# IAM Section

* **Summary**
  + Users: mapped to a physical user, has a password for AWS Console
  + Groups: contains users only
  + Policies: json document that outlines permissions for users or groups
  + Roles: for EC@ instances or AWS services
  + Security: MFA + password policy
  + AWS CLI: manage your AWS services using the command-line
  + AWS-SDK: manage your AWS services using a programming language
  + Access Keys: access AWS using the CLI or SDK
  + Audit: IAM Credentials Reports & IAM Access Advisor
* **AWS CLI**
  + AWS — version → shows to us the version of CLI
  + AWS iam list-users → shows to us the lis of users
* **CoudShell**
  + ls → list files on environment
  + echo name > name.extension -> Create a new file on environment
  + We can download this files
  + cat → show what is in our file
  + pwd → full path to my file
    - Than I will copy this path to start my download
* **IAM Roles for Service**
  + Some AWS service will need to perform actions on your behalf
  + To do so, we will assign permissions to AWS Services with IAM Roles
  + Common Roles:
    - EC2 Instance Roles
    - Lambda Function Roles
    - Roles for CloudFormation
* **IAM Security Tools**

IAM Credentials Report (account-level)

* + a report that list all your account’s users and the status of their various credentials

IAM Access Advisor (user-level)

Access advisor shows the service permissions granted to a user and when those services were last accessed.

You can use this information to revise your policies.

* **IAM Guidelines & Best Practices**
  + Don’t use the root account except for AWS account setup
  + One physical user = One AWS user
  + Assign users to groups and assign permissions to groups
  + Create a strong password policy
  + Use and enforce the use of MFA
  + Create and use Roles for giving permissions to AWS services
  + Use Access Keys for Programmatic Access (CLI/SDK)
  + Audit permissions of your account with the IAM Credentials **Report**
  + **Never share IAM users & Access Keys**
* **Shared Responsibility Model for IAM**

AWS

* + Infrastructure (global network security)
  + Configuration and vulnerability analysis
  + Compliance validation

YOU

* + Users, Groups, Policies management and monitoring
  + Enable MFA on all accounts
  + Rotate all your keys often
  + Use IAM tools to apply appropriate permissions
  + Analyze access partners & review permissions

# EC2 (Elastic Compute Cloud)

* **Section - Summary**
  + EC2 Instance: AMI (OS) + Instance Size (CPU + RAM) + Storage + security groups + EC2 User Data
  + Security Groups: Firewall attached to the EC2 instance
  + EC2 User Data: Script launched at the first of an instance
  + SSH: start a terminal into our EC2 Instances (port 22)
  + EC2 Instance Role: link to IAM roles
  + Purchasing Options: On-Demand, Spot, Reserved (Standard + Convertible + Scheduled), Dedicated Host, Dedicated Instance
* **Amazon EC2**
  + It mainly consists in the capability 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)
* **EC2 sizing & configuration options**
  + Operating System (OS): Linux, Windows or Mac OS
  + How much compute power & cores (CPU)
  + How much RAM
  + How much storage space:
    - Network-attached (EBS & EFS)
    - hardware (EC2 instance store)
    - By default, the root storage is terminated when you finish an instance.
    - You attach other volumes as you want in anytime. However, you need to create an storage in the same AZ - It’s possible to attach other storage volumes with diferentes AZ, but it’s out of scope from this course.
  + Network card: speed of the card, public IP address
  + Firewall rules: security group
  + Bootstrap script (configure at first launch): EC2 User Data
* **EC2 User Data**

It is possible to bootstrap our instances using an EC2 User data script.

* + bootstrapping means launching commands when a machine starts
  + That script is only run once at the instance first start
  + EC2 user data is used to automate boot tasks such as:
    - Installing updates
    - Installing software
    - Downloading common files from the internet
    - Anything you can think of
  + The EC2 User Data Script runs with the root user
* **EC2 Instance Types – Compute Optimized**

• Great for compute-intensive tasks that require high performance processors: • Batch processing workloads • Media transcoding • High performance web servers • High performance computing (HPC) • Scientific modeling & machine learning • Dedicated gaming servers

* **EC2 Instance Types – Memory Optimized**

• Fast performance for workloads that process large data sets in memory • Use cases: • High performance, relational/non-relational databases • Distributed web scale cache stores • In-memory databases optimized for BI (business intelligence) • Applications performing real-time processing of big unstructured data

* **EC2 Instance Types – Storage Optimized**

• Great for storage-intensive tasks that require high, sequential read and write access to large data sets on local storage • Use cases: • High frequency online transaction processing (OLTP) systems • Relational & NoSQL databases • Cache for in-memory databases (for example, Redis) • Data warehousing applications • Distributed file systems

* **Security Groups - Good to know**
  + Can be attached to multiple instances
  + Locked down to a region / VPC combination
  + Does live “outside” the EC2 – if traffic is blocked the EC2 instance won’t see it
  + It’s good to maintain one separate security group for SSH access
  + If your application is not accessible (time out), then it’s a security group issue
  + If your application gives a “connection refused“ error, then it’s an application error or it’s not launched
  + All inbound traffic is blocked by default
  + All outbound traffic is authorized by default
* **SSH for Windows**
  + We need de ‘.pem’ file to allow our connection
  + On PowerShell, go to our directory file and paste the command line
    - ssh -i .\course-key-pair.pem ec2-user@’the-instance’s-public-ip’
  + To logout, execute the command exit
* **EC2 Instances Purchasing Options**
  + On-Demand Instances – short workload, predictable pricing, pay by second
  + Reserved (1 & 3 years)
    - Reserved Instances – long workloads
    - 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
* **Shared Responsibility Model for EC2**

**AWS**

* + Infrastructure (global network security)
  + Isolation on physical hosts
  + Replacing faulty hardware
  + Compliance validation

**USER**

* + Security Groups rules
  + Operating-system patches and updates
  + Software and utilities installed on the EC2 instance
  + IAM Roles assigned to EC2 & IAM user access management
  + Data security on your instance

# EC2 Instance Storage Section

* **It’s a network drive (i.e. not a physical drive)**
  + It uses the network to communicate the instance, which means there might be a bit of latency
  + It can be detached from an EC2 instance and attached to another one quickly
* **It’s locked to an Availability Zone (AZ)**
  + An EBS Volume in us-east-1a cannot be attached to us-east-1b
  + To move a volume across, you first need to snapshot it
* **Have a provisioned capacity (size in GBs, and IOPS)**
  + You get billed for all the provisioned capacity
  + You can increase the capacity of the drive over time
* **Controls the EBS behavior when an EC2 instance terminates**
  + By default, the root EBS volume is deleted (attribute enabled)
  + By default, any other attached EBS volume is not deleted (attribute disabled)
  + This can be controlled by the AWS console / AWS CLI
  + Use case: preserve root volume when instance is terminated
* **EBS Snapshots**
  + Make a backup (snapshot) of your EBS volume at a point in time
  + Not necessary to detach volume to do snapshot, but recommended
  + Can copy snapshots across AZ or Region
* **EBS Snapshots Features**
  + **EBS Snapshot Archive**
    - Move a Snapshot to an ”archive tier” that is 75% cheaper
    - Takes within 24 to 72 hours for restoring the archive
  + **Recycle Bin for EBS Snapshots**
    - Setup rules to retain deleted snapshots so you can recover them after an accidental deletion
    - Specify retention (from 1 day to 1 year)

# AMI

* **Overview**
  + AMI = Amazon Machine Image
  + AMI are a **customization** of an EC2 instance
    - You 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**: AWS provided
    - **Your own AMI**: you make and maintain them yourself
    - **An AWS Marketplace AMI**: an AMI someone else made (and potentially sells)
* **AMI Process (from an EC2 instance)**
  + Start an EC2 instance and customize it
  + Stop the instance (for data integrity)
  + Build an AMI – this will also 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
  + **If you need a high-performance hardware disk, use EC2 Instance Store**
  + Better I/O performance
  + EC2 Instance Store lose their storage if they’re stopped (ephemeral)
  + 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 planning

* **EFS Infrequent Access (EFS-IA)**

**.** Storage class that is cost-optimized for files not accessed every day • Up to 92% lower cost compared to EFS Standard • EFS will automatically move your files to EFS-IA based on the last time they were accessed • Enable EFS-IA with a Lifecycle Policy • Example: move files that are not accessed for 60 days to EFS-IA • Transparent to the applications accessing EFS

* **Shared Responsibility Model for EC2 Storage**

**AWS**

**.** Infrastructure • Replication for data for EBS volumes & EFS drives • Replacing faulty hardware • Ensuring their employees cannot access your data

**ME**

**.** Setting up backup / snapshot procedures • Setting up data encryption • Responsibility of any data on the drives • Understanding the risk of using EC2 Instance Store

* **Amazon FSx – Overview**

**.** Launch 3rd party high-performance file systems on AWS • Fully managed service

* **Amazon FSx for Windows File Server**

**.** A fully managed, highly reliable, and scalable Windows native shared file system • Built on Windows File Server • Supports SMB protocol & Windows NTFS • Integrated with Microsoft Active Directory • Can be accessed from AWS or your on-premise infrastructure

* **Amazon FSx for Lustre**

**.** A fully managed, high-performance, scalable file storage for High Performance Computing (HPC) • The name Lustre is derived from “Linux” and “cluster” • Machine Learning, Analytics, Video Processing, Financial Modeling, … • Scales up to 100s GB/s, millions of IOPS, sub-ms latencies

* **EC2 Instance Storage - Summary**
  + **EBS volumes:**

• network drives attached to one EC2 instance at a time

• Mapped to an Availability Zones

• Can use EBS Snapshots for backups / transferring EBS volumes across AZ

* + **AMI**: create ready-to-use EC2 instances with our customizations
  + **EC2 Image Builder:** automatically build, test and distribute AMIs
  + **EC2 Instance Store**:
    - High performance hardware disk attached to our EC2 instance
    - Lost if our instance is stopped / terminated
  + **EFS**: network file system, can be attached to 100s of instances in a region
  + **EFS-IA**: cost-optimized storage class for infrequent accessed files
  + **FSx for Windows:** Network File System for Windows servers
  + **FSx for Lustre**: High Performance Computing Linux file system

# Elastic Load Balancing & Auto Scaling Groups Section

* **Scalability & High Availability**
  + Scalability means that an application / system can handle greater loads by adapting.
  + There are two kinds of scalability:
    - Vertical Scalability
    - Horizontal Scalability (= elasticity)
  + **Scalability is linked but different to High Availability**
* **Vertical Scalability**
  + Vertical Scalability means increasing the size of the instance
  + For example, your application runs on a t2.micro
  + Scaling that application vertically means running it on a t2.large
  + Vertical scalability is very common for non distributed systems, such as a database.
  + There’s usually a limit to how much you can vertically scale (hardware limit)
* **Horizontal Scalability**
  + Horizontal Scalability means increasing the number of instances / systems for your application
  + Horizontal scaling implies distributed systems.
  + This is very common for web applications / modern applications
  + It’s easy to horizontally scale thanks the cloud offerings such as Amazon EC2
* **High Availability**
  + High Availability usually goes hand in hand with horizontal calling
  + High availability means running your application / system in at least 2 Availability Zones
  + The goal of high availability is to survive a data center loss (disaster)
* **High Availability & Scalability For EC2**
  + Vertical Scaling: Increase instance size (= scale up / down) • From: t2.nano - 0.5G of RAM, 1 vCPU • To: u-12tb1.metal – 12.3 TB of RAM, 448 vCPUs
  + Horizontal Scaling: Increase number of instances (= scale out / in) • Auto Scaling Group • Load Balancer
  + High Availability: Run instances for the same application across multi AZ • Auto Scaling Group multi AZ • Load Balancer multi AZ
* **Scalability vs Elasticity (vs Agility)**
  + Scalability: ability to accommodate a larger load by making the hardware stronger (scale up), or by adding nodes (scale out)
  + Elaticity: once a system is scalable, elasticity means that there will be some “auto-scaling” so that the system can scale based on the load. This is “cloud-friendly”: pay-per-use, match demand, optimize costs
  + Agility: (not related to scalability - distractor) new IT resources are only a click away, which means that you reduce the time to make those resources available to your developers from weeks to just minutes.
* **What is load balancing?**
  + Load balancers are servers that forward internet traffic to multiple servers (EC2 Instances) downstream
* **Why use a load balancer?**
  + Spread load across multiple downstream instances
  + Expose a single point of access (DNS) to your application
  + Seamlessly handle failures of downstream instances
  + Do regular health checks to your instances
  + Provide SSL termination (HTTPS) for your websites
  + High availability across zone
* **Why use an Elastic Load Balancer?**

An ELB (Elastic Load Balancer) is a managed load balancer • AWS guarantees that it will be working • AWS takes care of upgrades, maintenance, high availability • AWS 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) • 4 kinds of load balancers offered by AWS: • Application Load Balancer (HTTP / HTTPS only) – Layer 7 • Network Load Balancer (ultra-high performance, allows for TCP) – Layer 4 • Gateway Load Balancer – Layer 3 • Classic Load Balancer (retired in 2023) – Layer 4 & 7

* **What’s an Auto Scaling Group?**
  + In real-life, the load on your websites and application can change
  + In the cloud, you can create and get rid of servers very quickly
  + The goal of an Auto Scaling Group (ASG) is to:
    - Scale out (add EC2 instances) to match an increased load
    - Scale in (remove EC2 instances) to match a decreased load
    - Ensure we have a minimum and a maximum number of machines running
    - Automatically register new instances to a load balancer
    - Replace unhealthy instances
  + Cost Savings: only run at an optimal capacity (principle of the cloud)
* **Auto Scaling Groups – Scaling Strategies**
  + Manual Scaling: Update the size of an ASG manually
  + Dynamic Scaling: Respond to changing demand
    - Simple / Step Scaling
      * When a CloudWatch alarm is triggered (example CPU > 70%), then add 2 units
      * When a CloudWatch alarm is triggered (example CPU < 30%), then remove 1
    - Target Tracking Scaling
      * Example: I want the average ASG CPU to stay at around 40%
    - Scheduled Scaling
      * Anticipate a scaling based on known usage patterns
      * Example: increase the min. capacity to 10 at 5 pm on Fridays
  + Predictive Scaling
    - Uses Machine Learning to predict future traffic ahead of time
    - Automatically provisions the right number of EC2 instances in advance
    - Useful when your load has predictable time based pattern
* **ELB & ASG – Summary**
  + **High Availability** vs **Scalability** (vertical and horizontal) vs **Elasticity** vs **Agility** in the Cloud
  + **Elastic Load Balancers (ELB)**
    - Distribute traffic across backend EC2 instances, can be Multi-AZ
    - Supports health checks
    - 3 types: Application LB (HTTP – L7), Network LB (TCP – L4), Classic LB (old)
  + Auto Scaling Groups (ASG)
    - Implement Elasticity for your application, across multiple AZ
    - Scale EC2 instances based on the demand on your system, replace unhealthy
    - Integrated with the ELB

# ****Amazon S3 Section****

* **Amazon S3 – Summary**
  + **Buckets vs Objects**: global unique name, tied to a region
  + **S3 security**: IAM policy, S3 Bucket Policy (public access), S3 Encryption
  + **S3 Websites**: host a static website on Amazon S3
  + **S3 Versioning**: multiple versions for files, prevent accidental deletes
  + **S3 Replication**: same-region or cross-region, must enable versioning
  + **S3 Storage Classes**: Standard, IA, 1Z-IA, Intelligent, Glacier (Instant, Flexible, Deep)
  + **Snow Family**: import data onto S3 through a physical device, edge computing
  + **OpsHub**: desktop application to manage Snow Family devices
  + **Storage Gateway**: hybrid solution to extend on-premises storage to S3
* **Section introduction**
  + Amazon S3 is one of the main building blocks of AWS
  + It’s advertised as ”infinitely scaling” storage
  + Many websites use Amazon S3 as a backbone
  + Many AWS services use Amazon S3 as an integration as well
  + We’ll have a step-by-step approach to S3
* **Amazon S3 Use cases**
  + Backup and storage
  + Disaster Recovery
  + Archive
  + Hybrid Cloud storage
  + Application hosting
  + Media hosting
  + Data lakes & big data analytics
  + Software delivery
  + Static website
* **Amazon S3 - Buckets**
  + Amazon S3 allows people to store objects (files) in “buckets” (directories)
  + Buckets must have a globally unique name (across all regions all accounts)
  + Buckets are defined at the region level
  + **S3 looks like a global service but buckets are created in a region**
  + Naming convention
    - No uppercase, No underscore
    - 3-63 characters long
    - Not an IP
    - Must start with lowercase letter or number
    - Must NOT start with the prefix xn--
    - Must NOT end with the suffix -s3alias
* **Amazon S3 - Objects**
  + Objects (files) have a Key
  + The key is the FULL path:
    - s3://my-bucket/my\_file.txt
    - s3://my-bucket/my\_folder1/another\_folder/my\_file.txt
  + The key is composed of prefix + object name
    - s3://my-bucket/my\_folder1/another\_folder/my\_file.txt
  + There’s 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:
    - Max. Object Size is 5TB (5000GB)
    - If uploading more than 5GB, must use “multi-part upload”
  + Metadata (list of text 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)
* **Amazon S3 – Security**
  + User-Based
    - IAM Policies – which API calls should be allowed for a specific user from IAM
  + Resource-Based
    - Bucket Policies – bucket wide rules from the S3 console - allows cross account
    - Object Access Control List (ACL) – finer grain (can be disabled)
    - Bucket Access Control List (ACL) – less common (can be disabled)
  + Note: an IAM principal can access an S3 object if
    - The user IAM permissions ALLOW it OR the resource policy ALLOWS it
    - AND there’s no explicit DENY
  + Encryption: encrypt objects in Amazon S3 using encryption keys
* **S3 Bucket Policies**
  + JSON based policies
    - Resources: buckets and objects
    - Effect: Allow / Deny
    - Actions: Set of API to Allow or Deny
    - Principal: The account or user to apply the policy to
  + 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)
* **Bucket settings for Block Public Access**
  + These settings were created to prevent company data leaks
  + If you know your bucket should never be public, leave these on
  + Can be set at the account level
* **Amazon S3 – Static Website Hosting**
  + S3 can host static websites and have them accessible on the Internet
  + The website URL will be (depending on the region)
    - <http://bucket-name.s3-website-aws-region.amazonaws.com> OR
    - <http://bucket-name.s3-website.aws-region.amazonaws.com>
  + If you get a 403 Forbidden error, make sure the bucket policy allows public reads!
* **Amazon S3 -Versioning**
  + You can version your files in Amazon S3
  + It is enabled at the bucket level
  + Same key overwrite will change 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
  + Notes:
    - Any file that is not versioned prior to enabling versioning will have version “null”
    - Suspending versioning does not delete the previous versions
* **Amazon S3 – Replication (CRR & SRR)**
  + Must enable Versioning in source and destination buckets
  + Cross-Region Replication (CRR)
  + Same-Region Replication (SRR)
  + Buckets can be in different AWS accounts
  + Copying is asynchronous
  + Must give proper IAM permissions to S3
  + Use cases:
    - CRR – compliance, lower latency access, replication across accounts
    - SRR – log aggregation, live replication between production and test accounts
* **S3 Storage Classes**

**.** Amazon S3 Standard - General Purpose • Amazon S3 Standard-Infrequent Access (IA) • Amazon S3 One Zone-Infrequent Access • Amazon S3 Glacier Instant Retrieval • Amazon S3 Glacier Flexible Retrieval • Amazon S3 Glacier Deep Archive • Amazon S3 Intelligent Tiering • Can move between classes manually or using S3 Lifecycle configurations

* **S3 Durability and Availability**
  + Durability:
    - High durability (99.999999999%, 11 9’s) of objects across multiple AZ
    - If you store 10,000,000 objects with Amazon S3, you can on average expect to incur a loss of a single object once every 10,000 years
    - Same for all storage classes
  + Availability:
    - Measures how readily available a service is
    - Varies depending on storage class
    - Example: S3 standard has 99.99% availability = not available 53 minutes a year
* **S3 Standard – General Purpose**
  + 99.99% Availability
  + Used for frequently accessed data
  + Low latency and high throughput
  + Sustain 2 concurrent facility failures
  + Use Cases: Big Data analytics, mobile & gaming applications, content distribution…
* **S3 Storage Classes – Infrequent Access**
  + For data that is less frequently accessed, but requires rapid access when needed
  + Lower cost than S3 Standard
  + Amazon S3 Standard-Infrequent Access (S3 Standard-IA) • 99.9% Availability • Use cases: Disaster Recovery, backups
  + Amazon S3 One Zone-Infrequent Access (S3 One Zone-IA) • High durability (99.999999999%) in a single AZ; data lost when AZ is destroyed • 99.5% Availability • Use Cases: Storing secondary backup copies of on-premise data, or data you can recreate
* **Amazon S3 Glacier Storage Classes**
  + Low-cost object storage meant for archiving / backup
  + Pricing: price for storage + object retrieval cost
  + Amazon S3 Glacier Instant Retrieval
    - Millisecond retrieval, great for data accessed once a quarter
    - Minimum storage duration of 90 days
  + Amazon S3 Glacier Flexible Retrieval (formerly Amazon S3 Glacier):
    - Expedited (1 to 5 minutes), Standard (3 to 5 hours), Bulk (5 to 12 hours) – free
    - Minimum storage duration of 90 days
  + Amazon S3 Glacier Deep Archive – for long term storage:
    - Standard (12 hours), Bulk (48 hours)
    - Minimum storage duration of 180 days
* **S3 Intelligent-Tiering**
  + Small monthly monitoring and auto-tiering fee
  + Moves objects automatically between Access Tiers based on usage
  + There are no retrieval charges in S3 Intelligent-Tiering
  + Frequent Access tier (automatic): default tier
  + Infrequent Access tier (automatic): objects not accessed for 30 days
  + Archive Instant Access tier (automatic): objects not accessed for 90 days
  + Archive Access tier (optional): configurable from 90 days to 700+ days
  + Deep Archive Access tier (optional): config. from 180 days to 700+ days
* **S3 Encryption**
  + No Encryption
  + Server-Side Encryption
  + Client-Side Encryptio
* **Shared Responsibility Model for S3**
  + AWS
    - Infrastructure (global security, durability, availability, sustain concurrent loss of data in two facilities)
    - Configuration and vulnerability analysis
    - Compliance validation
  + User
    - S3 Versioning
    - S3 Bucket Policies
    - S3 Replication Setup
    - Logging and Monitoring
    - S3 Storage Classes
    - Data encryption at rest and in transit
* **AWS Snow Family**
  + Highly-secure, portable devices to collect and process data at the edge, and migrate data into and out of AWS

**Snowball Edge (for data transfers)**

* + 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
  + Snowball Edge Storage Optimized
  + 80 TB of HDD capacity for block volume and S3 compatible object storage
  + Snowball Edge Compute Optimized • 42 TB of HDD capacity for block volume and S3 compatible object storage
  + Use cases: large data cloud migrations, DC decommission, disaster recovery

**AWS Snowcone**

* + Small, portable computing, anywhere, rugged & secure, withstands harsh environments
  + Light (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**

* + Transfer exabytes of data (1 EB = 1,000 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

**Snow Family – Usage Process**

* + Request Snowball devices from the AWS console for delivery
  + Install the snowball client / AWS OpsHub on your servers
  + Connect the snowball to your servers and copy files using the client
  + Ship back the device when you’re done (goes to the right AWS facility)
  + Data will be loaded into an S3 bucket
  + Snowball is completely wiped

**What is Edge Computing?**

* + What is Edge Computing?
    - A truck on the road, a ship on the sea, a mining station underground…
  + 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 transferring data for example)
* **AWS OpsHub**
  + Historically, to use Snow Family devices, you needed a CLI (Command Line Interface tool)
  + Today, you can use AWS OpsHub (a software you install on your computer / laptop) to manage your Snow Family Device
    - Unlocking and configuring single or clustered devices
    - Transferring files
    - Launching and managing instances running on Snow Family Devices
    - Monitor device metrics (storage capacity, active instances on your device)
    - Launch compatible AWS services on your devices
      * (ex: Amazon EC2 instances, AWS DataSync, Network File System (NFS))
* **Hybrid Cloud for Storage**

AWS is pushing for ”hybrid cloud” • Part of your infrastructure is on-premises • Part of your infrastructure is on the cloud • This can be due to • Long 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

* **AWS Storage Gateway**
  + Bridge between on-premise 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: • File Gateway • Volume Gateway • Tape Gateway
  + No need to know the types at the exam

# ****Databases Section****

* **Databases & Analytics Summary in AWS**
  + **Relational Databases - OLTP**: RDS & Aurora (SQL)
  + **Differences between Multi-AZ, Read Replicas, Multi-Region**
  + **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 Transactions Ledger (immutable journal, cryptographically verifiable)
  + **Amazon Managed Blockchain**: managed Hyperledger Fabric & Ethereum blockchains
  + **Glue**: Managed ETL (Extract Transform Load) and Data Catalog service
  + **Database Migration**: DMS
  + **Neptune**: graph database
* **Databases & Shared Responsibility on AWS**
  + AWS offers use to manage different databases
  + Benefits include:
    - Quick Provisioning, High Availability, Vertical and Horizontal Scaling
    - Automated Backup & Restore, Operations, Upgrades
    - Operating System Patching is handled by AWS
    - Monitoring, alerting
  + Note: many databases technologies could be run on EC2, but you must handle yourself the resiliency, backup, patching, high vailability, fault tolerance, scaling…
* **AWS RDS Overview**
  + RDS stands for Relational Database Service
  + It’s a managed DB service for DB use SQL as a query language.
  + It allows you to create databases in the cloud that are managed by AWS • Postgres • MySQL • MariaDB • Oracle • Microsoft SQL Server • Aurora (AWS Proprietary database)

**Advantage over using RDS versus deploying DB on EC2**

* + RDS is a managed service:
    - 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**
  + Aurora is a proprietary technology from AWS (not open sourced)
  + PostgreSQL and MySQL are both supported as Aurora DB
  + Aurora is “AWS cloud optimized” and claims 5x performance improvement over MySQL on RDS, over 3x the performance of Postgres on RDS
  + Aurora storage automatically grows in increments of 10GB, up to 64 TB.
  + Aurora costs more than RDS (20% more) – but is more efficient
  + Not in the free tier
* **RDS Deployments: Read Replicas, Multi-AZ**
  + Read Replicas: • Scale the read workload of your DB • Can create up to 5 Read Replicas • Data is only written to the main DB
  + Multi-AZ: • Failover in case of AZ outage (high availability) • Data is only read/written to the main database • Can only have 1 other AZ as failover
  + Multi-Region (Read Replicas) • Disaster recovery in case of region issue • Local performance for global reads • Replication cost
* **Amazon ElastiCache Overview**
  + The same way RDS is to get managed Relational Databases…
  + ElastiCache is to get managed Redis or Memcached
  + Caches are in-memory databases with high performance, low latency
  + Helps reduce load off databases for read intensive workloads
  + AWS takes care of OS maintenance / patching, optimizations, setup, configuration, monitoring, failure recovery and backups
* **DynamoDB**
  + Fully Managed Highly available with replication across 3 AZ
  + DynamoDB is a key/value database
  + **NoSQL database - not a relational database**
  + Scales to massive workloads, distributed “**serverless**” database
  + Millions of requests per seconds, 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
  + Standard & Infrequent Access (IA) Table Class
* **DynamoDB Accelerator - DAX**
  + Fully Managed in-memory cache for DynamoDB
  + 10x performance improvement – single- digit millisecond latency to microseconds latency – when accessing your DynamoDB tables
  + Secure, highly scalable & highly available
  + Difference with ElastiCache at the CCP level: DAX is only used for and is integrated with DynamoDB, while ElastiCache can be used for other databases
* **DynamoDB – Global Tables**
  + Make a DynamoDB table accessible with low latency in multiple-regions
  + Active-Active replication (read/write to any AWS Region)
* **Redshift Overview**
  + Redshift is based on PostgreSQL, but **it’s not used for OLTP**
  + **It’s OLAP – online analytical processing (analytics and data warehousing)**
  + Load data once every hour, not every second
  + 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 the instances provisioned
  + Has a SQL interface for performing the queries
  + BI tools such as AWS Quicksight or Tableau integrate with it
* **Amazon EMR**
  + EMR stands for “Elastic MapReduce”
  + EMR helps creating **Hadoop clusters** (Big Data) to analyze and process vast amount of data
  + The 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…**
* **Amazon Athena**
  + **Serverless** query service to analyze data stored in Amazon S3
  + Uses standard SQL language to query the files
  + Supports CSV, JSON, ORC, Avro, and Parquet (built on Presto)
  + Pricing: $5.00 per TB of data scanned
  + Use compressed or columnar data for cost-savings (less scan)
  + Use cases: Business intelligence / analytics / reporting, analyze & query VPC Flow Logs, ELB Logs, CloudTrail trails, etc...
  + **Exam Tip: analyze data in S3 using serverless SQL, use Athena**
* **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**
  + Aurora is an “AWS-implementation” of PostgreSQL / MySQL …
  + DocumentDB is the same for MongoDB (which is a NoSQL database)
  + MongoDB is used to store, query, and index JSON data
  + Similar “deployment concepts” as Aurora
  + Fully Managed, highly available with replication across 3 AZ
  + DocumentDB storage automatically grows in increments of 10GB, up to 64 TB.
  + 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
  + Users have friends
  + Posts have comments
  + Comments have likes from users
  + Users share and like posts…
  + Highly available across 3 AZ, with up to 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 and query the graph with milliseconds latency
  + Highly available with replications across multiple AZs
  + Great for knowledge graphs (Wikipedia), fraud detection, recommendation engines, social networking
* **Amazon QLDB**
  + QLDB stands for ”Quantum Ledger Database”
  + A ledger is a book recording financial transactions
  + Fully Managed, Serverless, High available, Replication across 3 AZ
  + Used to review history of all the changes made to your application data over time
  + Immutable system: no entry can be removed or modified, cryptographically verifiable
  + 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.
  + Amazon Managed Blockchain is a managed service to: • Join public blockchain networks • Or create your own scalable private network
  + Compatible with the frameworks Hyperledger Fabric & Ethereum
* **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
    - can be used by Athena, Redshift, EMR
* **DMS – Database Migration Service**
  + Quickly and securely migrate databases to AWS, resilient, self healing
  + The source database remains available during the migration
  + Supports:
    - Homogeneous migrations: ex Oracle to Oracle
    - Heterogeneous migrations: ex Microsoft SQL Server to Aurora

# Other Compute Section

* **What is Docker?**

Docker is a 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 behavior

• Less work

• Easier to maintain and deploy

• Works with any language, any 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 technologies or OS:

• Ubuntu

• MySQL

• NodeJS, Java…

• Private: Amazon ECR (Elastic Container Registry)

* **Docker versus Virtual Machines**
  + Docker is ”sort of” a virtualization technology, but not exactly
  + Resources are shared with the host => many containers on one server
* **ECS - Elastic Container Service**
  + Launch Docker containers on AWS
  + You must provision & maintain the infrastructure (the EC2 instances)
  + AWS takes care of starting / stopping containers
  + Has integrations with the Application Load Balancer
* **Fargate**
  + Launch Docker containers on AWS
  + You do not provision the infrastructure (no EC2 instances to manage) – simpler!
  + Serverless offering
  + AWS just runs containers for you based on the CPU / RAM you need
* **ECR - Elastic Container Registry**
  + Private Docker Registry on AWS
  + This is where you store your Docker images so they can be run by ECS or Fargate
* **What’s serverless?**
  + Serverless is a new paradigm in which the developers don’t have to manage servers anymore…
  + They just deploy code
  + They just deploy… 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 just don’t manage / provision / see them

So far in this course…

* + Amazon S3
  + DynamoDB
  + Fargate
  + Lambda
* **AWS 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
    - Free tier of 1,000,000 AWS Lambda requests and 400,000 GBs of 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 functions (up to 10GB of RAM!)
  + Increasing RAM will also improve CPU and network!
  + Lambda Container Image
    - The container image must implement the Lambda Runtime API
    - ECS / Fargate is preferred for running arbitrary Docker images
* **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 disk space

• Serverless

Batch:

• No time limit

• Any runtime as long as it’s packaged as a Docker image

• Rely on EBS / instance store for disk space

• Relies on EC2 (can be managed by AWS)

* **Amazon Lightsail**
  + Virtual servers, storage, databases, and networking
  + Low & predictable pricing
  + Simpler alternative to using EC2, RDS, ELB, EBS, Route 53…
  + Great for people with little cloud experience!
  + Can setup notifications and monitoring of your Lightsail resources
  + Use cases:
    - Simple web applications (has templates for LAMP, Nginx, MEAN, Node.js…)
    - Websites (templates for WordPress, Magento, 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
    - Serverless offering (no EC2 instances)
  + **ECR**: Private Docker Images Repository
  + **Batch**: run batch jobs on AWS across managed EC2 instances
  + **Lightsail**: predictable & low pricing for simple application & DB stacks
* **Lambda Summary**
  + Lambda is Serverless, Function as a Service, seamless scaling, reactive
  + **Lambda Billing**:
    - By the time run x by the RAM provisioned
    - By the number of invocations
  + **Language Support**: many programming languages except (arbitrary) Docker
  + **Invocation time**: up to 15 minutes
  + **Use cases**:
    - Create Thumbnails for images uploaded onto S3
    - Run a Serverless cron job
  + **API Gateway**: expose Lambda functions as HTTP API

# Deploying and Managing Infrastructure at Scale Section

* **What is CloudFormation**
  + CloudFormation is a declarative way of outlining your AWS Infrastructure, for any resources (most of them are supported).
  + For example, within a CloudFormation template, you say: • 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
  + Then CloudFormation creates those for you, in the right order, with the exact configuration that you specify
* **Benefits of AWS CloudFormation**
  + 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 using the CloudFormation template • Savings strategy: In Dev, you could automation deletion of templates at 5 PM and recreated at 8 AM, safely
  + Productivity • Ability to destroy and re-create an infrastructure on the cloud 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 web! • Leverage the documentation
  + Supports (almost) all AWS resources: • Everything we’ll see in this course is supported • You can use “custom resources” for resources that are not supported
* **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
* **Developer problems on AWS**
  + Managing infrastructure
  + Deploying Code
  + Configuring all the databases, load balancers, etc
  + Scaling concerns
  + Most web apps have the same architecture (ALB + ASG)
  + All the developers want is for their code to run!
  + Possibly, consistently across different applications and environments
* **AWS Elastic Beanstalk Overview**
  + Elastic Beanstalk is a developer centric view of deploying an application on AWS
  + It uses all the component’s we’ve seen before: EC2, ASG, ELB, RDS, etc…
  + 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 pay for the underlying instances
* **Elastic Beanstalk**
  + Managed service
    - Instance configuration / OS is handled by Beanstalk
    - Deployment strategy is configurable but performed by Elastic Beanstalk
    - Capacity provisioning
    - Load balancing & auto-scaling
    - Application health-monitoring & responsiveness
  + **Just the application code is the responsibility of the developer**
  + Three architecture models:
    - Single Instance deployment: good for dev
    - LB + ASG: great for production or pre-production web applications
    - ASG only: great for non-web apps in production (workers, etc..)
  + Support for many platforms
  + Single Container Docker
  + Multi-Container Docker
  + Preconfigured Docker
  + If not supported, you can write your custom platform (advanced)
* **Elastic Beanstalk – Health Monitoring**
  + Health agent pushes metrics to CloudWatch
  + Checks for app health, publishes health events
* **AWS CodeDeploy**
  + We want to deploy our application automatically
  + Works with EC2 Instances
  + Works with On-Premises Servers
  + Hybrid service
  + Servers / Instances must be provisioned and configured ahead of time with the CodeDeploy Agent
* **AWS CodeCommit**
  + Before pushing the application code to 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’ competing product is CodeCommit
  + CodeCommit:
    - 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 in the cloud (name is obvious)
  + 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)
  + Benefits: • Fully managed, compatible with CodeCommit, CodeBuild, CodeDeploy, Elastic Beanstalk, CloudFormation, GitHub, 3rd-party services (GitHub…) & custom plugins… • Fast delivery & rapid updates
* **AWS CodeArtifact**
  + Software packages depend on each other to be built (also called code dependencies), and new ones are created
  + 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 for software development
  + Works with common dependency management tools such as Maven, Gradle, npm, yarn, twine, pip, and NuGet
  + Developers and CodeBuild can then retrieve dependencies straight from CodeArtifac
* **AWS CodeStar**
  + Unified UI to easily manage software development activities in one place
  + “Quick way” to get started to correctly set-up CodeCommit, CodePipeline,CodeBuild, CodeDeploy, Elastic Beanstalk, EC2, etc…
  + Can edit the code ”in-the-cloud” using AWS Cloud9
* **AWS Cloud9**
  + AWS Cloud9 is a cloud IDE (Integrated Development Environment) for writing, running and debugging code
  + “Classic” IDE (like IntelliJ, Visual Studio Code…) are downloaded on a computer before being used
  + A cloud IDE can be used within a web browser, meaning you can work on your projects from your office, home, or anywhere with internet with no setup necessary
  + AWS Cloud9 also allows for code collaboration in real-time (pair programming)
* **AWS Systems Manager (SSM)**
  + Helps you manage your EC2 and On-Premises systems at scale
  + Another 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
* **How Systems Manager works**
  + We need to install the SSM agent onto the systems we control
  + Installed by default on Amazon Linux AMI & some Ubuntu AMI
  + If an instance can’t be controlled with SSM, it’s probably an issue with the SSM agent!
  + Thanks to the SSM agent, we can run commands, patch & configure our servers
* **Systems Manager – SSM Session Manager**
  + Allows you to start a secure shell on your EC2 and on-premises servers
  + No SSH access, bastion hosts, or SSH keys needed
  + No port 22 needed (better security)
  + Supports Linux, macOS, and Windows
  + Send session log data to S3 or CloudWatch Logs
* **AWS OpsWorks**
  + Chef & Puppet help you perform server configuration automatically, or repetitive actions
  + They work great with EC2 & On-Premises VM
  + AWS OpsWorks = Managed Chef & Puppet
  + It’s an alternative to AWS SSM
  + Only provision standard AWS resources: • EC2 Instances, Databases, Load Balancers, EBS volumes…
  + In the exam: Chef or Puppet needed => AWS OpsWorks
* **Deployment - Summary**
  + CloudFormation: (AWS only)
    - Infrastructure as Code, works with almost all of AWS resources
    - Repeat across Regions & Accounts
  + Beanstalk: (AWS only)
    - Platform as a Service (PaaS), limited to certain programming languages or Docker
    - Deploy code consistently with a known architecture: ex, ALB + EC2 + RDS
  + CodeDeploy (hybrid): deploy & upgrade any application onto servers
  + Systems Manager (hybrid): patch, configure and run commands at scale
  + OpsWorks (hybrid): managed Chef and Puppet in AWS
* **Developer Services - Summary**
  + CodeCommit: Store code in private git repository (version controlled)
  + CodeBuild: Build & test code in AWS
  + CodeDeploy: Deploy code onto servers
  + CodePipeline: Orchestration of pipeline (from code to build to deploy)
  + CodeArtifact: Store software packages / dependencies on AWS
  + CodeStar: Unified view for allowing developers to do CICD and code
  + Cloud9: Cloud IDE (Integrated Development Environment) with collab
  + AWS CDK: Define your cloud infrastructure using a programming language

# Global Infrastructure Section

* **Why make a global application?**
  + A global application is an application deployed in multiple geographies
  + On AWS: this could be Regions and / or Edge Locations
  + Decreased Latency
    - Latency is the time it 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, better experience
  + Disaster Recovery (DR)
    - If an AWS region goes down (earthquake, storms, power shutdown, politics)…
    - You can fail-over to another region and have your application still working
    - A DR plan is important to increase the availability of your application
  + Attack protection: distributed global infrastructure is harder to attack
* **Global AWS Infrastructure**
  + Regions: For deploying applications and infrastructure
  + Availability Zones: Made of multiple data centers
  + Edge Locations (Points of Presence): for content delivery as close as possible to users
* **Global Applications in AWS**
  + Global DNS: Route 53 • Great to route users to the closest deployment with least latency • Great for disaster recovery strategies
  + Global Content Delivery Network (CDN): CloudFront • Replicate part of your application to AWS Edge Locations – decrease latency • Cache common requests – improved user experience and decreased latency
  + S3 Transfer Acceleration • Accelerate global uploads & downloads into Amazon S3
  + AWS Global Accelerator: • Improve global application availability and performance using the AWS global network
* **Amazon Route 53 Overview**
  + Route53 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.
  + In AWS, the most common records are:
    - [www.google.com](http://www.google.com/) => 12.34.56.78 == A record (IPv4)
    - [www.google.com](http://www.google.com/) => 2001:0db8:85a3:0000:0000:8a2e:0370:7334 == AAAA IPv6
    - [search.google.com](http://search.google.com/) => [www.google.com](http://www.google.com/) == CNAME: hostname to hostname
    - [example.com](http://example.com/) => AWS resource == Alias (ex: ELB, CloudFront, S3, RDS, etc…)
* **Route 53 Routing Policies**
  + SIMPLE ROUTING POLICY
    - No health checks
  + WEIGHTED ROUTING POLICY
    - Distribute de charge
  + LATENCY ROUTING POLICY
    - Makes de latency more lower between the regions
  + FAILOVER ROUTING POLICY
    - Disaster Recovery
* **Amazon CloudFront**
  + Content Delivery Network (CDN)
  + Improves read performance, content is cached at the edge
  + Improves users 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 Control (OAC)
    - OAC is replacing 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 first enable the bucket as a static S3 website)
    - Any HTTP backend you want
* **CloudFront vs S3 Cross Region Replication**
  + 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 you want replication to happen • Files are updated in near real-time • Read only • Great for dynamic content that needs to be available at low-latency in few region
* **S3 Transfer Acceleration**
  + 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 to your application (60% improvement)
  + 2 Anycast IP are created for your application and traffic is sent through Edge Locations
  + The Edge locations send the traffic to your application
* **AWS Global Accelerator vs CloudFront**
  + They both use the AWS global network and its edge locations around the world
  + Both services integrate with AWS Shield for DDoS protection.
  + **CloudFront – Content Delivery Network**
    - Improves performance for your cacheable content (such as images and videos)
    - Content is served at the edge
  + **Global Accelerator**
    - No caching, proxying packets at the edge to applications running in one or more AWS Regions.
    - Improves performance for a wide range of applications over TCP or UDP
    - Good for HTTP use cases that require static IP addresses
    - Good for HTTP use cases that required deterministic, fast regional failover
* **AWS Outposts**
  + Hybrid Cloud: businesses that keep an on
  + premises infrastructure alongside a cloud infrastructure
  + Therefore, 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:
* **AWS WaveLength**
  + WaveLength Zones are infrastructure deployments embedded within the telecommunications providers’ datacenters at the edge of the 5G networks
  + Brings AWS services to the edge of the 5G networks
  + Example: EC2, EBS, VPC…
  + Ultra-low latency applications through 5G networks
  + Traffic doesn’t leave the Communication Service Provider’s (CSP) network
  + High-bandwidth and secure connection to the parent AWS Region
  + No additional charges or service agreements
  + Use cases: Smart Cities, ML-assisted diagnostics, Connected Vehicles, Interactive Live Video Streams, AR/VR, Real-time Gaming, …
* **AWS Local Zones**
  + Places AWS compute, storage, database, and other selected AWS services closer to end users to run latency-sensitive applications
  + Extend your VPC to more locations – “Extension of an AWS Region”
  + Compatible with EC2, RDS, ECS, EBS, ElastiCache, Direct Connect …
  + Example:
  + AWS Region: N. Virginia (us-east-1)
  + AWS Local Zones: Boston, Chicago, Dallas, Houston, Miami, …
* **Global Applications in AWS - Summary**
  + **Global DNS: Route 53** • Great to route users to the closest deployment with least latency • Great for disaster recovery strategies
  + **Global Content Delivery Network (CDN): CloudFront** • Replicate part of your application to AWS Edge Locations – decrease latency • Cache common requests – improved user experience and decreased latency
  + **S3 Transfer Acceleration** • Accelerate global uploads & downloads into Amazon S3
  + **AWS Global Accelerator:** • Improve global application availability and performance using the AWS global network
* **Global Applications in AWS - Summary**
  + **AWS Outposts**
    - Deploy Outposts Racks in your own Data Centers to extend AWS services
  + **AWS WaveLength**
    - Brings AWS services to the edge of the 5G networks
    - Ultra-low latency applications
  + **AWS Local Zones**
    - Bring AWS resources (compute, database, storage, …) closer to your users
    - Good for latency-sensitive applications

# Cloud Integration Section

* **Section Introduction**
  + When we start deploying multiple applications, they will inevitably need to communicate with one another
  + There are two patterns of application communication
    - 1. Synchronous communications (application to application)
      2. Asynchronous / Event based (application to queue to application)
  + Synchronous between applications can be problematic if there are sudden spikes of traffic
  + What if you need to suddenly encode 1000 videos but usually it’s 10?
  + In that case, it’s better to decouple your applications:
    - using SQS: queue model
    - using SNS: pub/sub model
    - using Kinesis: real-time data streaming model
  + These services can scale independently from our application!
* **Amazon SQS – Standard Queue**
  + Oldest AWS offering (over 10 years old)
  + Fully managed service (~serverless), use to decouple applications
  + Scales from 1 message per second to 10,000s per second
  + Default retention of messages: 4 days, maximum of 14 days
  + No limit to how many messages can be in the queue
  + Messages are deleted after they’re read by consumers
  + Low latency (<10 ms on publish and receive)
  + Consumers share the work to read messages & scale horizontally
  + SQS can be use to decouple between application tiers
* **Amazon Kinesis**

**For the exam: Kinesis = real-time big data streaming**

* + Managed service to collect, process, and analyze real-time streaming data at any scale
  + Too detailed for the Cloud Practitioner exam but good to know: • **Kinesis Data Streams**: low latency streaming to ingest data at scale from hundreds of thousands of sources • **Kinesis Data Firehose**: load streams into S3, Redshift, ElasticSearch, etc… • **Kinesis Data Analytics**: perform real-time analytics on streams using SQL • **Kinesis Video Streams**: monitor real-time video streams for analytics or ML
* **Amazon SNS**

What if you want to send one message to many receivers? SNS solves this issue.

* + 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 12,500,000 subscriptions per topic, 100,000 topics limit
* **Amazon MQ**
  + SQS, SNS are “cloud-native” services: proprietary protocols from AWS
  + Traditional applications running from on-premises may use open protocols such as: MQTT, AMQP, STOMP, Openwire, WSS
  + When migrating to the cloud, instead of re-engineering the application to use SQS and SNS, we can use Amazon MQ
  + Amazon MQ is a managed message broker service for RabbitMQ and ActiveMQ
  + Amazon MQ doesn’t “scale” as much as SQS / SNS
  + Amazon MQ runs on servers, can run in Multi-AZ with failover
  + Amazon MQ has both queue feature (~SQS) and topic features (~SNS)
* **Integration Section – Summary**
  + **SQS**: • Queue service in AWS • Multiple Producers, messages are kept up to 14 days • Multiple Consumers share the read and delete messages when done • Used to **decouple** applications in AWS
  + **SNS**: • Notification service in AWS • Subscribers: Email, Lambda, SQS, HTTP, Mobile… • Multiple Subscribers, send all messages to all of them • No message retention
  + **Kinesis**: real-time data streaming, persistence and analysis
  + **Amazon MQ**: managed message broker for ActiveMQ and RabbitMQ in the cloud (MQTT, AMQP.. protocols)

# Cloud Monitoring Section

* **Amazon CloudWatch Metrics**
  + CloudWatch provides metrics for every services in AWS
  + Metric is a variable to monitor (CPUUtilization, NetworkIn…)
  + Metrics have timestamps
  + Can create CloudWatch dashboards of metrics

Important Metrics

* + EC2 instances: CPU Utilization, Status Checks, Network (not RAM)
  + Default metrics 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
  + Alarms 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 into an SNS topic
  + Various options (sampling, %, max, min, etc…)
  + Can choose the period on which to evaluate an alarm
  + Example: create **a billing alarm** on the CloudWatch Billing metric
  + Alarm States: OK. INSUFFICIENT\_DATA, ALARM
* **Amazon CloudWatch Logs**
  + CloudWatch Logs 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
* **CloudWatch Logs for EC2**
  + By default, no logs from your EC2 instance will go to CloudWatch
  + You need to run a CloudWatch agent on EC2 to push the log files you want
  + Make sure IAM permissions are correct
  + **The CloudWatch log agent can be setup on-premises too**
* **Amazon EventBridge (formerly CloudWatch Events)**
  + **Schedule: Cron jobs (scheduled scripts)**
    - Schedule Every hour → Trigger script on Lambda function
  + Event Pattern: Event rules to react to a service doing something
    - IAM Root User Sign in Event → SNS Topic with Email Notification
  + Trigger Lambda functions, send SQS/SNS messages…
  + **Schema Registry**: model event schema
  + You can **archive events** (all/filter) sent to an event bus (indefinitely or set period)
  + Ability to **replay archived events**
* **AWS 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 from CloudTrail into CloudWatch Logs 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!
* **AWS X-Ray**
  + 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
  + Enter… AWS X-Ray
* **AWS X-Ray advantages**
  + 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 I am throttled?
  + Identify users that are impacted
* **Amazon CodeGuru**
  + An ML-powered service for automated code reviews and application performance recommendations
  + Provides two functionalities
    - CodeGuru Reviewer: automated code reviews for static code analysis (development)
    - CodeGuru Profiler: visibility/recommendations about application performance during runtime (production)
* **Amazon CodeGuru Reviewer**
  + Identify critical issues, security vulnerabilities, and hard-to-find bugs
  + Example: common coding best practices, resource leaks, security detection, input validation
  + 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
* **Amazon CodeGuru Profiler**
  + Helps understand the runtime behavior of your application
  + Example: identify if your application is consuming excessive CPU capacity on a logging routine
  + Features: • Identify and remove code inefficiencies • Improve application performance (e.g., reduce CPU utilization) • Decrease compute costs • Provides heap summary (identify which objects using up memory) • Anomaly Detection
  + Support applications running on AWS or on- premise
  + Minimal overhead on application
* **AWS Status - Service Health Dashboard**
  + Shows all regions, all services health
  + Shows historical information for each day
  + Has an RSS feed you can subscribe to
* **AWS Personal Health Dashboard**
  + AWS Personal Health Dashboard provides alerts and remediation guidance when AWS is experiencing events that may impact you.
  + While the Service Health Dashboard displays the general status of AWS services, Personal Health Dashboard 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.
  + Global service
  + Shows how AWS outages directly impact you & your AWS resources
  + Alert, remediation, proactive, scheduled activities
* **Monitoring Summary**
  + **CloudWatch**: • **Metrics**: monitor the performance of AWS services and billing metrics • **Alarms**: automate notification, perform EC2 action, notify to SNS based on metric • **Logs**: collect log files from EC2 instances, servers, Lambda functions… • **Events (or EventBridge)**: react to events in AWS, or trigger a rule on a schedule
  + **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 Dashboard**: status of all AWS services across all regions
  + **Personal Health Dashboard**: AWS events that impact your infrastructure
  + **Amazon CodeGuru**: automated code reviews and application performance recommendations

# VPC Section

* **VPC & Subnets Primer**
  + VPC -Virtual Private Cloud: private network to deploy your resources

(regional resource)

* + Subnets allow you to partition your network inside your VPC (Availability Zone resource)
  + A public subnet is a subnet that is accessible from the internet
  + A private subnet is a subnet that is not accessible from the internet
  + To define access to the internet and between subnets, we use Route Tables.
* **Internet Gateway & NAT Gateways**
  + Internet Gateways helps our VPC instances connect with the internet
  + Public Subnets have a route to the internet gateway.
  + NAT Gateways (AWS-managed) & NAT Instances (self-managed) allow your instances in your Private Subnets to access the internet while remaining private
* **Network ACL & Security Groups**
  + NACL (Network ACL)
    - A firewall which 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
    - A firewall that controls traffic to and from an ENI / an EC2 Instance
    - Can have only ALLOW rules
    - Rules include IP addresses and 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 & troubleshoot connectivity issues. Example:
  + Subnets to internet
  + Subnets to subnets Internet to subnets
  + Captures network information from AWS managed interfaces too: Elastic Load Balancers, ElastiCache, RDS, Aurora, etc…
  + 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 the public www network
  + This gives you enhanced security and lower latency to access AWS services
  + VPC Endpoint Gateway: S3 & DynamoDB
  + VPC Endpoint Interface: the rest
* **AWS PrivateLink (VPC Endpoint Services)**
  + Most secure & scalable way to expose a service to 1000s of VPCs
  + Does not require VPC peering, internet gateway, NAT, route tables…
  + Requires a network load balancer (Service VPC) and ENI (Customer VPC)
* **Site to Site VPN & Direct Connect**
  + Site to Site VPN • Connect an on-premises VPN to AWS • The connection is automatically encrypted • Goes over the public internet
  + Direct Connect (DX) • Establish a physical connection between on-premises and AWS • The connection is private, secure and fast • Goes over a private network • Takes at least a month to establish
* **Site-to-Site VPN**
  + On-premises: must use a Customer Gateway (CGW)
  + AWS: must use a Virtual Private Gateway (VGW)
* **AWS Client VPN**
  + Connect from your computer using OpenVPN to your private network in AWS and on-premises
  + Allow you to connect to your EC2 instances over a private IP (just as if you were in the private VPC network)
  + Goes over public Internet
* **Transit Gateway**
  + For having transitive peering between thousands of VPC and on-premises, hub-and-spoke (star) connection
  + One single Gateway to provide this functionality
  + Works with Direct Connect Gateway, VPN connections
* **VPC Closing Comments**
  + VPC: Virtual Private Cloud
  + Subnets:Tied to an AZ, network partition of the VPC
  + Internet Gateway: at the VPC level, provide Internet Access
  + NAT Gateway / Instances: give internet access to private subnets
  + NACL: Stateless, subnet rules for inbound and outbound
  + Security Groups: Stateful, operate at the EC2 instance level or ENI
  + VPC Peering: Connect two VPC with non overlapping IP ranges, nontransitive
* **VPC Closing Comments**
  + VPC Endpoints: Provide private access to AWS Services within VPC
  + PrivateLink: Privately connect to a service in a 3rd party VPC
  + VPC Flow Logs: network traffic logs
  + Site to Site VPN: VPN over public internet between on-premises DC and AWS
  + Client VPN: OpenVPN connection from your computer into your VPC
  + Direct Connect: direct private connection to AWS
  + Transit Gateway: Connect thousands of VPC and on-premises networks together

# Security & Compliance Section

* **AWS Shared Responsibility Model**
  + AWS responsibility - Security of the Cloud
    - Protecting infrastructure (hardware, software, facilities, and networking) that runs all the AWS services
    - Managed services like S3, DynamoDB, RDS, etc.
  + Customer responsibility - Security in the Cloud
    - For EC2 instance, customer is responsible for management of the guest OS (including security patches and updates), firewall & network configuration, IAM
    - Encrypting application data Shared controls:
  + Patch Management, Configuration Management, Awareness & Training
  + Shared controls:
    - Patch Management, Configuration Management, Awareness & Training
* **DDOS Protection on AWS**
  + AWS Shield Standard: protects against DDOS attack for your website and applications, for all customers at no additional costs
  + AWS Shield Advanced: 24/7 premium DDoS protection AWS WAF: Filter specific requests based on rules
  + CloudFront and Route 53:
    - Availability protection using global edge network
    - Combined with AWS Shield, provides attack mitigation at the edge
    - Be ready to scale – leverage AWS Auto Scaling
* **AWS Shield**
  + AWS Shield Standard:
    - Free service that is activated for every AWS customer
    - Provides protection from attacks such as SYN/UDP Floods, Reflection attacks and other layer 3/layer 4 attacks
  + AWS Shield Advanced:
    - Optional DDoS mitigation service ($3,000 per month per organization)
    - Protect against more sophisticated attack on Amazon EC2, Elastic Load Balancing (ELB), Amazon CloudFront, AWS Global Accelerator, and Route 53
    - 24/7 access to AWS DDoS response team (DRP)
    - Protect against higher fees during usage spikes due to DDoS
* **AWS WAF –Web Application Firewall**
  + Protects your web applications from common web exploits (Layer 7)
  + Layer 7 is HTTP (vs Layer 4 is 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 on AWS Cloud**
  + AWS customers are welcome to carry out security assessments or penetration tests against their AWS infrastructure without prior approval for 8 services:
    - Amazon EC2 instances, NAT Gateways, and Elastic Load Balancers
    - Amazon RDS
    - Amazon CloudFront
    - Amazon Aurora
    - Amazon API Gateways
    - AWS Lambda and Lambda Edge functions
    - Amazon Lightsail resources
    - Amazon Elastic Beanstalk environments
  + List can increase over time (you won’t be tested on that at the exam)
* **Penetration Testing on your AWS Cloud**
  + Prohibited Activities
    - DNS zone walking via Amazon Route 53 Hosted Zones
    - Denial of Service (DoS), Distributed Denial of Service (DDoS), Simulated DoS, Simulated DDoS
    - Port flooding
    - Protocol flooding
    - Request flooding (login request flooding, API request flooding)
    - For any other simulated events, contact

[aws-security-simulatedevent@amazon.com](mailto:aws-security-simulatedevent@amazon.com)

* + Read more: <https://aws.amazon.com/security/penetrationtesting/>
* **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, etc.
  + 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
* **AWS KMS (Key Management Service)**
  + Anytime you hear “encryption” for an AWS service, it’s most likely KMS KMS = 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**
  + KMS => AWS manages the software for encryption
  + CloudHSM => AWS provisions encryption hardware
  + Dedicated Hardware (HSM = Hardware Security Module)
  + You manage your own encryption keys entirely (not AWS)
  + HSM device is tamper resistant, FIPS 140-2 Level 3 compliance
* **Types of Customer Master Keys: CMK**
  + Customer Managed CMK:
    - Create, manage and used by the customer, can enable or disable
    - Possibility of rotation policy (new key generated every year, old key preserved)
    - Possibility to 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 CMKs that an AWS service owns and manages to use in multiple accounts
    - AWS can use those to protect resources in your account (but you 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 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
  + Integrations with (load TLS certificates on)
    - Elastic Load Balancers
    - CloudFront Distributions
    - APIs on API Gateway
* **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
  + Mostly meant for RDS integration
* **AWS Artifact (not really a service)**
  + Portal that provides customers with on-demand access to AWS compliance documentation and AWS agreements
  + 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 the Business Associate Addendum (BAA) or the Health Insurance Portability and Accountability Act (HIPAA) for an individual account or in your organization
  + Can be used to support internal audit or compliance
* **Amazon GuardDuty**
  + Intelligent Threat discovery to Protect AWS Account
  + Uses Machine Learning algorithms, anomaly detection, 3party data One click to enable (30 days trial), no need to install software

rd

* + Input data includes:
    - CloudTrail Events Logs – unusual API calls, unauthorized deployments
      * CloudTrail Management Events – create VPC subnet, create trail, …
      * CloudTrail S3 Data Events – get object, list objects, delete object, …
    - VPC Flow Logs – unusual internal traffic, unusual IP address
    - DNS Logs – compromised EC2 instances sending encoded data within DNS queries
    - Kubernetes Audit Logs – suspicious activities and potential EKS cluster compromises
  + Can setup CloudWatch Event rules to be notified in case of findings
  + CloudWatch Events rules can target AWS Lambda or SNS
  + Can protect against CryptoCurrency attacks (has a dedicated “finding” for it)
* **Amazon Inspector**
  + Automated Security Assessments
  + For EC2 instances
    - Leveraging the AWS System Manager (SSM) agent
    - Analyze against unintended network accessibility
    - Analyze the running OS against known vulnerabilities
  + For Container Images push to Amazon ECR
    - Assessment of Container Images as they are pushed
  + For Lambda Functions • Identifies software vulnerabilities in function code and package dependencies • Assessment of functions as they are deployed
  + Reporting & integration with AWS Security Hub
  + Send findings to Amazon Event Bridge

What does Amazon Inspector evaluate?

* + Remember: only for EC2 instances, Container Images & Lambda functions
  + Remember: only for EC2 instances, Container Images & Lambda functions
  + Package vulnerabilities (EC2, ECR & Lambda) – database of CVE
  + Network reachability (EC2)
  + A risk score is associated with all vulnerabilities for prioritization
* **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
* **AWS Macie**
  + Amazon Macie is a fully managed data security and data privacy service that uses machine learning and patt
  + Macie helps identify and alert you to sensitive data, such as personally identifiable information (PII)
* **AWS Security Hub**
  + Central security tool to manage security across several AWS accounts and automate security checks
  + Integrated dashboards showing current security and compliance status 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
* **Amazon Detective**
  + GuardDuty, Macie, and Security Hub are used to identify potential security issues, or findings
  + Sometimes security findings require deeper analysis to isolate the root cause and take action – it’s a complex process
  + Amazon Detective 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 create a unified view
  + Produces visualizations with details and context to get to the root cause
* **AWS Abuse**
  + Report suspected AWS resources used for abusive or illegal purposes
  + Abusive & prohibited behaviors are:
    - Spam – receving undesired emails from AWS-owned IP address, websites & forums spammed by AWS resources
    - Port scanning – sending packets to your ports to discover the unsecured ones
    - DoS or DDoS attacks – AWS-owned IP addresses attempting to overwhlem or crash your servers/softwares
    - Intrusion attempts – logging in on your resources
    - Hosting objectionable or copyrighted content – distributing illegal or copyrighted content without consent
    - Distributing malware – AWS resources distributing softwares to harm computers or machines
  + Contact the AWS Abuse team: AWS abuse form, or [abuse@amazonaws.com](mailto:abuse@amazonaws.com)
* **Root user privileges**
  + Root user = Account Owner (created when the account is created)
  + Has complete access to all AWS services and resources
  + Lock away your AWS account root user access keys!
  + Do not use the root account for everyday tasks, even administrative tasks
  + Actions that can be performed only by the root user: • Change account settings (account name, email address, root user password, root user access keys) • View certain tax invoices • Close your AWS account • Restore IAM user permissions • 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 • Sign up for GovCloud
* **Section Summary: Security & Compliance**
  + **Shared Responsibility on AWS**
  + **Shield**: Automatic DDoS Protection + 24/7 support for advanced
  + **WAF**: Firewall to filter incoming requests based on rules
  + **KMS**: Encryption keys managed by AWS
  + **CloudHSM**: Hardware encryption, we manage encryption keys
  + **AWS Certificate Manager**: provision, manage, and deploy SSL/TLS Certificates
  + **Artifact**: Get access to compliance reports such as PCI, ISO, etc…
  + **GuardDuty**: Find malicious behavior with VPC, DNS & CloudTrail Logs
  + **Inspector**: For EC2 only, install agent and find vulnerabilities
  + **Config**: Track config changes and compliance against rules
  + **Macie**: Find sensitive data (ex: PII data) in Amazon S3 buckets
  + **CloudTrail**: Track API calls made by users within account
  + **AWS Security Hub**: gather security findings from multiple AWS accounts
  + **Amazon Detective**: find the root cause of security issues or suspicious activities
  + **AWS Abuse**: Report AWS resources used for abusive or illegal purposes
  + **Root user privileges**:
    - Change account settings
    - Close your AWS account
    - Change or cancel your AWS Support plan
    - Register as a seller in the Reserved Instance Marketplace

# Machine Learning Section

* **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 • Face Detection and Analysis (gender, age range, emotions…) • Face Search and Verification • Celebrity Recognition • Pathing (ex: for sports game analysis)
* **Amazon Transcribe**
  + Automatically convert speech to text
  + Uses a deep learning process called automatic speech recognition (ASR) to convert speech to text quickly and accurately
  + Automatically remove Personally Identifiable Information (PII) using Redaction
  + Supports Automatic Language Identification for multi-lingual audio
  + 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 tal
* **Amazon Translate**
  + Natural and accurate language translation
  + Amazon Translate allows you to localize content - such as websites and applications - for international users, and to easily translate large volumes of text efficiently.
* **Amazon Lex & Connect**
  + Amazon 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
  + Amazon Connect: • Receive calls, create contact flows, cloud-based virtual contact center • Can integrate 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 • Understands how positive or negative the text is • Analyzes text using tokenization and parts of speech • Automatically organizes a collection of text files by topic
  + Sample use cases: • analyze customer interactions (emails) to find what leads to a positive or negative experience • Create and groups articles by topics that Comprehend will uncover
* **Amazon 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, FAQs…)
  + 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
    - Example: User bought gardening tools, provide recommendations on the next one to buy
  + Same technology used by [Amazon.com](http://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…
* **Amazon Textract**
  + Automatically extracts text, handwriting, and data from any scanned documents using AI and ML
  + Extract data from forms and tables
  + Read and process any type of document (PDFs, images, …)
  + Use cases: • Financial Services (e.g., invoices, financial reports) • Healthcare (e.g., medical records, insurance claims) • Public Sector (e.g., tax forms, ID documents, passports)
* **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
  + Textract: detect text and data in documents

# Account Management, Billing & Support Section

* AWS Organizations
  + Global service
  + Allows to manage multiple AWS accounts
  + The main account is the master account
  + 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
  + API is available to automate AWS account creation
  + **Restrict account privileges using Service Control Policies (SCP)**
* 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
* Service Control Policies (SCP)
  + Whitelist or blacklist IAM actions
  + Applied at the OU or Account level
  + Does not apply to the Master Account
  + SCP is applied to all the Users and Roles of the Account, including Root user
  + The SCP does not affect service-linked roles • Service-linked 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 example: can’t use EMR) • Enforce PCI compliance by explicitly disabling services
* AWS Organization – Consolidated Billing
  + When enabled, provides you with: • Combined Usage – combine the usage across all AWS accounts in the AWS Organization to share the volume pricing, Reserved Instances and Savings Plans discounts • One Bill – get one bill for all AWS Accounts in the AWS Organization
  + The management account can turn off Reserved Instances discount sharing for any account in the AWS Organization, including itself
* AWS Control Tower
  + Easy way to set up and govern a secure and compliant multi-account AWS environment based on best practices
  + Benefits: • Automate the set up of your environment in a few clicks • Automate ongoing policy management using guardrails • Detect policy violations and remediate them • Monitor compliance through an interactive dashboard
  + AWS Control Tower runs on top of AWS Organizations:
  + It automatically sets up AWS Organizations to organize accounts and implement SCPs (Service Control Policies)
* Pricing Models in AWS

AWS has 4 pricing models:

* + **Pay as you go**: pay for what you use, remain agile, responsive, meet scale demands
  + **Save when you reserve**: minimize risks, predictably manage budgets, comply with long-terms requirements
    - Reservations are available for EC2 Reserved Instances, DynamoDB Reserved Capacity, ElastiCache Reserved Nodes, RDS Reserved Instance, Redshift Reserved Nodes
  + **Pay less by using more**: volume-based discounts
  + **Pay less as AWS grows**
* Free services & free tier in AWS
  + IAM
  + VPC
  + Consolidated Billing
  + Elastic Beanstalk
  + CloudFormation
  + Auto Scaling Groups
  + Free Tier:
    - EC2 t2.micro instance for a year
    - S3, EBS, ELB, AWS Data transfer
* Compute Pricing – EC2
  + Only charged for what you use
  + Number of instances
  + Instance configuration:
    - Physical capacity
    - Region
    - OS and software
    - Instance type
    - Instance size
  + ELB running time and amount of data processed
  + Detailed monitoring
  + On-demand instances:
    - Minimum of 60s
    - Pay per second (Linux/Windows) or per hour (other)
  + Reserved instances:
    - Up to 75% discount compared to On-demand on hourly rate
    - 1- or 3-years commitment
    - All upfront, partial upfront, no upfront
  + Spot instances:
    - Up to 90% discount compared to On-demand on hourly rate
    - Bid for unused capacity
  + Dedicated Host:
    - On-demand
    - Reservation for 1 year or 3 years commitment
  + Savings plans as an alternative to save on sustained usage
* Compute Pricing – Lambda & ECS
  + Lambda:
    - Pay per call
    - Pay per duration
    - ECS:
      * EC2 Launch Type Model: No additional fees, you pay for AWS resources stored and created in your application
* Fargate:
  + Fargate Launch Type Model: Pay for vCPU and memory resources allocated to your applications in your containers
* Storage Pricing – S3
  + Storage class: S3 Standard, S3 Infrequent Access, S3 One-Zone IA, S3 Intelligent Tiering, S3 Glacier and S3 Glacier Deep Archive
  + Number and size of objects: Price can be tiered (based on volume)
  + Number and type of requests
  + Data transfer OUT of the S3 region
  + S3 Transfer Acceleration
  + Lifecycle transitions
  + Similar service: EFS (pay per use, has infrequent access & lifecycle rules)
* Storage Pricing - EBS
  + Volume type (based on performance)
  + Storage volume in GB per month provisionned
  + IOPS:
    - General Purpose SSD: Included
    - Provisioned IOPS SSD: Provisionned amount in IOPS
    - Magnetic: Number of requests
  + Snapshots:
    - Added data cost per GB per month
  + Data transfer:
    - Outbound data transfer are tiered for volume discounts
    - Inbound is free
* Database Pricing - RDS
  + Per hour billing
  + Database characteristics: • Engine • Size • Memory class
  + Purchase type: • On-demand • Reserved instances (1 or 3 years) with required up-front
  + Backup Storage: There is no additional charge for backup storage up to 100% of your total database storage for a region.
  + Additional storage (per GB per month)
  + Number of input and output requests per month
  + Deployment type (storage and I/O are variable):
    - Single AZ
    - Multiple AZs
  + Data transfer:
    - Outbound data transfer are tiered for volume discounts
    - Inbound is free
* Content Delivery – CloudFront
  + Pricing is different across different geographic regions
  + Aggregated for each edge locaton, then applied to your bill
  + Data Transfer Out (volume discount)
  + Number of HTTP/HTTPS requests
* Networking Costs in AWS per GB - Simplified
  + Use Private IP instead of Public IP for good savings and better network performance
  + Use same AZ for maximum savings (at the cost of high availability)
* Savings Plan
  + Commit a certain $ amount per hour for 1 or 3 years
  + Easiest way to setup long-term commitments on AWS
  + EC2 Savings Plan
    - Up to 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 (Linux/Windows) or tenancy
    - All upfront, 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 Options: EC2, Fargate, Lambda
  + Machine Learning Savings Plan: SageMaker…
  + Setup from the AWS Cost Explorer console
* AWS Compute Optimizer
  + Reduce costs and improve performance by recommending optimal AWS resources for your workloads
  + Helps you choose optimal configurations and right- size your workloads (over/under provisioned)
  + Uses Machine Learning to analyze your resources’ configurations and their utilization CloudWatch metrics
  + Supported resources
    - EC2 instances
    - EC2 Auto Scaling Groups
    - EBS volumes
    - Lambda function
  + Lower your costs by up to 25%
  + Recommendations can be exported to S3
* Billing and Costing Tools
  + Estimating costs in the cloud:
    - Pricing Calculator
  + Tracking costs in the cloud:
    - Billing Dashboard
    - Cost Allocation Tags
    - Cost and Usage Reports
    - Cost Explorer
  + Monitoring against costs plans:
    - Billing Alarms
    - Budget
* AWS Pricing Calculator
  + Estimate the cost for your solution architecture
* Cost Allocation Tags
  + Use cost allocation tags to track your AWS costs on a detailed level
  + AWS generated tags
    - Automatically applied to the resource you create
    - Starts with Prefix aws: (e.g. aws: createdBy)
  + User-defined tags • Defined by the user • Starts with Prefix user:
* Tagging and Resource Groups
  + 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
  + The AWS Cost & Usage Report contains the most comprehensive set of AWS cost and usage data available, including additional metadata about AWS services, pricing, and reservations (e.g., Amazon EC2 Reserved Instances (RIs)).
  + The AWS Cost & Usage Report lists 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 purposes.
  + 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 in CloudWatch
  + Billing data metric is stored in CloudWatch us-east1
  + Billing data are for overall worldwide AWS costs
  + It’s for actual cost, not for projected costs
  + Intended a simple alarm (not as powerful as AWS Budgets
* AWS Budget
  + Create budget and send alarms when costs exceeds the budget
  + 3 types of budgets: Usage, Cost, Reservation
  + For Reserved Instances (RI) • Track utilization • Supports EC2, ElastiCache, RDS, Redshift
  + Up to 5 SNS notifications per budget
  + Can filter by: Service, Linked Account, Tag, Purchase Option, Instance Type, Region, Availability Zone, API Operation, etc…
  + Same options as AWS Cost Explorer!
  + 2 budgets are free, then $0.02/day/budget
* Trusted Advisor
  + No need to install anything – high level AWS account assessment
  + Analyze your AWS accounts and provides recommendation on 5 categories
  + **Cost optimization**
  + **Performance**
  + **Security**
  + **Fault tolerance**
  + **Service limits**
* AWS Support Plans Pricing

Basic Support: free

* + Customer Service & 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

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 hour
* AWS Business Support Plan (24/7)
  + Intended to be used if you have production workloads
  + Trusted Advisor – Full set of checks + API access
  + 24x7 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
* AWS Enterprise On-Ramp Support Plan (24/7)
  + Intended to be used if you have production or business critical workloads
  + All of Business Support Plan +
  + Access to a pool of Technical Account Managers (TAM)
  + Concierge Support Team (for billing and account best practices)
  + Infrastructure Event Management, Well-Architected & Operations Reviews
  + Case severity / response times: • … • Production system impaired: < 4 hours • Production system down: < 1 hour • Business-critical system down: < 30 minutes
* AWS Enterprise Support Plan (24/7)
  + Intended to be used if you have mission critical workloads
  + All of Business Support Plan +
  + Access to a designated Technical Account Manager (TAM)
  + Concierge Support Team (for billing and account best practices)
  + Infrastructure Event Management, Well-Architected & Operations Reviews
  + Case severity / response times: • … • Production system impaired: < 4 hours • Production system down: < 1 hour • Business-critical system down: < 15 minutes
* Account Best Practices – Summary
  + Account Best Practices – Summary
  + Operate multiple accounts using Organizations
  + Use SCP (service control policies) to restrict account power
  + Easily setup multiple accounts with best-practices with AWS Control Tower
  + Use Tags & Cost Allocation Tags for easy management & billing
  + IAM guidelines: MFA, least-privilege, password policy, password rotation
  + Config to record all resources configurations & compliance over time
  + CloudFormation to deploy stacks across accounts and regions
  + Trusted Advisor to get insights, Support Plan adapted to your needs
  + Send Service Logs and Access Logs to S3 or CloudWatch Logs
  + CloudTrail to record API calls made within your account
  + If your Account is compromised: change the root password, delete and rotate all passwords / keys, contact the AWS support
* Billing and Costing Tools – Summary
  + **Compute Optimizer:** recommends resources’ configurations to reduce cost
  + **Pricing Calculator**: cost of services on AWS
  + **Billing Dashboard**: high level overview + free tier dashboard
  + **Cost Allocation Tags**: tag resources to create detailed reports
  + **Cost and Usage Reports**: most comprehensive billing dataset
  + **Cost Explorer**:View current usage (detailed) and forecast usage
  + **Billing Alarms**: in us-east-1 – track overall and per-service billing
  + **Budgets**: more advanced – track usage, costs, RI, and get alerts
  + **Savings Plans**: easy way to save based on long-term usage of AWS