→ enabled at bucket level
→ it is best practice to version your buckets
  → protect against unintended delets
  → easy roll back to previous version
→ no versioning will be represented as null
→ Suspending version doesn't delete previous version
→ same name files can be uploaded to update the web page
→ when we accidently delete a object, we can restore it by deleting the object that is of type delete marker

S3 - Replication (Asynchronous replication)

CRR - cross region replication
SRR - same region replication

CRR - compliance, lower latency access, replication cross accounts
SRR - log aggregation, live replication between production and test accounts

→ After you enable replication, only new objects are replicated

→ For existing objects we can perform batch operations

→ Can replicate delete markers from source to target

→ Deletions with a version ID are not replicated

→ No chaining of replication

→ To enable replication, versioning should be applied on both the buckets

S3 - Storage classes

→ Standard - General purpose

→ Standard - Infrequent access

→ One zone - infrequent access

→ glacier instant retrival

→ glacier flexible retrival

→ glacier deep archive

→ Intelligent tiering

→ can move between classes manually or using S3 life cycle configurations

Durability or Availability

S3 standard - general purpose

→ frequently accessed data

→ low latency high throughput

→ Sustain 2 concurrent facility failures

Use cases Big data analytics, mobile & gaming applications, content distribution

S3 - Infrequent access

→ less frequently accessed, require rapid access
→ less cost than above
→ use cases: disaster recovery, backups

one-zone infreament access

→ high durability in single AZ, data lost when AZ is destroyed
→ use cases: storing secondary backup copies of on-premises data, or data to recreate

Glacier

→ low-cost object storage meant for archiving/backup
→ price for storage + object retrivial cost

Instant retrival

→ millisecond retrial (90 days in a year)
→ low latency retrival

Flexible retrival

→ expited (1-5 mins) (90 days)
standard (3-5 hrs)
Bulk (5-12 hrs) -free

Deep archive - long term storage - cheapest

→ Standard (12 hrs), Bulk (48 hrs) (180 days)

Intelligent tiering

→ automatically move between classes

→ have to pay for monthly monitoring and auto-tiering

→

Auto
- IA tier – object not accessed for 30 days
- Archive instant – object not accessed for 90 days
- Archive Access – (config) 90 – 700+ days
- Deep Archive – (config) 180 – 700+ days

→ alternative to Intelligent tiering is manual life cycles (free cost)

	Standard	Intelligent-Tiering	Standard-IA	One Zone-IA	Glacier Instant Retrieval	Glacier Flexible Retrieval	Glacier Deep Archive
Storage Cost (per GB per month)	$0.023	$0.0025 - $0.023	%0.0125	$0.01	$0.004	$0.0036	$0.00099
Retrieval Cost (per 1000 request)	**GET:** $0.0004 **POST:** $0.005	**GET:** $0.0004 **POST:** $0.005	**GET:** $0.001 **POST:** $0.01	**GET:** $0.001 **POST:** $0.01	**GET:** $0.01 **POST:** $0.02	**GET:** $0.0004 **POST:** $0.03 **Expedited:** $10 **Standard:** $0.05 **Bulk:** free	**GET:** $0.0004 **POST:** $0.05 **Standard:** $0.10 **Bulk:** $0.025
Retrieval Time	Instantaneous					**Expedited** (1 – 5 mins) **Standard** (3 – 5 hours) Bulk (5 – 12 hours)	**Standard** (12 hours) **Bulk** (48 hours)
Monitoring Cost (pet 1000 objects)		$0.0025					

	Standard	Intelligent-Tiering	Standard-IA	One Zone-IA	Glacier Instant Retrieval	Glacier Flexible Retrieval	Glacier Deep Archive
Durability	99.999999999% == (11 9's)						
Availability	99.99%	99.9%	99.9%	99.5%	99.9%	99.99%	99.99%
Availability SLA	99.9%	99%	99%	99%	99%	99.9%	99.9%
Availability Zones	>= 3	>= 3	>= 3	1	>= 3	>= 3	>= 3
Min. Storage Duration Charge	None	None	30 Days	30 Days	90 Days	90 Days	180 Days
Min. Billable Object Size	None	None	128 KB	128 KB	128 KB	40 KB	40 KB
Retrieval Fee	None	None	Per GB retrieved	Per GB retrieved	Per GB retrieved	Per GB retrieved	Per GB retrieved

S3 - life cycle Rules

Transition actions: config obj to transition to another storage class

expiration actions: config objects to expire after some time

→ can be used to delete old versions of files

→ can be used to delete incomplete multi-part uploads

S3 - Analytics

→ helps you decide when to transition objects to the right storage class

→ does not work for one zone-IA or glacier

→ Report updated daily

→ 24 to 48 hrs to start seeing analytics

S3 - Requester pays

→ if a person (requester) has to pay networking / transfer costs then Requester pays bucket

→ helpful to share large datasets)

→ the requester must be authenticated in AWS

S3 event notifications (SNS, SQS, λ fn)

use case: generate thumbnails of images uploaded to S3

S3 → putobject → λ fn → Thumbnail

→ can be created as desired (sec / min ↑)

Event Bridge (18 AWS Services)

→ multiple destinations

→ event-bridge capabilities - Archive, Replay events, Reliable delivery

S3 Performance

→ multi-part upload (>100mB >5GB)

→ S3 Transfer Acceleration

→ Increase transfer speed by transferring file to AWS edge location which will forward data to S3 bucket

→ compaitable with multi-part upload

→ Byte-Range fetches (partial access of file)

S3 Select & glacier-Select

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

→ less network transfer, less CPU cost client-side

→ Same thing on many obj's use batch operation

→ can use S3 inventory to get object list

Elastic compute cloud (EC2)

→ infrastructure as a service

→ Renting virtual machines (EC2)

→ storing data on virtual drivers (EBS)
→ Distributing load across machines (ELB)
→ scaling services using Auto-scaling (ASG)

Config options

→ OS → CPU → RAM → EBS | EFS
→ Ec2 store → Network card → Firewall rules
→ Bootstrap script

User data - EC2

→ Bootstrap instances using Ec2 user data script
→ Bootstraping: lauching commands when machine start!
→ Ec2 user data is used to automate
  → installing updates
  → installing softwares
  → downloding files from internet
→ Ec2 data script runs with root user
→ public IPV4 will change freaquently

Ec2 - Instance types

→ General Purpose
→ Compute optimized
→ memory optimized
→ Accelerated computing
→ storage optimized
→ Instance features
→ measuring instance perfomance

→ General purpose

→ great for diversity of workloads such as web servers or code repositories

→ balance between compute, memory and networking

→ compute optimized

→ good for computing tasks that require high performance
  → Batch processing workloads
  → media transcoding
  → high performance web servers
  → high performance computing (HPC)
  → scientific modeling
  → machine learning
  → Dedicated gaming servers

→ memory optimized (volatile)
  → fast performances for workloads that process large data sets in memory

use cases: High performance, relational / non-relation databases
  → distributed web scale cache stores
  → In memory database optimized for BI
  → Applications performing real-time processing of big unstructured data

→ storage optimized (storing, non-volatile)
  → great for storage-intensive tasks that require high, sequential read and write access to large data sets on local storage

use cases: High frequency OLTP systems

→ Relational & nosql databases

→ Cache for in-memory databases (Redis)

→ Data warehousing applications

→ Distributed file systems

Instance	vCPU	Mem (GiB)	Storage	Network Performance	EBS Bandwidth (Mbps)
t2.micro	1	1	EBS-Only	Low to Moderate	
t2.xlarge	4	16	EBS-Only	Moderate	
c5d.4xlarge	16	32	1 x 400 NVMe SSD	Up to 10 Gbps	4,750
r5.16xlarge	64	512	EBS Only	20 Gbps	13,600
m5.8xlarge	32	128	EBS Only	10 Gbps	6,800

EC2- Security groups

→ it controls how traffic is allowed into or out of our EC2 instances

→ They only contain rules (rules can be ref by IP or by security group)

→ Firewall

→ Access to ports

→ Authorised IP ranges

→ one security group can be attached to multiple instances

→ Regional service

→ if application gives "connection refused" error, it's an application error or it's not launched

→ All inbound traffic is blocked by default
→ All outbound traffic is authorised by default

SSH - 22 — log into a linux instance
FTP - 21 — file share
SFTP - 22 — " using SSH
HTTP - 80 — access unsecured websites
HTTPS - 443 — access secured websites
RDP - 3389 — log into windows instance

→ windows < 10 SSH X (SSH using putty only)

EC2 Instance connect

→ within the browser (no key pairs)
→ works only with Amazon-Linux-2 AMI
→ make sure port 22 is opened.

Instance purchasing - EC2

→ on demand instance - short workload, predictable pricing, pay by second
→ reserved (1 & 3 years)
  → Reserved instances - long work locals
  → convertible Reserved instances - long workloads with flexible instances
→ savings plans (1 & 3 years) - commitment to an amount of usage, long workload

→ Spot instances – short workloads, cheap, can lose instances (less reliable)

→ Dedicated hosts – book an entire physical server, control instance placement

→ Dedicated instances – no other customers will share your hardware

→ Capacity Reservations – reserve capacity in a specific AZ for any duration

→ EC2 on demand
  → highest cost, no upfront payment
  → short term, un-interrupted workloads

→ Reserved instances
  → 72% discount
  → discount based on time period and upfront payment
  → scope is Regional or Zonal
  → steady state usage applications like databases

→ Convertible Reserve instance
  → can change EC2 instance type, instance family, OS, scope and tenancy
  → upto 66% discount

→ Savings plan
  → discount on long-term usage (72%)
  → commit to a certain type of usage

→ locked to a specific instance type & AWS region

→ Flexible with instance size, OS, Tenancy

→ **spot instances**

→ most discounted (90%)

→ can lose at any point of time if your max price is less than current spot price

→ workloads that are resilient to failure

→ Batch jobs

→ Data analysis

→ Image processing

→ Any distributed workloads

→ workloads with a flexible start and end time

→ NO suitable for critical jobs or databases

→ **Dedicated hosts**

→ physical server fully dedicated

→ Pay per second | reserved (1-3 years)

→ most expensive

→ useful for software that have complicated licensing model

→ companies that have compliance needs

→ **Dedicated instances**

→ Instances run on hardware that's dedicated to you

→ may share hardware with other instances

in same account

→ no control over instance placement

→ capacity Reservations

→ on-demand instance in specific AZ for any duration

→ no billing discount

→ always have EC2 capacity when needed

→ create/cancel anytime

→ on-demand rate whether you run or not

→ suitable for short-term, uninterrupted workloads that needs to be in a specific AZ

→ you can only cancel spot instance requests that are open, active or disabled.

→ cancelling a spot request does not terminate instances

→ you must first cancel a spot request, and then terminate the associated spot instances

Spot fleets

→ spot fleets = set of spot instances + on-demand instances

→ spot fleet will try to meet the target capacity with price constraints

→ Private Ip machines connect to WWW using a NAT + internet gateaway

Elastic Ip's

→ if fixed Ip for instance is needed, we need an Elastic Ip
→ can attach it to one instance at a time
→ not a very good practice to use Elastic Ip (Best is Load balancer)

Placement group

→ control over EC2 instance placement strategy
  → cluster
  → spread
  → partition

cluster placement group

→ high perfomance and great throughput with low latency
→ Great network (10Gbps)
→ if rack fails, all instances fail at the same time

Spread placement group

→ spans across diff AZ
→ Reduced risk
→ maximize high availability
→ Disaster managment, each instance is isolated from the other

partition placement group

→ can span across multiple Az in same region
→ one rack failure affects many instances but not the other rack
→ HDFS, HBase, cassandra, Kafka.

Elastic network Interface (ENI)

→ logical component in a vpc that represents virtual network card
  → attributes
    → private IPv4, one or more 2nd Ip
    → elastic Ip for every IPv4
    → public IPv4
    → one or more security groups
    → A mAc address
  → Bound to specific Az

Hibernation - EC2

→ The Ram state is preserved

use cases
  → Long-running processing
  → Saving the RAM state
  → services that takes time to initialize

→ cannot be hibernated more than 60 days

EBS (Elastic block store)

→ It cannot be used directly
→ can be combined with EC2, RDS, ECS, RCDIS

→ For Ec2 we need harddisk to the server
→ harddisk will act as EBS
→ It is a block-level storage
→ OS launching can only be done by EBS
→ recommended for data that requires frequent and granular updates
→ An EBS volume can be attached to single instance at a time
→ EBS allows encryption using Amazon EBS encryption feature
→ EBS volumes can be backed up by creating snapshot of the volume, which is stored in S3
→ EBS created in one Az can attach to an instance in the same Az
→ To make volume available outside of Az create a snapshot and restore the snap shot to new volume anywhere in that region

EBS volume types

→ General purpose SSD volumes (gp2)
→ provisioned IOPS SSD volumes (io1)
→ magnetic volumes (standard)

EBS SSD Volume Types

Characteristic	General Purpose (SSD)	Provisioned IOPS (SSD)
Use Cases	• System boot volumes • Virtual desktops • Small to medium DBs • Dev and test	• I/O intensive • Relational DBs • NoSQL DBs
Volume Size	1 GB - 16 TB	4 GB - 16 TB
Max. Throughput	160 MB/s	320 MB/s
Max IOPS / volume	10,000	20,000
Max IOPS / instance	65,000	65,000
Max. Throughput / instance	1,250 Mbps	1,250 Mbps
API Name	gp2	io1

amazon web services

EBS HDD Volume Types

Characteristic	Throughput Optimized HDD	Cold HDD	Magnetic
Use Cases	• Infrequent Data Access • Streaming • Big Data • Logs • Cannot be a boot volume	• Throughput-oriented for large volumes of data • Lowest storage cost is important • Cannot be a boot volume	• Infrequent Data Access
Volume Size	500 GB - 16 TB	500 GB - 16 TB	1 GB – 1 TB
Max. Throughput	500 MB/s	250 MB/s	40 – 90 MB/s
Max IOPS / volume	500	250	40 – 200
Max IOPS / instance	65,000	65,000	48,000
Max. Throughput / instance	1,250 Mbps	1,250 Mbps	1,250 Mbps
API Name	st1	sc1	standard

amazon web services

General purpose SSD volumes (gp2)

→ cost-effective storage that is ideal for broad range of workloads
→ Single digit millisecond latencies

Magnetic volumes (standard)

→ lower cost per gigabyte
→ ideal for workloads performing sequential reads, where data is accessed infrequently
→ They can be striped together in a RAID configuration for larger size and greater Perfomance

EBS creation

→ create new volumes
→ Restore volumes from snapshots

EBS deletion

→ will wipe out data
→ can be backed up before deletion using EBS snapshot
→ snapshots can be used to create new volumes, increase the size of volumes or replicate data across AZ's

Benefits

→ Data availability

→ EBS volume is automatically replicated in an AZ to prevent data loss of failure of any single hardware component

→ Data persistance
  → EBS volume persists independently of running life of EC2 instance
  → Root EBS volume is deleted, by default, on instance termination but can be modified by changing the Delete on Termination flag

→ Data Encryption (EBS encryption feature)

EBS

→ Access to raw unformatted block level storage

→ Persistent storage

→ Detailed metrics captured via cloud watch

Entreprises, Applications, gaming, social, mobile, education, marketing analytics

Standard volumes

→ workloads with low or moderate IOPS needs and occasional bursts

→ File server, log processing, website, analytics, boot etc.

Provisoned IOPS volumes

→ Transactional workloads requiring consistent IOPs
→ Business applications, mongoDB, SQL server, mysql, postgres, oracle
→ 10% of IOPs perfomance 99.9% of time over a given year
→ good latency after PIOPS

consistent snapshot perfomance

To minmize impact of snapshots on perfomance
→ create snapshots from a read replica of your data
→ plan snapshots during off-peak usage
→ use cross-region snapshot copy to keep distant regional copies for low latency access
→ For workloads targeting above 100 IOPS use PIOPS.

Elastic Load balancer (ELB)

→ It distributes and manages the incoming traffic load among several devices to improve network perfomance.
→ distributes client traffic across servers
→ improves perfomance of application

→ Routing protocols for load balancer are
  → Round- Robin
  → Server with least no. of requests

Types of Load balancer

→ Classic Load balancer
  → Traffic across multiple instances in multiple AZ
  → Supports both EC2 classic EC2-VPC
  → increases availability of application by sending traffic to healthy instances
  → Supports HTTP, HTTPS, TCP and SSL Listeners
  → Supports sticky sessions

Limitations
  → max 20 per region
  → 50 listeners
  → 8 security groups
  → Registered instances 1000
  → Subnets per AZ is 1.

→ Network Load balancer
  → It handles sudden and violates traffic across EC2 instances in order to avoid any latency
  → new layer 4 load balancing platform
  → connection based load balancing
  → Supports TCP Protocol

→ can handle millions of requests/sec

Limitations

→ 20 LB per region
→ 3000 target groups per region
→ 50 listeners
→ 200 targets per AZ with cross-zone load balancing disabled
→ 200 targets with cross-zone load balancing enabled
→ subnets per AZ 1

→ Application Load balancer

→ will distribute traffic based on port no (layer 7 of OSI) (content)
→ Reduces hourly cost
→ supports, web sockets, HTTP and HTTPS
→ supports micro services and container based application including deep integration with Ec2 container service

Limitations

→ 20 LB per region
→ TG 3000 per region
→ https 25 certificates (remaing all same)

Key benefits of Load balancer

→ Better performance
→ supports monitoring high check independently

→ supports registering target IP

→ supports host and path based routing

→ Route requests to multiple application hosted on single EC2 instance

→ Gateway Load balancer

→ easy to deploy, scale and manage third-party virtual appliances

→ eliminates potential point failures and increases availability

→ Scale your virtual appliancy automatically

→ monitor continious health and performance metrics (Firewall)

→ Runs within one AZ

Limitations

→ 20 LB per region

→ LB per vpc - 10.

Web application Firewall (WAF)

→ It helps address critical attack surface in Application security Ecosystem

→ Application vulnerability

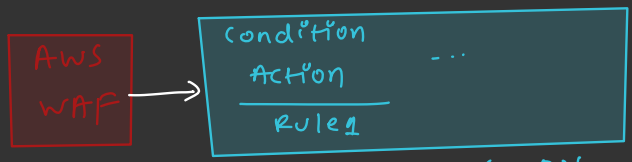
→ zero-days

→ Denial of service

→ BOT attacks

→ multi-layered security

→ no operational overhead
→ Default and customizable security
→ Frictionless set up



web access control list (web ACL)

→ monitors HTTP and HTTPS requests that are forwaded

→ web ACL capacity unit (WCU) is a dimension that is used to calculate and control the operating resources that are used to process your rules within a web ACL. WCU does not represent a rule for pricing consideration

→ can use Amazon cloud watch / Amazon Kinesis Firehose (request details)
Amazon cloud watch - incoming traffic metrics

→ Aws managed rules address issues like the open web application security project (OWASP) security risks.

→ charges per web ACL's created, no. of rules added per web ACL and the no. of web requests that you recieve

Virtual private networks (VPC)

CIDR - subnet mask

→ It allows part of underlying IP to get additional next values from base IP

Public vs private IP (IPv4)

→ Private IP → 10.0.0.0 — 10.255.255.255
Big networks
→ 172.16.0.0 — 172.31.255.255
Aws default vpc in range
→ 192.168.0.0 - 192.168.255.255
home networks

→ All the rest are public IP's

Default VPC on AWS

→ has internet connectivity and all Ec2 instances inside it have public IPv4 address.
→ we also get a public and private IPv4
→ VPC's are regional in nature.
→ can have multiple VPC's in Aws region (5)
→ max CIDR Range per VPC is 5
min size is /28 (16 ip addresses)
max size is /16 (65536 ip addresses)
→ your VPC CIDR should not overlap with other networks
→ 5 ip address are reserved (first 4 and last 1)

→ For CIDR block 10.0.0.0/24

10.0.0.0 – Network Address

10.0.0.1 – VPC router

10.0.0.2 – mapping to amazon-provided DNS

10.0.0.3 – AWS future use

10.0.0.255 – Network Broadcast address

→ web servers should be in public subnet and databases in private subnet

Internet Gateway (IGW)

→ Allows resources in a VPC connect to internet

→ Scales horizantally, highly available and redundant

→ one VPC ⇄ (attached / attached) one Gateway

Bastian Hosts

→ can be used to SSH into private Ec2 instances

→ Bastion should be in public

→ Bastion host security group must allow inbound from internet on port 22 from restricted CIDR

NAT Gateway (Network Address Translation)

→ Aws managed NAT, higher bandwidth, high availability, no administration

→ Pay per hour for usage and bandwidth

→ NatGw is created in a specific AZ, uses an elastic IP

→ Can't be used by Ec2 instance in the same subnet

→ Requires an IGw (Private subnet → NATGw → IGw)

→ No security groups required

→ NAT is resilient within single AZ

→ multiple NAT's in multiple AZ's for fault tolerance

Security groups and NACLs

(Stateful) (Stateless)

Network Access control List (NACL)

→ like firewalls from and into subnets

→ one NACL per subnet

→ higher precedence with lower number

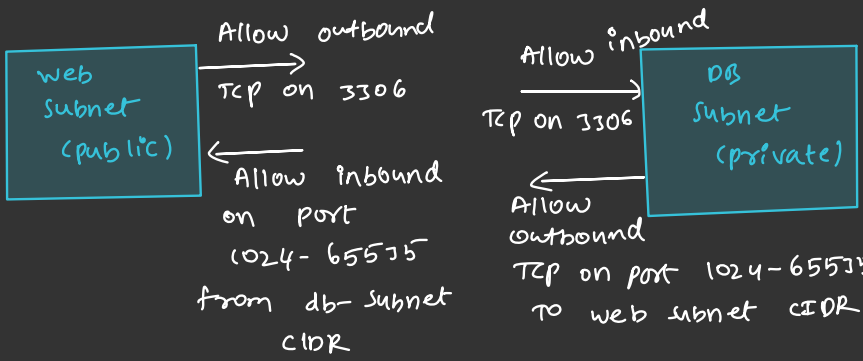
→ Nacl are a great way of blocking a specific Ip address at the subnet level

Ephemeral ports

→ For any two endpoints to establish a connection, they must use ports

→ clients connect to a defined port and expect a response on ephemeral port

NACL with Ephemeral ports



→ one nacl can have multiple subnets

Vpc peering

→ privately connect two vpc's using AWS network
→ must not have overlapping CIDR's
→ it is not transitive
→ must update route tables
→ can create between vpcs in diff AWS accounts/regions
→ can reference sg in peered connection

Vpc Endpoints (Aws private Link)

→ accessing objects in s3 bucket for an Ec2 instance in private subnet it has to be through internet and we will charged for that. is there any way to access objects of s3 on a private network. yes using vpc End points

Interface Endpoints (Amazon SNS)

→ Provisions an ENI (must attach SG)
→ Supports most AWS services
→ $ per hour + $ per GB of data processed

Gateway Endpoints

→ provisions a gateway and must be used as a target in a route table (no SG)
→ supports both S3 and DynamoDB
→ Free

Lamda in VPC accessing DynamoDB

→ Access from public internet
  → CUZ lambda is in a VPC, it needs a NAT gateway in a public subnet and an internet gateway
→ Access from the private VPC network (free)
  → Deploy a VPC gateway endpoint for DynamoDB
  → change the Route tables

VPC Flow logs

→ capture about IP traffic going into interfaces
  - VPC flow logs
  - Subnet flow logs

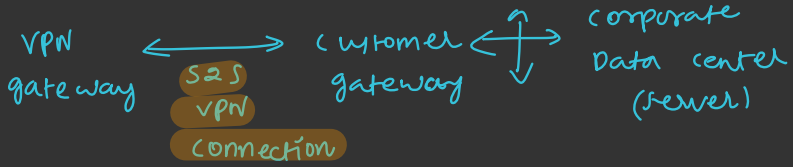
— ENI Flow logs

→ helps monitor and troubleshoot connectivity issues

→ Flow logs data can go to S3| cloud watch logs

→ ELB, RDS, Elastic cache, Redshift, workspaces, NATGW, Transit gateway

Site to site VPN



Direct connect (DX) — 30 days to connect

→ Fastest and costliest way of communication

→ need to setup a virtual private gateway on your VPC

→ Access S3 and EC2 on same connection.

→ use cases

- → increased bandwidth throughput
- → working with large datasets
- → lower cost
- → more consistent network experience
- → applications using real-time data feed
- → hybrid environments (prem + cloud)

Types

→ Dedicated connections

100Gbps capacity

physical ethernet port

→ Hosted connections

10Gbps capacity

can be added or removed on demand

→ can have backup connection incase direct connect fails (expensive)

Amazon cloudwatch metrics

→ It provides metrics for every service in AWS

→ metric is a variable to monitor

→ Dimension is an attribute of metric

→ metrics have timestamps

→ can create cloudwatch dashboard of custom metrics

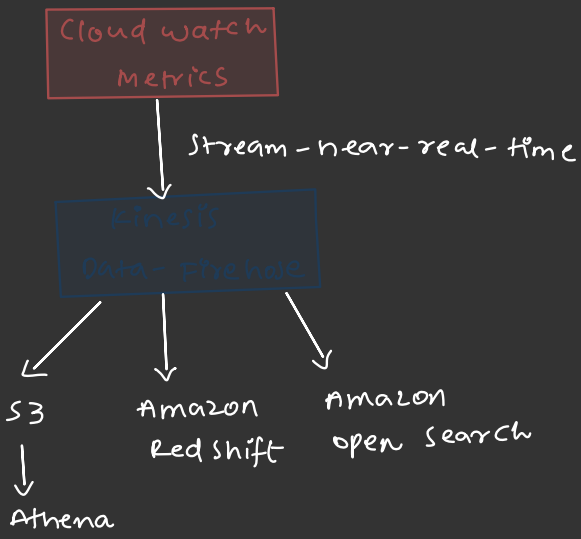
Cloudwatch metric streams

→ continually stream cloudwatch metrics to a destination of your choice, with near-real-time delivery and low latency

→ Amazon Kinesis Data Firehose

→ 3rd party service providers:

Datadog, Dynatrace, New Relic, Splunk, Sumo, Logic...



Cloudwatch Logs

→ Log groups: arbitary name, usually representing an application

→ Log stream: instances within application

→ Can define log expiration policies

→ These logs can be sent to

- → Amazon S3
- → Kinesis Data streams
- → kinesis Data Firehose
- → AWS Lambda
- → Elastic search

Cloud watch Logs - Sources

→ SDK, cloudwatch Logs Agent, Cloudwatch unified Agent

→ Elastic Beanstalk: collection of logs from application

→ Ecs: collection from containers

→ Aws Lambda: collection from function logs

→ vpc flow logs: vpc specific logs

→ Api gateway

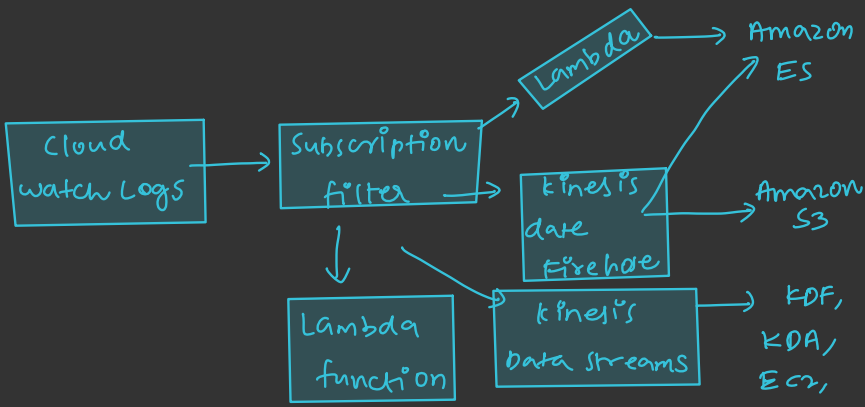
→ Cloudtrail based on filter

→ Route 53: Log DNS queries

Cloudwatch Logs – S3 Export

→ Log data can take up to 12 hrs to become available for export

→ the Api call is createExportTask



→ Kinetic Data Firehose can only delivery to S3, redshift, opensearch (Amazon ES)

Cloudwatch Logs for EC2

→ we need to run a cloudwatch agent on EC2 to push the log files you want (IAM permissions)

Cloudwatch Logs Agent

→ old version of agent
→ can only send to cloudwatch Logs

Cloudwatch unified Agent

→ collect additional system-level metrics such as RAM, processes etc..
→ collect logs to send cloudwatch Logs
→ centralized configuration using SSM Parameter Store
→ CPU, disk metrics, Disk IO, RAM, netstat, processes, swap space

Cloud watch Alarms

→ To trigger notifications for any metric

States: OK
INSUFFICIENT_DATA
ALARM

Cloudwatch Alarm Targets

→ stop, terminate, reboot or recover an EC2 instance
→ Trigger Auto scaling Action

→ Send notification to SNS

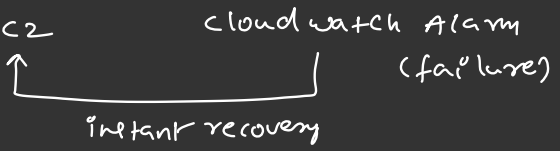
Composite Alarms

→ For monitoring the states of multiple other Alarms

→ We use AND and OR conditions

→ Helpful to reduce alarm noise

EC2 Instance Recovery



→ To test alarms and notifications, set the alarm state to Alarm using CLI

Cloudwatch container Insights

→ Collect aggregate, summarize metrics and logs from containers

→ Amazon ECS, Amazon EKS, Kubernets Platform on EC2, Fargate

→ In amazon EKS and kubernetes, Cloudwatch insights is using a containerized version of the cloudwatch Agent to discover containers

Cloudwatch Lambda insights

→ monitoring and troubleshooting solution

for serverless applications running on AWS Lambda

→ Lambda insights are provided as a Lambda layer

Cloudwatch contributor Insights

→ see metrics about the top-N contributors

→ helps you find top talkers and understand who or what is impacting system performance

→ cloudwatch Application insights is powered by sagemaker

→ Finding and alerts are sent to Amazon Event bridge and SSM ops center.

Cloud Trail

→ provides governance, compliance and audit for your AWS account

→ get an history of events / API calls made within AWS account

→ if a resource is deleted in AWS, investigate cloud trail first

→ can send to cloud watch Logs or S3 bucket or (Event bridge event)

→ can enable cloud trail insights to detect

→ inaccurate resource provisioning

→ hitting service limits
→ bursts of AWS IAM actions
→ gaps in periodic maintanence activity

→ events are stored for 90 days in cloud trail
→ to keep events beyound this period, log them to S3 and use Athena
→ cloud trail records
→ managment Events
→ Data Events
→ Insights Events

Aws config

→ Helps with auditing and compliance of Aws account
→ can recieve SNS notification
→ per region service
→ data in S3 analyzed by S3
→ custom config Rules must be defined in Aws Lambda
→ Aws config Rules does not prevent actions from happening
→ No free tier, a bit expensive.

CloudWatch vs CloudTrail vs Config

- CloudWatch
  - Performance monitoring (metrics, CPU, network, etc…) & dashboards
  - Events & Alerting
  - Log Aggregation & Analysis
- CloudTrail
  - Record API calls made within your Account by everyone
  - Can define trails for specific resources
  - Global Service
- Config
  - Record configuration changes
  - Evaluate resources against compliance rules
  - Get timeline of changes and compliance

Cloud Front

→ managed Aws Service that provides solution for content Delivery network (CDN)

→ A CDN is a geographically distributed network of proxy servers and their data centers

- → placing datacenters closer
- → high/ global availability
- → content caching at locations
- → static content, High Read performance
- → Reducing bandwidth costs
- → Improving page load times

→ lambda @ edge for run custom mode

→ Integrates with Aws web Application Firewall

→ Pay-as you go method

→ Can be used to secure and accelerate web socket traffic as well as API calls

→ Allows you to communicate with external HTTPS and talk to internal HTTP backends

→ Regional Edge cache are global cloud Front servers that is a mediator between host and the edge location

→ Edge locations are placements of proxy server

Cloud Front - origin

→ origin is a location where content is stored, and from which cloudFront gets conent to serve the viewer

S3 origin config

→ use this type to specify an Amazon S3 bucket that is not configured with static web site hosting

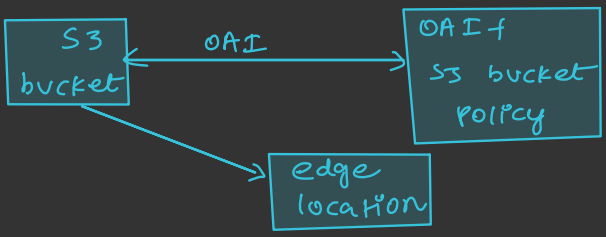
custom origin config

→ S3 for static website hosting
→ elastic load balancing load balancer
→ AWS elemental media package
→ AWS elemental media story container
→ HTTP server, running on an Amazon

EC2 or any kind of other host

Origin Access Identity (OAI)

→ used for sharing private content via Cloud Front



→ Adding the public IP of edge location to the security group of EC2 instance

→ Adding the public IP of edge location to the security group of ALB and then Add the load balancer to the security group of EC2 instance

Restriction

→ use Cloud Front geo restriction

whitelist: Allow your users to access your content only if they're in one of the countries on whitelist of approved countries (Allow list)

(Block list) Black list: prevent users to acess - - -

Process:

→ add a whitelist that contains only the

name of location

→ when you send a request, DNS routes the request to the CloudFront edge location

→ edge location determines that the user is not allowed

→ CloudFront returns an HTTP 403 (Forbidden) to the user

→ use a third party geo location service

→ with this you can restrict based on city, zip or postal code, or even longitude or latitude

→ it is recommended to use CloudFront signed URL's when using 3rd party

→ low latency, high bandwidth, redundant, scalable, global, cost-effective

→ Log analyzer for elastic map reduce

→ generate usage reports containing total traffic volume, object popularity, a break down of traffic by client IPs and edge location

→ Standard W3C format

→ Identify performance bottlenecks caused by slow loading content

→ Tiered pricing, rates go down as volume increases

→ Reserved cloud Front capacity pricing reduces with a longer term commitment

→ RTmp distributions for streaming data

→ DDOS protection

→ price class All: best perfomance, expensive

→ price class 200: most regions excluding expensive ones

→ price class 100: only the least expensive regions

Cloud Front - catch Invalidations

→ Forcing an entire partial cache refresh to reset the cache

unicast Ip vs Anycast Ip

→ unicast: every server has diff Ip address

→ Anycast Ip: every Server has same Ip. and the nearest one is chosen

Aws Global Accelartor

→ 2 Anycast Ip addresses are created for your application

→ The Anycast Ip send traffic directly to

edge locations

→ The edge locations send the traffic to your application

→ works with elastic IP, EC2 instance, ALB, NLB, public or private

→ consistent performance (low latency, NO cache) client

→ health checks (disaster recovery)

→ security (DDOS protection, only 2 IP's)

Cloud Front vs global accelarator

→ Improves perfomance for both cacheable content and dynamic content

→ content is served at the edge

Global

→ perfomance for a wide range over TCP or UDP

→ proxying packets running in one or more AWS regions

→ good fit for non-HTTP use cases, such as gaming (UDP), IOT (MQTT) or voice over IP

→ good for HTTP use cases that require static IP address

→ good for HTTP use cases that require deterministic, fast regional failover

Aws Databases - RDS

→ Divided into SQL and no-SQL

→ we can have databases running on EC2 instance

→ webserver application in one AZ, and db on another AZ (extra cost for communication between different AZ)

→ we run EC2 because

- → Some db's require OS level access
- → Advanced DB option tuning (DB Root)
- → vendor demands
- → DB version that Aws doesn't provide

→ we shouldn't run on EC2 because

- → Admin overhead
- → Backup and dPsaster recovery
- → EC2 running on a single AZ
- → will miss out features from AWS DB
- → skills and setup time to monitour
- → perfomance will be slower

Relational database Service (RDS)

→ Database server - as - a - service

- → my SQL
- → maria DB
- → postgreSQL
- → oracle
- → microsoft SQL
- → Amazon Aurora

RDS database Instance

→ DB connects with a CNAME

→ db.m6 general, db.r5 memory, db.t3 burst

→ EBS is located in the same AZ as RDS

→ RDS is vulnerable to failures in that AZ

→ Billing per instance and hourly rate for that compute. you will be billed for storage allocated

→ we can connect to the database using the database endpoint. And it doesn't change and unique for every RDS instance

Migrating DB from EC2 to RDS

→ get the dump of your existing DB on EC2

→ connect to your RDS DB instance

→ migrate the DB Dump that you created to RDS

→ verify if the data is available

RDS Multi-AZ (High-availability)

→ RDS Access only via db CNAME. The CNAME will point at the primary instance. you cannot access the standby replica for any reason via RDS

→ The standby replica cannot be used for extra capacity

→ synchronous Replication

→ The standby replica cannot be accessed directly unless a failure occurs

→ Failover is highly availabe but not fault tolerant

RDS Backup and restores

Rpo - Recovery point objective

→ Time between the last backup and when the failure occured

→ Amount of max data loss

→ influences technical solution and cost

→ Business usally provides an Rpo value

RTO — Recovery time objective

→ time between the DR event and full recovery

→ influenced by process, staff, tech and documentation.

RDS Backups

→ First snap is FULL size of consumed data and next snap will be taking of what's new.

→ Snapshot need to manually deleted

→ Automatic snapshots

→ Every 5 min translation logs are saved to S3. A database can then be restored to a 5 min snapshots in time

→ when you delete the db, they can be retained but they will expire based on their retention period.

RDS - Read Replicas

→ Asynchronous replication.

→ first stored in primary instance. once it is stored - on disk, it is pushed to the replica. so a small lag

→ 5 Read replicas per db instance

→ can provide global performance improvements

→ Cross - region replication doesn't change RTO.

Athena

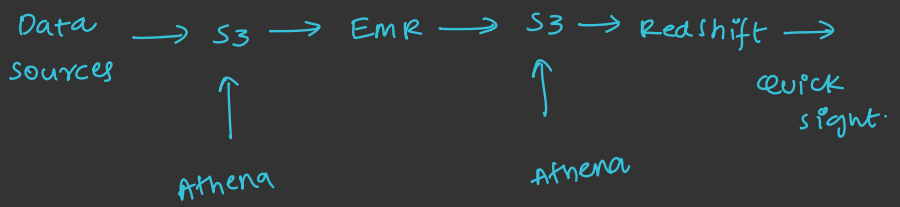
→ It is an interactive query service that makes it easy to analyze data directly from Amazon S3 using standard SQL

→ Serverless, No infrastructure, No spin - up time, Transparent upgrades.

→ Highly available

Query data from Amazon S3

→ No loading of data
→ Query data in it's raw format
→ convert to an optimized form like ORC or Parquet for the best performance and lowest cost
→ No ETL required.
→ ANSI / SQL
→ Presto is used for SQL Queries, in-memory distributed query engien ANSI-SQL compatible with extensions
→ HIVE is used for DDL functionality
→ Athena supports multiple data formats
→ To improve Query performance
  → compress your data
  → use columnar formats
→ pay per query, $5 per TB scanned from S3
→ high degree of flexibility
→ Hive allows other concepts such " external tables" and partioting data
→ meta data will be stored in meta data store.



Metadata

→ Highly available and durable

→ Access via DDL statments

Converting ORC to PARQUET

→ can use hive CATS

→ can use spark

→ 20 lines of Pyspark code, running on EMR

→ converts 1TB data into 130GB ($5)

→ ways to save costs

→ compress

→ convert to columnar format

→ use partitioning

Use cases

quicksight ⟷ Athena ⟷ Amazon S3

Macie

→ A fully managed data security and data privacy secure that uses machine learni-

ng and pattern matching to discover and protect sensitive data

→ lowers the cost of protecting data

→ works on S3 to identify and alert if any sensitive data, such PII is found

→ can be sent to Amazon cloud watch events

→ it can help you meet HIPAA and GDPR

→ allows to define our own custom sensitive data types

→ multi-account support

→ charges on the basis of no. of Amazon S3 buckets that are evaluated and the quantity of data processed

→ configure a repository if you want to store the info of macie for more than 90 days (repository - S3 bucket)

AWS Kinesis

→ The cloud Formation template will launch

→ A VPC

→ A cloud9 instance to run kinesis client

→ two S3 buckets store your source taxi trip dataset

→ A kinesis data analytics studio application and an associated glue database

→ A lambda function to process data from kinesis data streams

Kinesis

→ Real time streaming solution which allows you to ingest buffer and process data.

→ fully managed

→ scalable and can handle any amount of data with low-latencies.

→ kinesis stream consists of shrads, shards consists of records and records consist of data

→ Data records are composed of
  → sequence number
  → partition key
  → data blob (will not change - 1 MB)

→ A partion key is used by group data by shrad within a stream. It uses the partition key that is associated with each data record to determine which shrad a given data record belongs to.

→ each data record has a sequence no. that is unique per partition-key within its shrad.

→ consumers get records from Amazon kinesis Data streams and process them. these consumers are known as Amazon kinesis Data streams Application.

SDK vs KPL

→ Put record operation allows a single data record within an API call.

→ write multiple records to multiple shrads per request

→ only available in Java

→ will increase latency

Lambda consumer

→ It will consume data from kinesis data stream

→ it inspects the incoming message for unclean records with missing fields and filters them out

→ it sends clean records to dynamo DB.

→ A throttling error (all failed or only some)

kinesis client library

→ connects to the data stream

→ instantiates a record processor for every

shrad it manages.

→ pulls data records from the data stream

→ Balances shrad-worker associations

→ Check points processed records

→ handles instance failure, automatic load balanced

kinesis studio notebook

→ Allows you to interactively query data streams in real time, and easily build and run stream processing applications using standard SQL, python and scala.

→ Data visualization

→ exporting data to files

→ converting output format for easier analysis

kinesis Firehose

→ easiest way to load streaming data into data stores and analytic tools

→ near real-time analysis (by loading data into various other AWS services)

→ Fully managed service and requires no ongoing administration

→ can also batch, compress and encrypt data before loading, minimizing data storage

→ video streams (live video, stored and encrypts at rest, specify custom retention period)

→ Data stream (data availability in milliseconds store data for a min of 24 hours)

→ Kinesis Analytics (process data by SQL)

Data Tranfer

Data sync

→ Data tranfer service for moving large amounts of data into AWS

→ Typically stored on-premises but can also be moved between AWS storage services

→ Built in security capabilities (encryption in transit)

→ AWS Data Sync Agent on a server that connects you to a file system - it copies data and writes to AWS

→ Can connect with S3, EFS or FSX

Snowball

→ Physical device used for petabyte-scale data transport that can securely transfer large amounts of data

→ Addresses lot of common challenges

large scale data-transfer

including high network costs

long transfer times

security concerns

→ very secure, has multiple security layers, is tamper resistance and once the transfer is complete everything is completely wiped from snowball device

→ Two sizes 50TB & 80 TB

Snowball Edge

→ 100TB data Transfer device

→ has both storage and compute capabilities

→ supports specific amazon EC2 instance type and AWS Lambda functions

→ secure as data is encrypted and secure in Transit.

→ Temporary storage Tier in remote area locations

Snow mobile

→ 45 foot long container pulled by a truck for moving extremely large amounts of data to AWS

→ can transfer 100PB per Snowmobile

→ can be used for complete data center migration and can import to S3

→ Fast, secure and cost effective

ENI – Elastic Network Interface

→ An ENI is a logical component in a VPC that represent a virtual network card.

→ when we move a network interface from one instance to another, network traffic is redirected to the new instance

Why do we need network Interface?

→ creating a managment network

→ using security and network appliances in our VPC

→ creating dual homed instances with workloads / roles on distinct subnets

→ creating low-budget, high-availability solution

→ NO additional changes of EN, but your EC2 instances need to support it

→ we EN instead of ENI

→ used when you need a reliably high throughputs and speeds between 10 - 100 Gbps

→ using single root I/O virtualisation to provide high performance networking

→ provides higher bandwidth and packets for second