In28 Minutes

AWS
Interview Guide

Getting Started

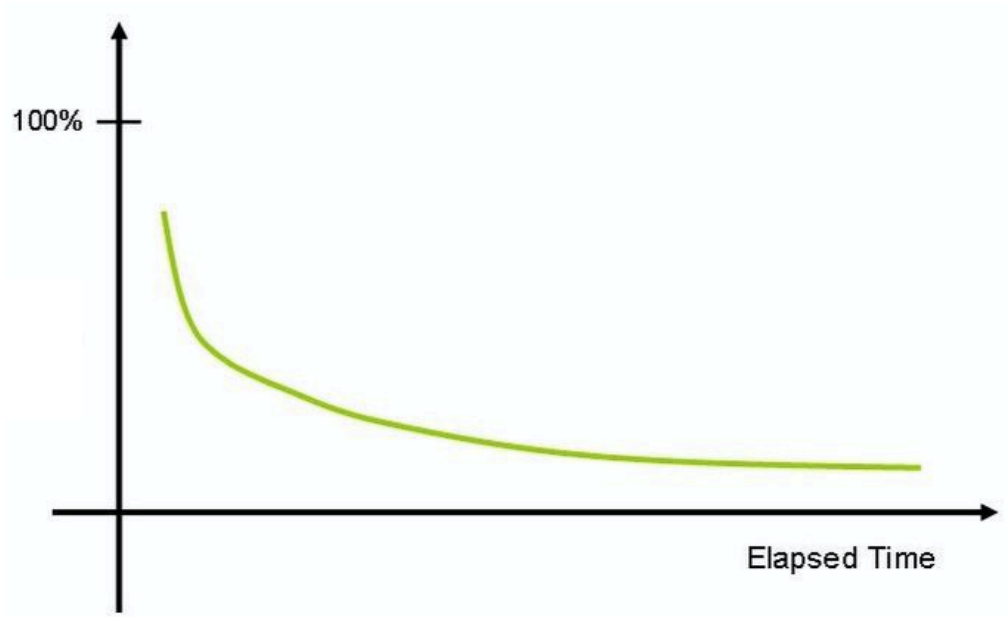
In28 Minutes

- AWS has **200+ services**
- You might have a lot of experience with AWS
  - BUT you might not remember all the details
- **How do you quickly review before an AWS interview?**
- **Our Goal** : Enable you to quickly review for **an AWS interview**
  - This is **NOT an AWS 101** course
- **What's More**: We are heading to a **multi-cloud** world
  - We will touch upon Azure and Google Cloud!

How do you put your best foot forward?

In28 Minutes

- **AWS can be difficult** - 100+ services to remember
- As time passes, humans forget things
- How do you improve your chances of remembering things?
  - **Active learning** - think & take notes
  - **Review** the presentation every once in a while



FASTEST ROADMAPS

in28minutes.com

In28 Minutes

- Google Cloud Certifications
- Azure Certifications
- AWS Certifications
- DevOps
- Java Full Stack
- Java Microservices

AWS - Getting started

Why Cloud?

In28 Minutes

- Cloud: **On-demand resource provisioning**
  - **Provisioning (renting) resources** when you want them
  - Releasing them back when you do not need them

- Advantages:
  - Avoid **undifferentiated heavy lifting**
    - Stop spending money running and maintaining data centers
  - **"Go global"** in minutes
  - Benefit from massive **economies of scale**
  - Stop **guessing** capacity
  - Trade **"capital expense"** for **"variable expense"**

Why AWS?

In28 Minutes

- One of the 3 **leading** cloud service providers
- Reliable, secure and cost-effective
- Provides **200+ services** under various categories
  - **Compute** - EC2, Elastic Beanstalk, Lambda, ECS, EKS, ..
  - **Storage** - EBS, EFS, S3, S3 Glacier, Storage Gateway, ..
  - **Database** - RDS, DynamoDB, Amazon DocumentDB
  - **Networking & Content Delivery** - VPC, Direct Connect, ..
  - **Security, Identity, & Compliance** - IAM, KMS, CloudHSM, ...
  - **DevOps** - CloudFormation, CodePipeline, CloudWatch, X-Ray, CloudTrail, ..
  - **Analytics** - Amazon Redshift, Athena, Data Pipeline, Kinesis, EMR
  - **Machine Learning** - Amazon SageMaker, Amazon Textract, Amazon Transcribe, Amazon Translate, Amazon Rekognition, Amazon Polly
  - **And a lot of others** - Simple Notification Service, Simple Queue Service, ..

Regions and Availability Zones

In28 Minutes

Region Code	Region	Availability Zones	Availability Zones List
us-east-1	US East (N. Virginia)	6	us-east-1a us-east-1b us-east-1c us-east-1d us-east-1e us-east-1f
eu-west-2	Europe (London)	3	eu-west-2a eu-west-2b eu-west-2c

- AWS provides **25+ regions** around the world
  - **Advantages** - High Availability, Low Latency, Global Footprint & Adhere to **regulations**
  - Choose region(s) based on - Users Location, Data Location, Regulatory and compliance needs
  - AWS Services can be Regional (OR) Global
- Each AWS Region has **at least two** Availability Zones
  - **Isolated locations** in a Region (**independent** & **redundant** power, networking & connectivity)
  - **GOAL**: **Increase availability** of applications in the same region

Compute Services

Managed Services

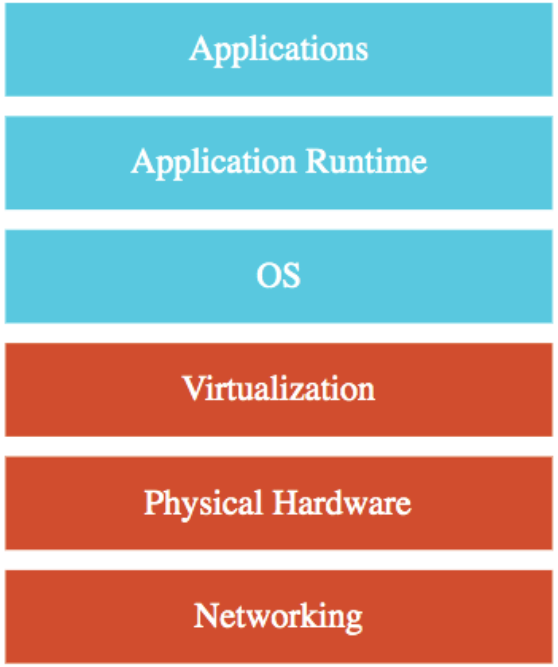
In28 Minutes

- Do you want to continue **running applications in the cloud**, the **same way you run them** in your **data center**?
- OR **are there OTHER approaches**?
- Let's quickly **review some terminology**:
  - **IaaS** (Infrastructure as a Service)
  - **PaaS** (Platform as a Service)
  - **Serverless**
  - **Containers**
  - **Container Orchestration**

IAAS (Infrastructure as a Service)

In 28 Minutes

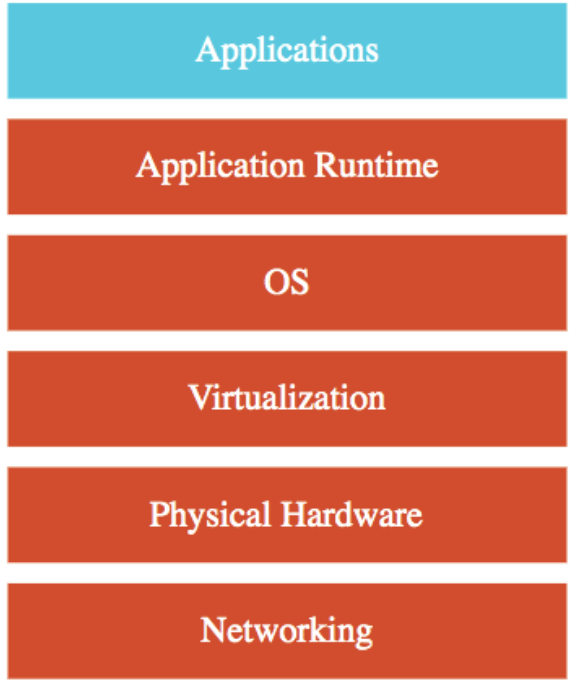
- Use **only infrastructure** from cloud provider
  - **Ex**: Using VM service to deploy your apps/databases
- **Cloud provider** is responsible for:
  - Hardware, Networking & Virtualization
- You are responsible for:
  - OS upgrades and patches
  - Application Code and Runtime
  - Configuring load balancing
  - Auto scaling
  - Availability
  - etc.. (and a lot of things!)



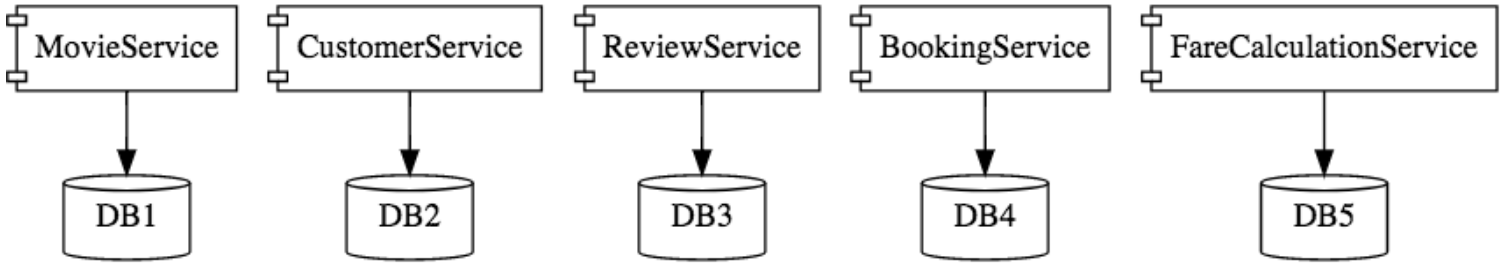
PAAS (Platform as a Service)

In28 Minutes

- Use a platform provided by the cloud
  - **Cloud provider** is responsible for:
    - Hardware, Networking & Virtualization
    - OS (incl. upgrades and patches)
    - Application Runtime
    - Auto scaling, Availability & Load balancing etc..
  - **You** are responsible for:
    - Configuration (of Application and Services)
    - Application code (if needed)
- **Examples:**
  - Compute: AWS Elastic Beanstalk, Azure App Service, Google App Engine
  - Databases - Relational & NoSQL (Amazon RDS, Google Cloud SQL, Azure SQL Database etc)
  - Queues, AI, ML, Operations etc!



Microservices

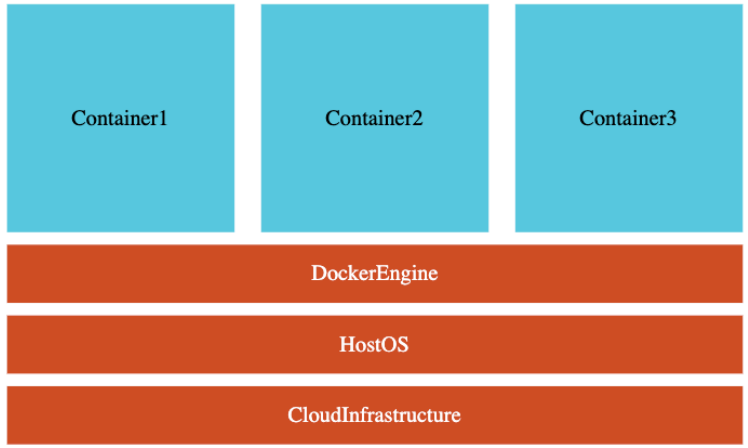


- Enterprises are **heading towards microservices architectures**
  - Build small focused microservices
  - **Flexibility to innovate** and build applications in different programming languages (Go, Java, Python, JavaScript, etc)
- BUT **deployments become complex**!
- How can we have **one way of deploying** Go, Java, Python or JavaScript .. microservices?
  - Enter **containers**!

Containers - Docker

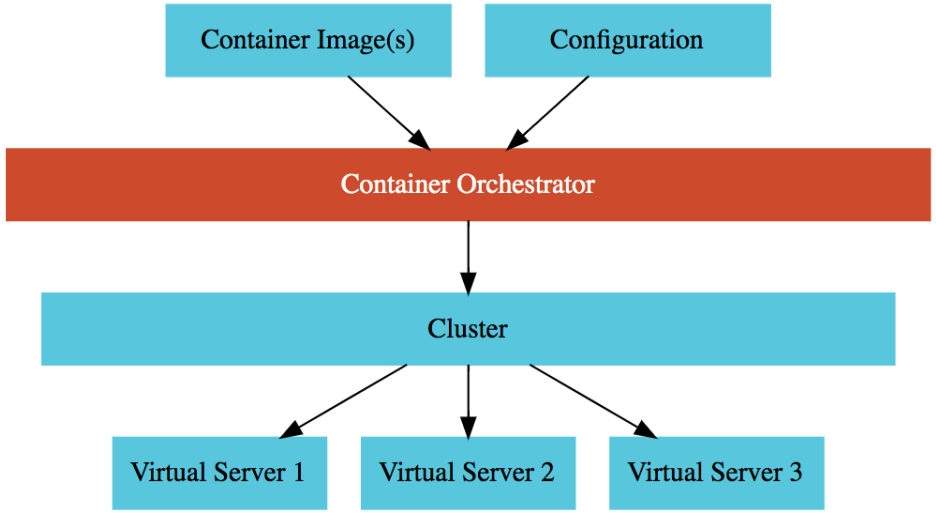
In28 Minutes

- Create **Docker images** for each microservice
- Docker image **has all needs of a microservice**:
  - Application Runtime (JDK or Python or NodeJS)
  - Application code and Dependencies
  - Runs **the same way** on any infrastructure:
    - Your local machine
    - Corporate data center
    - Cloud
- Advantages
  - Docker containers are **light weight**
    - Compared to Virtual Machines as they do not have a Guest OS
  - Docker provides **isolation** for containers
  - Docker is **cloud neutral**



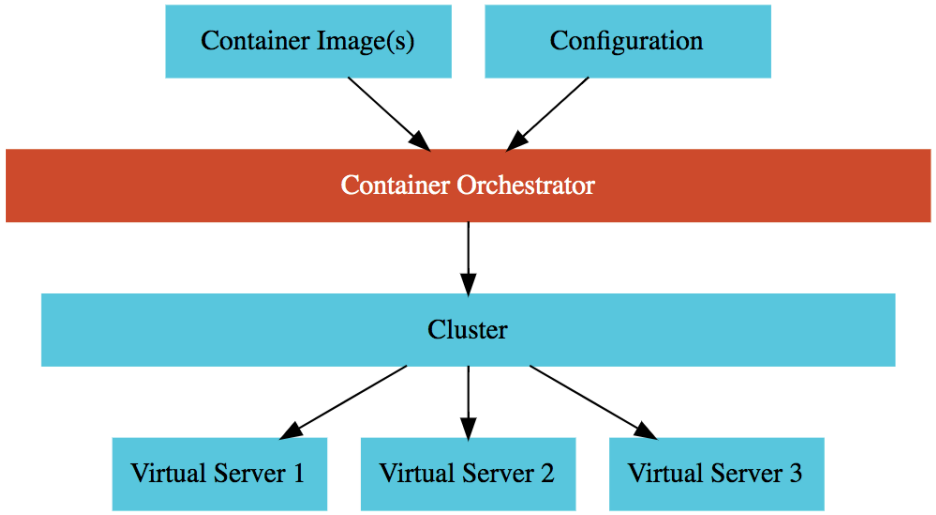
Container Orchestration

- **Requirement** : I want 10 instances of Microservice A container, 15 instances of Microservice B container and ....
- Typical Features:
  - **Auto Scaling** - Scale containers based on demand
  - **Service Discovery** - Help microservices find one another
  - **Load Balancer** - Distribute load among multiple instances of a microservice
  - **Self Healing** - Do health checks and replace failing instances
  - **Zero Downtime Deployments** - Release new versions without downtime



Container Orchestration Services

- Using a **Container Orchestrator**:
  - 1: Create a Cluster
  - 2: Deploy & Orchestrate Microservices
- **Managed Kubernetes Services**:
  - **Elastic Kubernetes Service** (AWS EKS)
  - **Google Kubernetes Engine** (GKE)
  - **Azure Kubernetes Service** (AKS)
- **Cloud Specific Services**:
  - Elastic Container Service (AWS ECS)
  - Azure Service Fabric



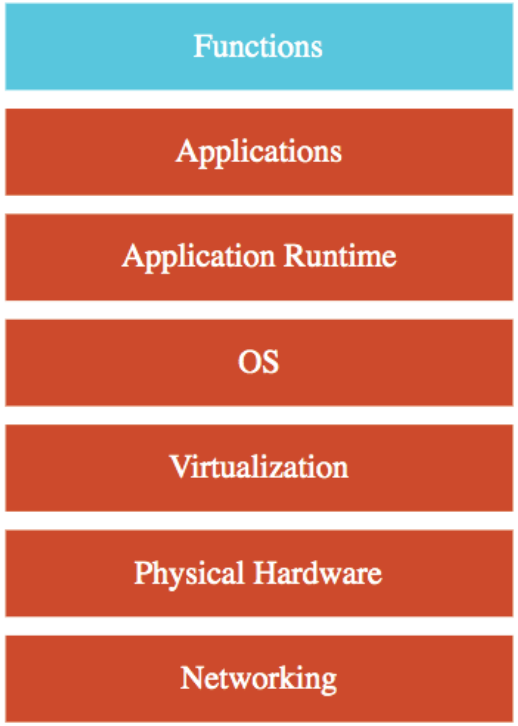
Serverless

- What do we think about **when we build an application**?
  - Where to deploy? What kind of server? What OS?
  - How do we take care of scaling and availability of the application?
- **WHAT IF** you **don't need to worry about servers** & **focus on code?**
  - Enter **Serverless**
    - Remember: **Serverless does NOT mean "No Servers"**
- **Serverless for me:**
  - You **don't worry** about infrastructure (ZERO visibility into infrastructure)
    - Flexible scaling and automated high availability
  - Most Important: **Pay for use**
    - Ideally ZERO REQUESTS => ZERO COST
- **You focus on code** and the cloud managed service takes care of all that is needed to scale your code to **serve millions of requests**!
  - And **you pay for requests** and NOT servers!

Serverless Function Services

In28 Minutes

- Serverless **Function** services are very popular
  - You **ONLY worry about code (or functions)**
  - and **PAY for USE**
    - Number of requests
    - Duration of requests
    - Memory consumed
  - **Typically supported**: C#, Python, JavaScript, Typescript, Java, ..
- **Serverless Function Cloud Services**
  - AWS Lambda
  - Azure Functions
  - Google Cloud Functions



Serverless is NOT limited to Functions

In28 Minutes

- Serverless **container** services:
  - AWS Fargate (for EKS and ECS)
  - Google Cloud Run
  - GKE Autopilot Mode
- Serverless **database** services:
  - AWS DynamoDB
  - Cloud Datastore (Google Cloud)
  - BigQuery (Google Cloud)
- And a lot of others:
  - Queuing (Cloud Pub/Sub, Amazon SQS, ...)
  - ....

How to choose a Compute Service?

In28 Minutes

Service	Description
Amazon EC2 + ELB	Traditional Approach (Virtual Servers + Load Balancing)
AWS Elastic Beanstalk	Simplify management of web apps and simple batch apps. Automatically creates EC2 + ELB (load balancing and auto scaling)
AWS Elastic Container Service (ECS)	Simplify running of microservices with Docker containers. Run containers in EC2 based ECS Clusters.
Amazon EKS (Elastic Kubernetes Service)	Run Managed Kubernetes Clusters in AWS
AWS Fargate	Serverless version of ECS/EKS
AWS Lambda	Serverless - Do NOT worry about servers
AWS Batch	Manage batch jobs (AWS Fargate, Amazon EC2 & Spot Instances)

EC2 (Elastic Compute Cloud)

EC2 (Elastic Compute Cloud)

In28 Minutes

- **EC2 instances** - Virtual machines in AWS (billed by second)
- **EC2 service** - Provision EC2 instances
- **Features**:
  - Create and manage lifecycle of EC2 instances
  - **Load balancing** and **auto scaling** for multiple EC2 instances
  - **Attach storage** (& network storage) to your EC2 instances
  - Manage **network connectivity** for an EC2 instance

EC2 Instance Types - Choose Hardware

In28 Minutes

- Optimized combination of **compute(CPU, GPU), memory, disk (storage) and networking** for specific workloads
  - **m (m5, m6) - General Purpose.** Balance of compute, memory, and networking.
  - **t (t2, t3, t3a) - Burstable performance instances** (accumulate CPU credits when inactive). Workloads with spikes : web servers, developer environments and small databases.
  - **c (c5, c6, c7) - Compute optimized.**. Batch processing, high performance computing (HPC)
  - **r (R4, R5, R6) - Memory (RAM) optimized**. Memory caches and in-memory databases.
  - **i (i4, d3) - Storage (I/O) optimized**. NoSQL databases and data warehousing.
  - **g (g4, g5) - GPU optimized**. FP Calculations, graphics processing, or video compression.
  - **t2.micro**:
    - **t** - Instance Family
    - **2** - generation. Improvements with each generation.
    - **micro** - size. *(nano < micro < small < medium < large < xlarge < .....)*
- Size increases => compute, memory and networking capabilities increase

EC2 Amazon Machine Image - AMI - Choose OS and Software

In 28 Minutes

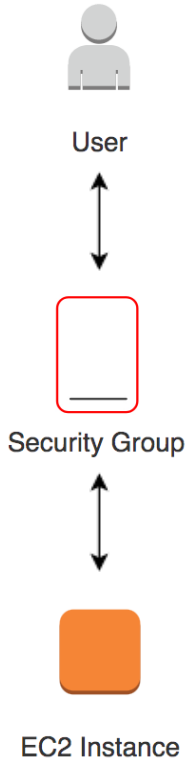
- **AMI**: What OS and what software do you want on the instance?
- Three AMI sources:
  - Provided by AWS
  - **AWS Market Place**: Online store for customized AMIs. Per hour billing
  - **Customized AMIs**: Created by you.
- AMIs contain:
  - Root volume block storage (OS and applications)
  - Block device mappings for non-root volumes
- Configure launch permissions on an AMI(Who can use the AMI?)
  - **Share your AMIs** with other AWS accounts
- AMIs are stored in Amazon S3 (**region specific**)
- **Best Practice**: Backup upto date AMIs in multiple regions
  - Critical for Disaster Recovery

EC2 AMI

EC2 Security - Security Groups

In28 Minutes

- **Virtual firewall** to control **incoming and outgoing** traffic to/from AWS resources (EC2 instances, databases etc)
  - Provides additional layer of security - Defense in Depth
  - Security groups are **default deny**. NO RULES => NO ACCESS.
  - You can specify **allow rules ONLY**
  - You can configure **separate rules** for inbound and outbound traffic
  - Security Groups are **stateful**:
    - If an outgoing request is allowed, the incoming response for it is automatically allowed.
    - If an incoming request is allowed, an outgoing response for it is automatically allowed
- You can add and delete security groups to EC2 instances at any time.
  - Changes are immediately effective
- Traffic NOT explicitly allowed by Security Group **will not reach** the EC2 instance



EC2 Security - Key Pairs

- EC2 uses **public key cryptography** to protect login credentials
- **Key pair** - public key and a private key
  - Public key is stored in EC2 instance
- Connecting to EC2 instance(s) - Troubleshooting:
  - You need to have the **private key** with you
  - Change permissions to **0400** (*chmod 400 /path/my-key-pair.pem*)
    - Default permissions on private key - 0777 (**VERY OPEN**)
  - (Windows Instances) In addition to private key, you need admin password
    - (At Launch) Random admin password is generated and encrypted using public key
    - Decrypt the password using the private key and use it to login via RDP
  - Security Group should allow inbound SSH or RDP access:
    - Port 22 - Linux EC2 instance (SSH)
    - Port 3389 - RDP (Remote Desktop - Windows)

EC2 Pricing Models

EC2 Pricing Models Overview

In28 Minutes

Pricing Model	Description
On Demand	Request when you want it. Flexible and Most Expensive. Web app with spiky traffic. Unpredictable batch which cannot be interrupted.
Spot	Cheapest (upto 90% off). Quote the maximum price. Terminated with **2 minute** notice. Cost sensitive, Fault tolerant, Non immediate workloads.
Reserved	Reserve ahead of time. Upto 75% off. 1 or 3 years reservation. Long Term Workloads. No Upfront or Partial Upfront or All Upfront Payments
Savings Plans	Commit spending $X per hour on (EC2 or AWS Fargate or Lambda). Upto 66% off. No restrictions. 1 or 3 years reservation.

EC2 Spot instances

In28 Minutes

- **(Old Model)** Bid a price. Highest bidder wins
- **(New Model)** Quote your maximum price. Prices decided by long term trends.
- Up to 90% off (compared to On-Demand)
- Can be terminated with a **2 minute** notice
- Ideal for **Non time-critical workloads** that can **tolerate interruptions** (fault-tolerant)
  - A batch program that does not have a strict deadline AND can be stopped at short notice and re-started
- Variations:
  - **Spot Block**: Request Spot instances for **specific duration** (1 or 2 .. or 6 hrs)
  - **Spot Fleet**: Request multiple spot instances across a **range of instance types**
    - More instance types => better chances of fulfilling your spot request

EC2 Reserved Instances

In28 Minutes

- **Standard**: Commit to EC2 platform and an instance family for 1 year or 3 years. (Up to 75% off)
- **Convertible**: Standard + **flexibility** to change EC2 platform and instance family. (Up to 54% off)
- (NOT AVAILABLE ANYMORE)**Scheduled**: Reserve for **specific time period** in a day. (5% to 10% off)
- You can **sell reserved instances** on the AWS Reserved instance marketplace if you do not want to use your reservation

EC2 Savings Plans

In28 Minutes

- **Compute Savings Plans**
  - **Commitment** : I would spend X dollars per hour on AWS compute resources (Amazon EC2 instances, AWS Fargate and/or AWS Lambda) for a 1 or 3 year period
  - Up to 66% off (compared to on demand instances)
  - Provides **complete flexibility**:
    - You can change instance family, size, OS, tenancy or AWS Region of your Amazon EC2 instances
    - You can switch between Amazon EC2, AWS Fargate and/or AWS Lambda
- **EC2 Instance Savings Plans**
  - **Commitment** : I would spend X dollars per hour on Amazon EC2 instances of a specific instance family (General Purpose, for example) within a specific region (us-east-1, for example)
  - Up to 72% off (compared to on demand instances)
  - You can switch operating systems (Windows to Linux, for example)

Scaling EC2 instances

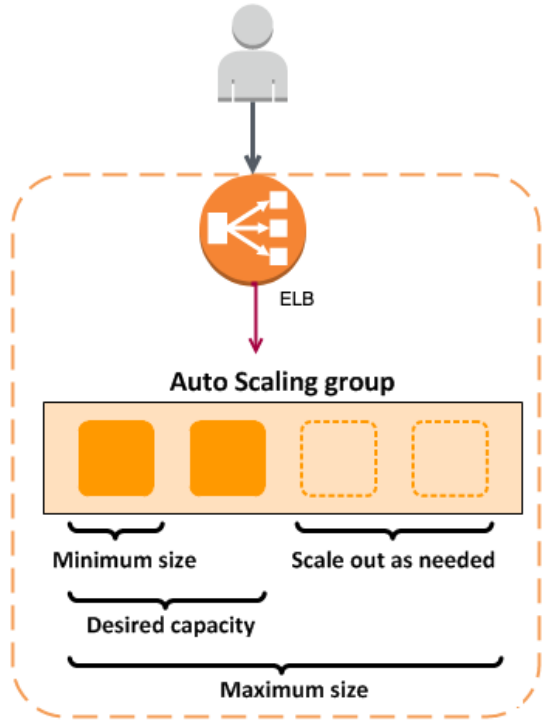
Scalability

In 28 Minutes

- Can we handle **growth in users, traffic, or data** without drop in performance?
  - Does ability to serve more growth increase **proportionally** with resources?
- **Vertical Scaling** - Deploying to a **bigger instance**:
  - More RAM, CPU, I/O, or networking capabilities
  - Increasing **EC2 instance size**: Example: *t2.micro* to *t2.small*
- **Horizontal Scaling** - Deploying multiple instances:
  - (Typically but not always) Horizontal Scaling is preferred to Vertical Scaling:
    - Horizontal scaling increases availability
    - Vertical scaling has limits and can be expensive
  - (BUT) Horizontal Scaling needs additional infrastructure: Load Balancers etc.
  - Create Multiple EC2 instances:
    - In one AZ (OR) multiple AZs in one region (OR) multiple AZs in multiple regions
    - **Auto scale**: Auto Scaling Group. **Distribute load** : Elastic Load Balancer.

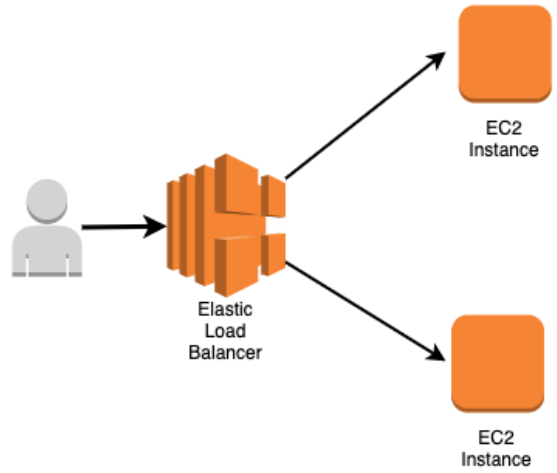
Auto Scaling Group

- How to scale out and scale in **automatically**?
  - Configure a Auto Scaling Group
- Auto Scaling Group responsibilities:
  - **Maintain** configured number of instances (using health checks)
    - If an instance goes down, ASG launches replacement instance
  - **Auto scale** to adjust to load
    - Scale-in and scale-out based on auto scaling policies
  - Can launch On-Demand Instances, Spot Instances, or both
    - **Best Practice**: Use Launch Template
- Auto Scaling Group components:
  - **Launch Configuration/Template** - EC2 instance size and AMI
  - **Auto Scaling Group**
    - Min, max and desired size of ASG
    - EC2 health checks by default. Optionally enable ELB health checks.
    - **Auto Scaling Policies** - When and How to execute scaling?



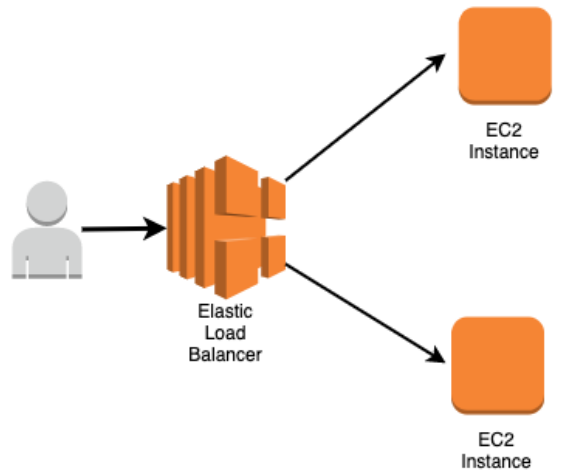
Elastic Load Balancer

- **ELB**: Distribute traffic across Multiple Targets
  - EC2 instances can be in one or more AZs in a single region
  - **Managed service**:
    - AWS ensures that it is highly available
    - Auto scales to handle huge loads
    - Load Balancers can be **public or private**
  - Types:
    - **Classic** Load Balancer (Layer 4 and Layer 7): (NOT RECOMMENDED)
      - Old generation supporting Layer 4(TCP/TLS) and Layer 7(HTTP/HTTPS) protocols
    - **Application** Load Balancer (Layer 7)
      - New Gen - HTTP/HTTPS and advanced routing approaches.
    - **Network** Load Balancer (Layer 4)
      - New Gen - TCP/TLS and UDP
      - Very high performance usecases
    - **Gateway Load Balancers** (Layer 3):
      - Manage third-party virtual appliances (typically)
        - Virtual appliances enable you to improve security, compliance, and policy controls



Application Load Balancer

- **Most popular** and frequently used ELB in AWS
- Supports WebSockets and HTTP/HTTPS (Layer 7)
- Supports all important load balancer features
- Scales **automatically** based on demand (Auto Scaling)
- Can load balance between:
  - EC2 instances (AWS)
  - Containerized applications (Amazon ECS)
  - Web applications (using IP addresses)
  - Lambdas (serverless)



Best Practices for EC2 & ELB

In28 Minutes

Security

- Use **Security Groups** to restrict traffic
- Place EC2 instances in **private subnets**
- Use **Dedicated Hosts** when you have regulatory needs

Performance

- Choose right **instance family** (Optimized combination of compute, memory, disk (storage) and networking)
- Prefer creating an **custom AMI** to installing software using userdata
- Choose the right ELB for your use case
  - Prefer Network Load Balancer for **high performance** load balancing

Best Practices for EC2 & ELB - 2

In28 Minutes

Cost Efficiency

- Have optimal **number and type** of EC2 instances running
- Use the **right mix** of:
  - Savings Plans
  - Reserved Instances
  - On demand Instances
  - Spot Instances

Resiliency

- Configure the right **health checks**
- Use CloudWatch for monitoring
- **(Disaster recovery)** Upto date AMI copied to multiple regions

AWS Elastic BeanStalk

AWS Elastic BeanStalk

In 28 Minutes

- **Simplest way** to deploy and scale your web application in AWS
  - Provides end-to-end web application management
    - Programming languages (Go, Java, Node.js, PHP, Python, Ruby)
    - Application servers (Tomcat, Passenger, Puma)
    - Docker containers (Single and Multi Container Options)
    - **No usage charges** - Pay only for AWS resources you provision
    - **Features**: Load Balancing, Auto scaling and Managed Platform updates
- **Concepts**:
  - **Application** - A container for environments, versions and configuration
  - **Application Version** - A specific version of deployable code (stored in S3)
  - **Environment** - An application version deployed to AWS resources.
    - Create multiple environments running diff. application versions
  - **Environment Tier**:
    - For batch applications, use **worker tier**
    - For web applications, use **web server tier**

AWS Elastic Beanstalk Environment Tiers

In28 Minutes

- **Web Server Tier** : Run web applications
  - **Single-instance environments**: EC2 + Elastic IP
  - **Load-balanced environments**: ELB + ASG + EC2
  - (OPTIONAL) Add **database** to Elastic Beanstalk Env:
    - Use environment properties to connect to database
      - RDS_HOSTNAME, RDS_PORT, RDS_DB_NAME, RDS_USERNAME, RDS_PASSWORD
  - Lifecycle of database **tied** to Elastic Beanstalk Env:
    - If you delete Elastic Beanstalk environment, database also deleted
      - (WORKAROUND): Enable Delete Protection on RDS
      - (WORKAROUND): Take Database Snapshot and Restore
    - NOT RECOMMENDED for Production Deployment
- **Worker Tier**: Run Batch Applications: ASG + EC2 + SQS
  - Process messages from SQS queues
  - Trigger auto scaling using AWS CloudWatch alarms
  - **Schedule** tasks using `cron.yaml`

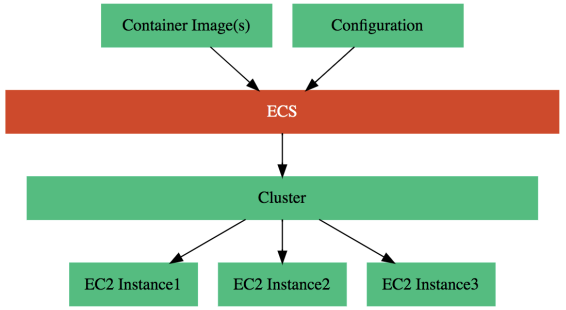
AWS Elastic BeanStalk - Remember

In 28 Minutes

- You **retain full control** over AWS resources created
- **Ideal for simple web** applications
  - **NOT ideal for microservices** architectures
- Logs can be stored in Amazon S3 or in CloudWatch Logs
- You can choose to **apply patches and platform updates** automatically
- Metrics are send to Amazon CloudWatch
- You can configure SNS notifications based on health

Container Orchestration in AWS

- Microservices are built in multiple languages
  - Go, Java, Python, JavaScript, etc
- Containers simplify deployment of microservices:
  - Step I : Create a self contained Docker image
    - Application Runtime (JDK or Python), Application code and Dependencies
  - Step II : Run it as a container any where
    - Local machine OR Corporate data center OR Cloud
- **Container Orchestration**: Manage 1000s of containers
  - **Elastic Container Service (ECS)** - AWS Specific
    - **Step I** : Create a Cluster (Group of one or more EC2 instances)
    - **Step II**: Deploy your microservice containers
  - **Cloud Neutral**: Kubernetes
    - AWS - AWS Elastic Kubernetes Service (EKS)
  - **AWS Fargate**: Serverless ECS/EKS. DONT worry about EC2 instances.



AWS Lambda

Going Serverless with AWS Lambda

In 28 Minutes

- **Serverless - Don't worry about servers. Focus on building your app**
  - Remember: **Serverless does NOT mean "No Servers"**
  - **Serverless for me**:
    - You don't worry about infrastructure
    - Flexible scaling and automated high availability
    - Pay for use NOT FOR SERVERS
  - **You focus on code** and the cloud managed service takes care of all that is needed to scale your code to serve millions of requests!
- **AWS Lambda** - Write and Scale Your Business Logic
  - Supports Node.js (JavaScript), Java, Python, Go, C# and more..
  - Don't worry about servers or scaling or availability
  - Pay for Use: Number of requests, Duration of requests and Memory Configured
    - Free tier - 1M free requests per month
  - Integrates with AWS X-Ray(tracing), AWS CloudWatch (monitoring and logs)

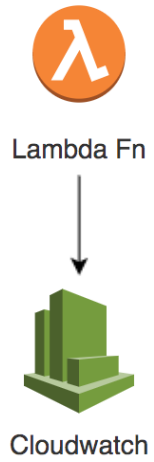
AWS Lambda

API Gateway

DynamoDB

AWS Lambda - Remember

- Allocate memory (128 MB to 10,240 MB)
  - More Memory => More Cost and More CPU
- Maximum allowed execution time - 900 seconds (default - 3 seconds)
- **Lambda execution role** - Grants permissions to AWS Resources
  - Assigned when creating a function
    - Function assumes this role when invoked
  - Predefined policies simplify permission configuration:
    - **AWSLambdaBasicExecutionRole** – Upload logs to CloudWatch.
    - Others include **AWSLambdaDynamoDBExecutionRole**, **AWSLambdaSQSQueueExecutionRole**, **AWSLambdaVPCAccessExecutionRole**, **AWSXRayDaemonWriteAccess**
- Lambda logs are automatically stored in CloudWatch Logs
  - Default log group name: `/aws/lamdba/function-name`
  - If logs are not visible:
    - Check if Lambda Function has permissions to write to CloudWatch Logs(Execution role)



Data - Databases & Storage

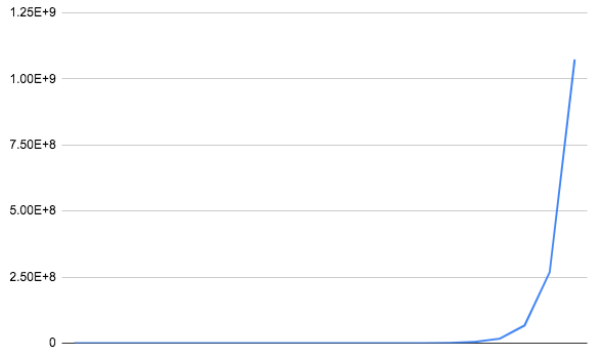
Data

- Data is the **"oil of the 21st Century Digital Economy"**
- Amount of data generated **increasing exponentially**
  - Mobile devices, IOT devices, application metrics etc
  - **Variety of**
    - **Data formats**: Structured, Semi Structured and Unstructured
    - **Data store options**: Relational databases, NoSQL databases, Analytical databases, Object/Block/File storage ...
- Store data efficiently and gain intelligence
- **Goal**: Help you choose specific data format and the data store for your use case

Data Formats & Data Stores

In28 Minutes

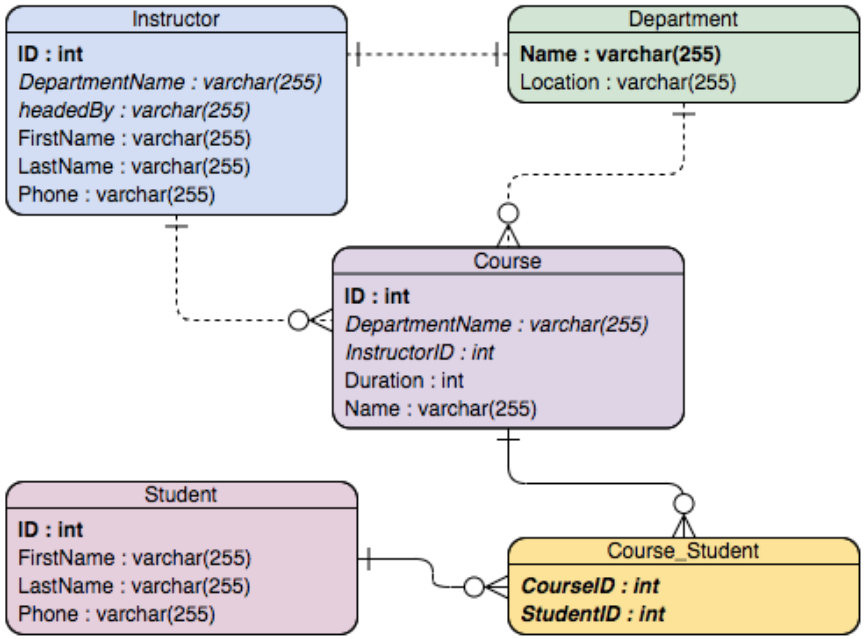
- **Data formats**:
  - **Structured**: Tables, Rows and Columns (Relational)
  - **Semi Structured**: Key-Value, Document (JSON), Graph, etc
  - **Unstructured**: Video, Audio, Image, Text files, Binary files ...
- **Data stores**:
  - Relational databases
  - NoSQL databases
  - Analytical databases
  - Object/Block/File storage



Structured Data - Relational Databases

- **Data stored in Tables** - Rows & Columns
- **Predefined schema** - Tables, Relationships and Constraints
- **Define indexes** - Query efficiently on all columns
- Used for
  - OLTP (Online Transaction Processing) use cases and
  - OLAP (Online Analytics Processing) use cases

ID	DepartmentName	Name	Duration	InstructorID
1	Computer Science	Algorithms	8	2
2	Computer Science	Data Structures	6	4
3	Computer Science	Operating Systems	5	4
4	Computer Science	Database Management Systems	20	2



Relational DB - OLTP (Online Transaction Processing)

In 28 Minutes

- **OLTP**: Applications where **large number of users make large number (millions) of transactions**
  - Transaction - small, discrete, unit of work (Ex: Transfer money to your friend's account)
  - Heavy writes and moderate reads
  - Quick processing expected
- **Use cases**: Most traditional applications - banking, e-commerce, ..
- Cloud **Managed Services**:
  - **AWS**: Amazon RDS (PostgreSQL/MySQL/MariaDB/Oracle/SQLServer), Amazon Aurora
  - **Azure**: Azure SQL Database (SQL Server), Azure Database for MySQL/PostgreSQL/MariaDB
  - **Google Cloud**: Cloud SQL (MySQL/PostgreSQL/SQL Server), Cloud Spanner

Relational DB - OLAP (Online Analytics Processing)

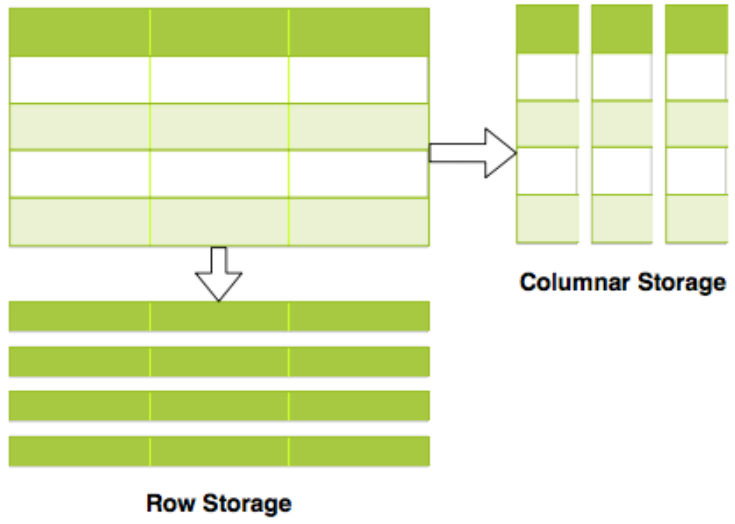
In 28 Minutes

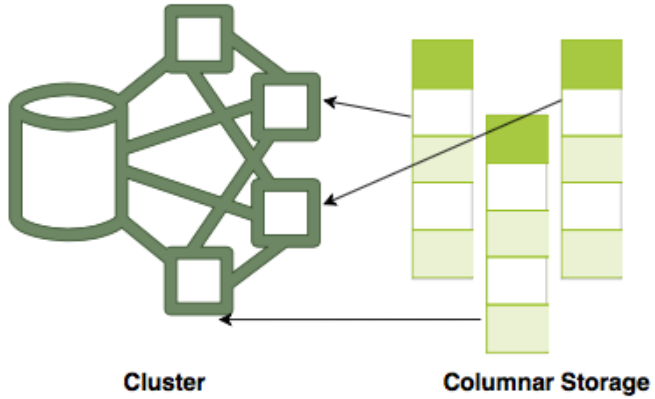
- **OLAP**: Applications allowing users to **analyze/query petabytes of data**
  - **Examples**: Reporting applications, Data warehouses, Business intelligence applications, Analytics systems
  - Data is consolidated from multiple (typically transactional) databases
  - **Sample application** : Decide insurance premiums analyzing data from last hundred years
- **Cloud** Managed Services:
  - Azure Synapse Analytics
  - Amazon Redshift
  - BigQuery (Google Cloud)
- Manage petabytes of data and run queries efficiently

Relational Databases - OLAP vs OLTP

In28 Minutes

- OLAP and OLTP use **similar data structures**
- BUT **very different approach in how data is stored**
- **OLTP databases** use row storage
  - Each table row is stored together
  - Efficient for processing small transactions
- **OLAP databases** use columnar storage
  - Each table column is stored together
  - **High compression** - store petabytes of data efficiently
  - **Distribute data** - one table in multiple cluster nodes
  - **Execute single query across multiple nodes** - Complex queries can be executed efficiently





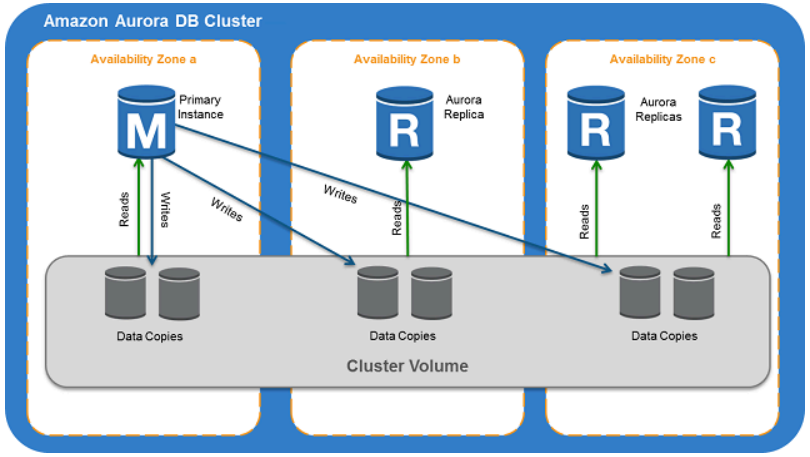
Amazon RDS (Relational Database Service)

In 28 Minutes

- **Managed relational database service** for OLTP use cases
  - Manages setup, backup, scaling, replication and patching
  - Supports PostgreSQL, MySQL (InnoDB storage engine full supported), MariaDB (Enhanced MySQL), Oracle Database and Microsoft SQL Server
  - **Features**:
    - Multi-AZ deployment (standby in another AZ)
    - Read replicas (Same AZ or Multi AZ (Availability+) or Cross Region(Availability++))
    - Storage auto scaling (up to a configured limit)
    - Automated backups (restore to point in time)
    - Manual snapshots
  - **Responsibilities**:
    - AWS is responsible for:
      - Availability, Scaling (based on your cnfgn.), Durability, Maintenance (patches) and Backups
    - You are responsible for:
      - Managing database users
      - App optimization (tables, indexes etc)

Amazon Aurora

- **MySQL and PostgreSQL**-compatible
- **2 copies** of data each in a **minimum of 3 AZ**
- Up to 15 read replicas
- Provides **"Global Database"** option
  - Up to five read-only, secondary AWS Regions
    - Low latency for global reads
    - Safe from region-wide outages
  - Minimal lag time, typically less than 1 second
- Deployment Options
  - Single master (One writer and multiple readers)
  - Multi master deployment (multiple writers)
  - Serverless
- Uses **cluster volume** (multi AZ storage)



https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGu

RDS & Aurora - Scaling

In28 Minutes

- **Vertical Scaling**: Change DB instance type and scale storage
  - Storage and compute changes are applied during maintenance window
  - You can also choose to “apply-immediately”
  - RDS would take care of data migration
    - Takes a few minutes to complete
  - Scaling limits:
    - MySQL, MariaDB, Oracle, and PostgreSQL engines - 64 TB
    - SQL Server - 16 TB
    - Aurora - 128 TB
  - RDS also supports storage auto scaling
- **Horizontal Scaling**
  - Configure Read Replicas
  - **Aurora**: Single master - One writer & multiple readers, Multi master deployment

Amazon RDS

Amazon RDS - When to use?

In28 Minutes

- **Use Amazon RDS for transactional applications** needing
  - Pre-defined schema
  - Strong transactional capabilities
  - Complex queries
- Amazon RDS is **NOT recommended** when
  - You need highly scalable massive read/write operations - for example millions of writes/second
    - Go for DynamoDB
  - When you want to upload files using simple GET/PUT REST API
    - Go for Amazon S3
  - When you need heavy customizations for your database or need access to underlying EC2 instances

Amazon RDS

Amazon Redshift

In28 Minutes

- **Amazon RDS** - Relational DB for OLTP (reads and writes)
- **Amazon Redshift** - Relational DB for OLAP (reads >>>> writes)
  - **Petabyte-scale distributed data ware house** based on PostgreSQL
    - Traditional ETL(Extract, Transform, Load), OLAP and Business Intelligence (BI) use cases
    - Integrates with data loading, reporting, mining and analytics tools
    - Supports SQL, Tables and Relationships
    - Columnar data storage resulting in High data compression
    - Automated replication (3 copies) and backups (to S3. Default retention - 1 day. Max - 35 days).
  - Massively parallel processing (MPP) - Create cluster and split storage and execution of queries across multiple nodes
    - Start with a single node and scale to multi nodes (Dynamically add and remove nodes)
    - Automatic recovery from any node failures
  - Remember:
    - A single row of data might be stored across multiple nodes (Columnar Storage)
    - A query to Redshift leader node is distributed to multiple compute nodes for execution

Semi Structured Data

In28 Minutes

- Data has **some structure BUT not very strict**
- Semi Structured Data is stored in NoSQL databases
  - NoSQL = not only SQL
  - Flexible schema
    - Structure data **the way your application needs it**
    - Let the structure evolve with time
  - Horizontally scale to petabytes of data with millions of TPS
- **Types of Semi Structured Data:**
  - Document
  - Key Value
  - Graph
  - Column Family

```
{
  "customerId": "99999999",
  "firstName": "Ranga",
  "lastName": "Ranga",
  "address": {
    "number": "505",
    "street": "Main Street",
    "city": "Hyderabad",
    "state": "Telangana"
  },
  "socialProfiles": [
    {
      "name": "twitter",
      "username": "@in28minutes"
    },
    {
      "name": "linkedin",
      "username": "rangaraokaranam"
    }
  ]
}
```

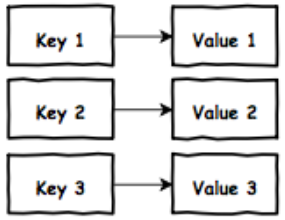
Semi Structured Data - 1 - Document

In28 Minutes

- Data stored as **collection of documents**
  - Typically **JSON** (Javascript Object Notation)
    - Be careful with formatting (name/value pairs, commas etc)
    - address - Child Object - {}, socialProfiles - Array - []
  - Documents are retrieved by unique id (called the key)
    - Typically, you can define additional indexes
  - Documents don't need to have the same structure
    - No strict schema defined on database
    - Apps should handle variations (application defined schema)
  - Typically, information in one document would be stored in multiple tables, if you were using a relational database
- **Use cases**: Product Catalog, Profile, Shopping Cart etc
- **Managed Services**: Azure Cosmos DB SQL & MongoDB API, Amazon DynamoDB, Google Cloud Datastore

```
{
  "customerId": "99999999",
  "firstName": "Ranga",
  "lastName": "Ranga",
  "address": {
    "number": "505",
    "street": "Main Street",
    "city": "Hyderabad",
    "state": "Telangana"
  },
  "socialProfiles": [
    {
      "name": "twitter",
      "username": "@in28minutes"
    },
    {
      "name": "linkedin",
      "username": "rangaraokaranam"
    }
  ]
}
```

Semi Structured Data - 2 - Key-Value



Key Value Database

userId	session
user1	{ "name": "Jane", "previousAction" : "someAction1" }
user2	{ "name": "Doe", "previousAction" : "someAction2" }
user3	{ "name": "Doe", "previousAction" : "someAction3" }

- Similar to a **HashMap**
  - **Key** - Unique identifier to retrieve a specific value
  - **Value** - Number or a String or a complex object, like a JSON file
  - Supports simple lookups - query by keys
    - NOT optimized for query by values
    - Typically, no other indexes allowed
  - **Use cases**: Session Store, Caching Data
- **Managed Services**: Azure Cosmos DB Table API, Amazon DynamoDB, Google Cloud Datastore

Semi Structured Data - 3 - Graph

- Social media applications have data with complex relationships
- How do you store such data?
  - As a graph in **Graph** Databases
  - Used to store data with **complex relationships**
- Contains nodes and edges (relationships)
- **Use cases**: People and relationships, Organizational charts, Fraud Detection
- **Managed Service**: Azure Cosmos DB Gremlin API, Amazon Neptune

Semi Structured Data - 4 - Column Family

In28 Minutes

ID	ColumnFamily:Identity	ColumnFamily:ContactInfo
001	**First Name**: Ranga **Last Name**: Karanam	**Phone**: ABC-DEF-GHI
002	**First Name**: Sathish	**Phone**: ABC-DEF-GHI
003	**First Name**: Ravisankar **Last Name**: Munusamy **Title**: Sr	**Phone**: ABC-DEF-GHI **Email**: SPAM@SPAM.COM

- Data organized into **rows and columns**
- Can appear similar to a relational database
- **IMPORTANT FEATURE:** Columns are divided into groups called column-family
  - Rows can be sparse (does NOT need to have value for every column)
- **Use cases**: IOT streams and real time analytics, financial data - transaction histories, stock prices etc
- **Managed Service**: Azure Cosmos DB Cassandra API, Google Cloud Bigtable,

Amazon DynamoDB

In28 Minutes

- Fast, scalable, **distributed** for any scale
- Flexible **NoSQL** Key-value & document database (schemaless)
- **Single-digit millisecond** responses for **million of TPS**
- Do not worry about scaling, availability or durability
  - Automatically partitions data as it grows
  - Maintains 3 replicas within the same region
- No need to provision a database
  - Create a table and configure read and write capacity (RCU and WCU)
  - Automatically scales to meet your RCU and WCU
- Provides an **expensive serverless mode**
- **Use cases**: User profiles, shopping carts, high volume read write applications

DynamoDB

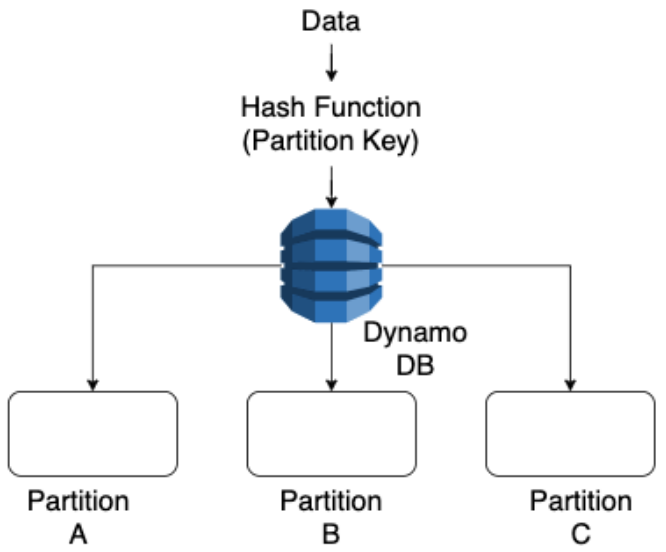
DynamoDB Tables

- **Hierarchy** : Table > item(s) > attribute (key value pair)
- **Mandatory** primary key
- Other than the primary key, tables are **schemaless**
  - No need to define the other attributes or types
  - Each item in a table can have distinct attributes
- Max 400 KB per item in table
- DynamoDB Tables are region specific.
  - If your users are in multiple regions, mark the table as **Global Table**
  - Replicas are created in selected regions


```
{
    "id": 1,
    "name": "Jane Doe",
    "username": "abcdefgh",
    "email": "someone@gmail.com",
    "address": {
      "street": "Some Street",
      "suite": "Apt. 556",
      "city": "Hyderabad",
      "zipcode": "500018",
      "geo": {
        "lat": "-3.31",
        "lng": "8.14"
      }
    },
    "phone": "9-999-999-9999",
    "website": "in28minutes.com",
    "company": {
      "name": "in28minutes"
    }
}
```

DynamoDB - Primary Key

- Two Types
  - **Simple**: Partition Key (or Hash)
  - **Composite**: Partition Key + Sort Key (or Hash + Range)
- Primary key should be **unique** (Cannot be changed later)
- Partition key **decides the partition** (input to hash function)
  - Same partition key items stored together (sorted by sort key, if it exists)
  - Choose a partition key that helps you to distribute items evenly across partitions
    - Prefer High Cardinality (column with many possible values)
    - Append a Random Value (if needed)



DynamoDB - Secondary Indexes

In 28 Minutes

- **Local secondary index**: Same partition key as primary key (diff. sort key)
  - Example:
    - Attributes: CustomerId (partition key) + OrderId (sort key), OrderCreationDt, ProductDetails, OrderStatus
    - LSI: [CustomerId (partition key) + OrderCreationDt] OR [CustomerId (partition key) + OrderStatus]
  - Defined at table creation (Modifications NOT allowed)
- **Global secondary index**: Partition and sort key CAN be diff. from primary key
  - Example:
    - Attributes: storeCode (Primary Key), city, state, country, address
    - GSI: state or city or country
  - Can be added, modified & removed later
  - Stored separately (Separate RCU and WCU) from the main table
  - (Recommended) Project fewer attributes to save cost
  - (Recommended) Avoid throttling on main table - Assign RCU and WCU at least equal to main table

DynamoDB Consistency Levels

- (DEFAULT) **Eventually Consistent Reads** : Might NOT get latest data
  - If tried after few seconds, you will get the latest data
- **Strongly Consistent Reads**: Get the most up-to-date data
  - Reflects updates from all the previous successful write operations
  - Set `ConsistentRead` to true
  - Disadvantages:
    - Returns 500 error in case of network delay
    - May have higher latency
    - Not supported on Global Secondary Indexes
    - Uses more throughput capacity units
- **Supports transactions** (TransactWriteItems, TransactGetItems)
  - All-or-nothing changes to multiple items both within and across tables
    - Include PutItem, UpdateItem and DeleteItem operations
  - More expensive

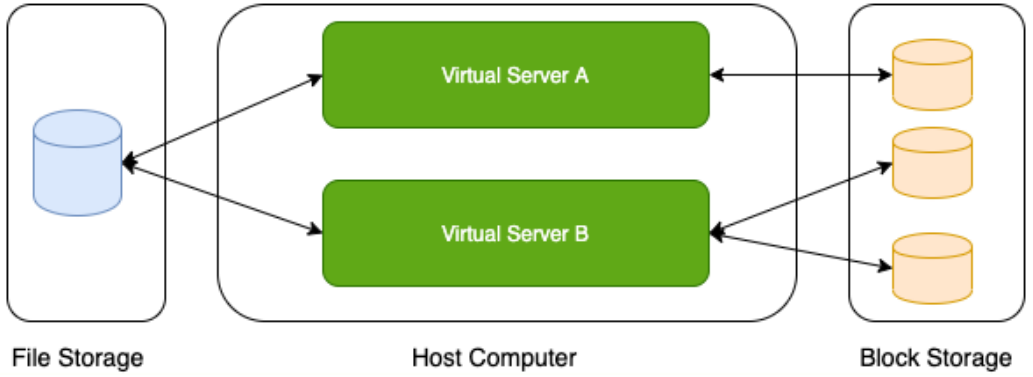
DynamoDB Read/Write Capacity Modes

In 28 Minutes

- **Provisioned**: Provision read (RCU) and write (WCU) capacity needed per second
  - Dynamically adjustable (Unused capacity can be used in bursts)
  - Billed for the provisioned capacity whether its used or not
- **On Demand**: Truly serverless and expensive
  - For unknown workloads or traffic with huge spikes
  - Use On Demand only when:
    - Workloads are really spiky causing low utilization of Provisioned Capacity OR
    - Usage is very low (for example, in test environments) making manual adjustments expensive

DynamoDB

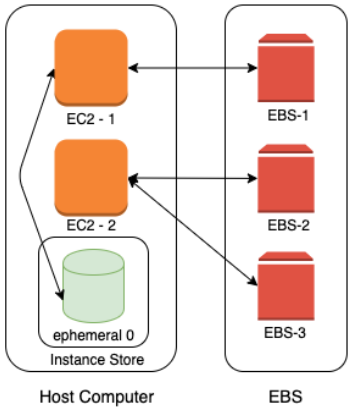
Unstructured Data



- Data which does not have any structure (Audio files, Video files, Binary files)
  - What is the type of storage of your hard disk?
    - **Block Storage** (Azure Disks, Amazon EBS, Google Cloud Persistent Disk)
  - You've created a file share to share a set of files with your colleagues in a enterprise. What type of storage are you using?
    - **File Storage** (Azure Files, Amazon EFS, Google Cloud Filestore)
  - You want to be able to upload/download objects using a REST API without mounting them onto your VM. What type of storage are you using?
    - **Object Storage** (Azure Blob Storage, Amazon S3, Google Cloud Storage)

Block Storage - Types

- **Instance Store**: Physically attached to EC2 instance
  - Ephemeral storage - Temporary data (Data lost - hardware fails or instance termination)
  - **CANNOT take a snapshot** or restore from snapshot
  - **Use case**: cache or scratch files
- **Elastic Block Store (EBS)**: Network Storage
  - More Durable. Very flexible **Provisioned capacity**
  - **Increase size when you need it** - when attached to EC2 instance
  - *99.999% Availability* & replicated within the same AZ
  - **Use case** : Run your custom database



Amazon EBS vs Instance Store

In28 Minutes

Feature	Elastic Block Store (EBS)	Instance Store
Attachment to EC2 instance	As a network drive	Physically attached
Lifecycle	Separate from EC2 instance	Tied with EC2 instance
Cost	Depends on provisioned size	Zero (Included in EC2 instance cost)
Flexibility	Increase size	Fixed size
I/O Speed	Lower (network latency)	2-100X of EBS
Snapshots	Supported	Not Supported
Use case	Permanent storage	Ephemeral storage
Boot up time	Low	High

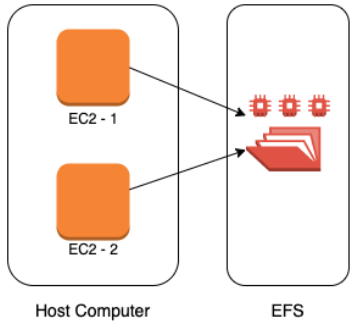
Amazon Elastic Block Store (EBS) Types

In 28 Minutes

- **General Purpose SSD** (gp2) ($$$): **Balance price & performance**
  - I/O performance **increases with size**
  - **Burst** up to 3,000 IOPS above the baseline
    - **Use cases** : Transactional workloads (Cost sensitive) & boot volumes
      - Small/medium databases, dev/test environments,
- **Provisioned IOPS SSD** (io1) ($$$$): Provision IOPS you need
  - Delivers consistent performance for **random and sequential** access
  - Designed for **low latency transactional** workloads
    - **Use cases** : large relational or NoSQL databases
- **Throughput Optimized HDD** (st1) ($$): High Throughput
  - For **frequently accessed, throughput-intensive sequential** workloads
    - **Use cases** : MapReduce, Kafka, log processing, data warehouse, and ETL
- **Cold HDD** (sc1) ($): Lowest Cost
  - **Use cases** : infrequent data access - very low transaction databases

Amazon EFS

- **Petabyte scale, Auto scaling, Pay for use** shared file storage
- Compatible with Amazon EC2 Linux-based instances
- **(Usecases)** Home directories, file share, content management
- (Alternative) Amazon FSx for Lustre
  - File system **optimized for performance**
  - High performance computing (HPC) and media processing use cases
  - Automatic encryption at-rest and in-transit
- (Alternative) Amazon FSx Windows File Servers
  - Fully managed Windows file servers
  - Accessible from Windows, Linux and MacOS instances
  - Integrates with Microsoft Active Directory (AD) to support Windows-based environments and enterprises.
  - Automatic encryption at-rest and in-transit



Amazon S3 (Simple Storage Service)

In 28 Minutes

- **Most popular, very flexible & inexpensive** storage service
- Store objects using a **key-value** approach (objects in buckets)
- Provides REST API to access and modify objects
- Provides **unlimited storage**:
  - (S3 storage class) **99.99% availability** & **(11 9's - 99.999999999) durability**
  - Objects are **replicated in a single region (across multiple AZs)**
- **Store all file types** - text, binary, backup & archives:
  - Media files and archives, Application packages and logs
  - Backups of your databases or storage devices
  - Staging data during on-premise to cloud database migration
- Supports Versioning(**Optional** - Enabled at bucket level):
  - Protects against **accidental deletion**

Amazon S3

Amazon S3 Storage Classes - Introduction

In 28 Minutes

- **Different kinds of data** can be stored in Amazon S3
  - Media files and archives
  - Application packages and logs
  - Backups of your databases or storage devices
  - Long term archives
- Huge variations in **access patterns**
- **Trade-off** between access time and cost
- **S3 storage classes** help to optimize your costs while meeting access time needs
  - Designed for durability of **99.999999999%(11 9's)**

Amazon S3

Amazon Glacier

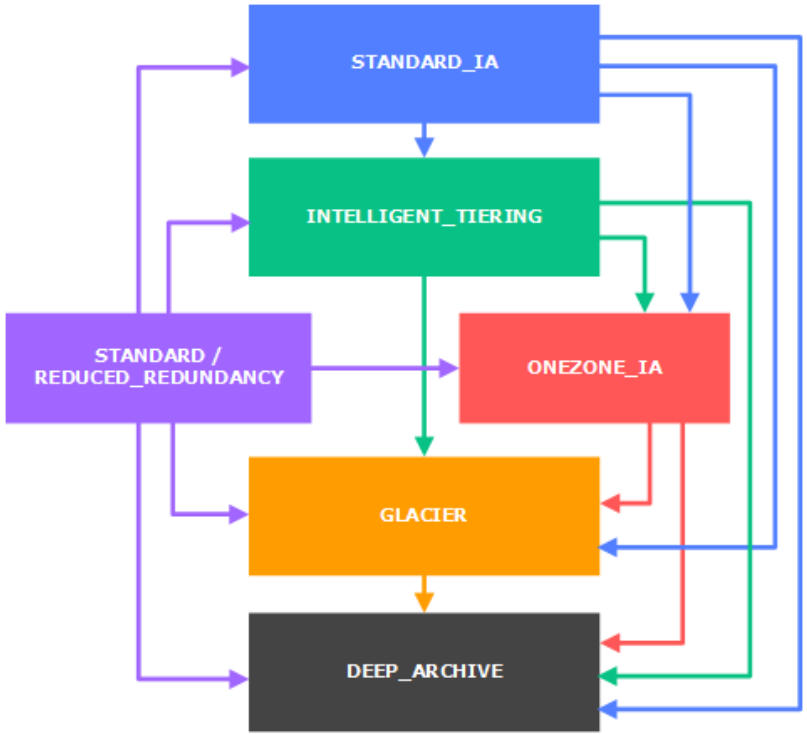
Amazon S3 Storage Classes

In28 Minutes

Storage Class	Scenario	AZs
Standard	Frequently accessed data	>=3
Standard-IA	Long-lived, infrequently accessed data (backups for disaster recovery)	>=3
One Zone-IA	Long-lived, infrequently accessed, non-critical data (Easily re-creatable data - thumbnails for images)	1
Intelligent-Tiering	Long-lived data with changing or unknown access patterns	>=3
Glacier	Archive data with retrieval times ranging from minutes to hours	>=3
Glacier Deep Archive	Archive data that rarely, if ever, needs to be accessed with retrieval times in hours	>=3
Reduced Redundancy (Not recommended)	Frequently accessed, non-critical data	>=3

S3 Lifecycle configuration

- Files are frequently accessed when they are created
- Generally **usage reduces with time**
- How do you save costs and **move files automatically between storage classes**?
  - Solution: **S3 Lifecycle configuration**
- **Two kinds of actions**:
  - **transition** actions (one storage class to another)
  - **expiration** actions (delete objects)
- Object can be identified by tags or prefix.



https://docs.aws.amazon.com/AmazonS3/latest/dev/lifecy
transition-general-considerations.html

Amazon S3 - Important Features

In 28 Minutes

- **Static Website Hosting**: Use S3 to host a static website using a bucket
  - Step 1 : Upload website content
  - Step 2 : Enable **Static website hosting**
  - Step 3 : Disable "Block public access"
  - Step 4 : Configure "Bucket policy" to enable public read access
- **Event Notifications**: Configure **notifications** when **certain events** happen
  - **Event Sources**: New object created events, Object removal events , Reduced Redundancy Storage (RRS) object lost events, Replication events etc.
  - **Event Destinations**: Amazon SNS topic, Amazon SQS queue, AWS Lambda function etc.
- **Presigned URL** : Grant **time-limited permission** (few hours to 7 days) to download objects
  - **Avoid** web site scraping and unintended access

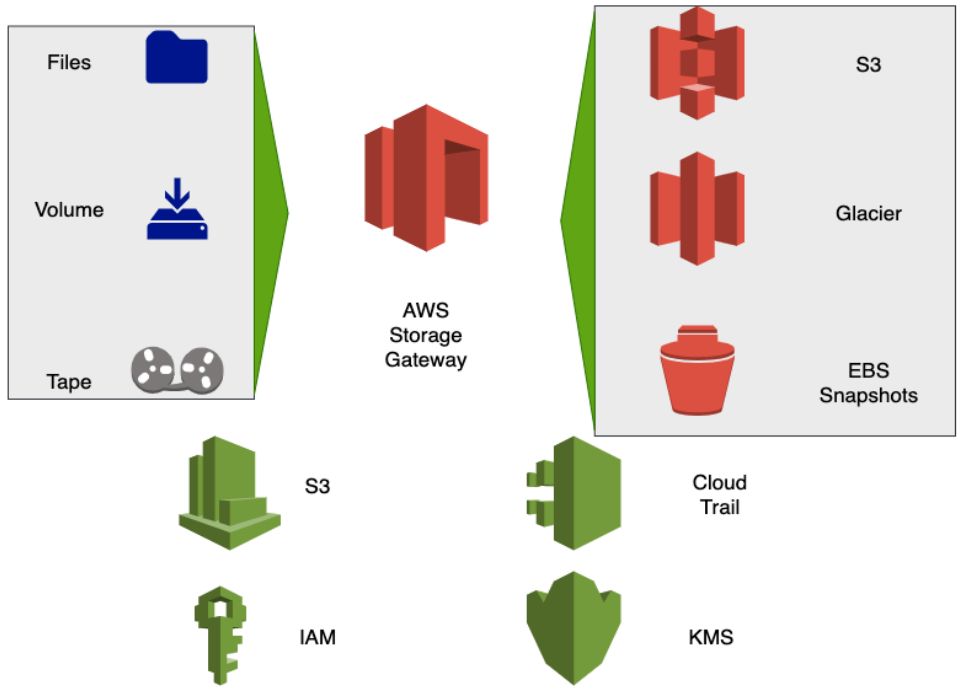
Amazon S3 Glacier

In28 Minutes

- In addition to existing as a S3 Storage Class, S3 Glacier is a separate AWS Service on it own!
- **Extremely low cost storage** for archives and long-term backups:
  - Old media content
  - Archives to meet regulatory requirements (old patient records etc)
  - As a replacement for magnetic tapes
- High durability (11 9s - 99.999999999%)
- High scalability (unlimited storage)
- High security (**encrypted** at rest and in transfer)

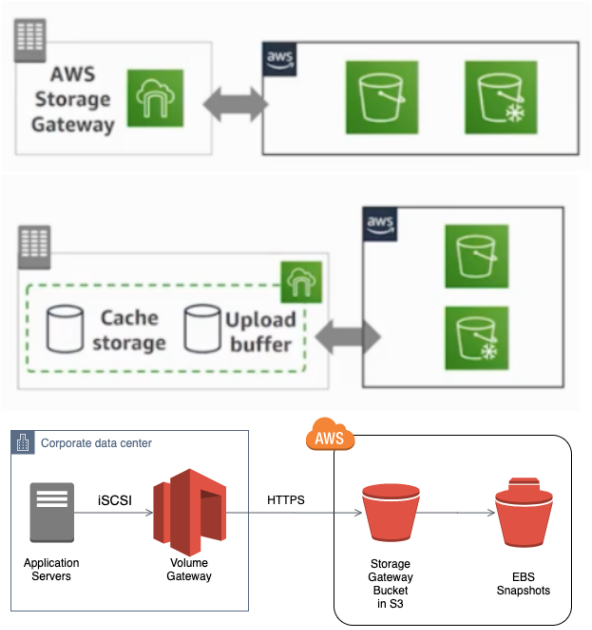
AWS Storage Gateway

- **Hybrid storage**: cloud + on premise
- Unlimited cloud storage for on-premise software applications and users with good performance
- (Remember) Storage Gateway and S3 Glacier **encrypt data** by default
- **Three Options**
  - AWS Storage File Gateway
  - AWS Storage Tape Gateway
  - AWS Storage Volume Gateway



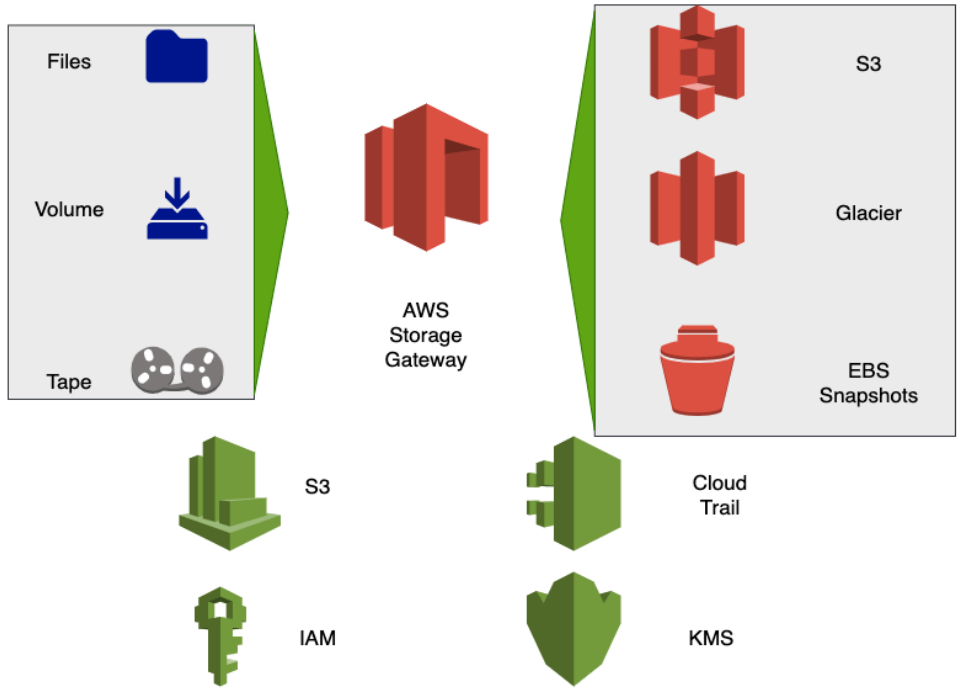
AWS Storage Gateway - Types

- **Storage File Gateway** - Storage for file shares
  - Files **stored** in **Amazon S3 & Glacier**
- **Storage Tape Gateway** - Virtual tape backups
  - Tapes stored in Amazon S3 & Glacier
    - Avoid complex physical tape backups (wear and tear)
    - **No change needed** for tape backup infrastructure
- **Storage Volume Gateway** : Cloud Block Storage
  - Use cases: Backup, Disaster Recovery, Cloud Migration
  - (Option 1) **Cached** (Gateway Cached Volumes):
    - Primary Data Store - **AWS - Amazon S3**
    - **On-premise cache** stores frequently accessed data
  - (Option 2) **Stored** (Gateway Stored Volumes):
    - Primary Data Store - **On-Premises**
    - Asynchronous copy to AWS
    - Stored as EBS snapshots



AWS Storage Gateway - Review

- Key to look for : **Hybrid storage** (cloud + on premise)
- File share moved to cloud => **AWS Storage File Gateway**
- Tape Backups on cloud => **AWS Storage Tape Gateway**
- Volume Backups on cloud (Block Storage) => **AWS Storage Volume Gateway**
  - High performance => **Stored**
  - Otherwise => **Cached**



Storing Unstructured data in AWS - Review

In28 Minutes

Type	Description
Object	**Amazon S3** (Very Flexible) Store large objects using a key-value approach
Block	Storage connected to one EC2 instance. Your Hard Disks. **Elastic Block Store**(EBS - Permanent) **Instance Store** (Ephemeral)
File	File Share. Share storage between EC2 instances. **EFS** (Linux) **FSx Windows** **FSx for Lustre** (High Performance)
Archival	**Amazon Glacier** Extremely low cost storage for archives and long-term backups.
Hybrid	**AWS Storage Gateway** Cloud + On Premise

Identity and Access Management

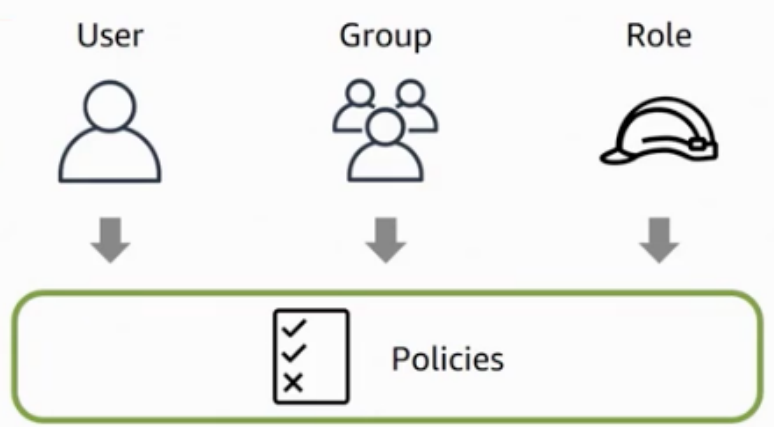
AWS Identity and Access Management (IAM)

In 28 Minutes

- **Authentication** (the right user?) and
- **Authorization** (the right access?)
- **Identities** can be
  - AWS users or
  - Federated users (externally authenticated users)
- Provides very **granular** control
  - Limit a single user:
    - to perform single action
    - on a specific AWS resource
    - from a specific IP address
    - during a specific time window

Important IAM Concepts

- **IAM users**: Users created in an AWS account
  - Has credentials attached (name/password or access keys)
- **IAM groups**: Collection of IAM users
- **Roles**: Temporary identities
  - Does NOT have credentials attached
  - (Advantage) Expire after a set period of time
- **Policies**: Define permissions
  - **AWS managed policies** - Standalone policy predefined by AWS
  - **Customer managed policies** - Standalone policy created by you
  - **Inline policies** - Directly embedded into a user, group or role



AWS IAM Policies - Authorization

In28 Minutes

```
{
 "Version": "2012-10-17",
 "Statement": [
           {
            "Effect": "Allow",
            "Action": "*", //["s3:Get*","s3:List*"],
            "Resource": "*" //"arn:aws:s3:::mybucket/somefolder/*"
         }
    ]
}
```

- Policy is a JSON document with one or more permissions
  - **Effect** - Allow or Deny
  - **Resource** - Which resource are you providing access to?
  - **Action** - What actions are allowed on the resource?
  - **Condition** - Are there any restrictions on IP address ranges or time intervals?
  - Example above: AWS Managed Policy : AdministratorAccess
  - Give Read Only Access to S3 buckets - `"Action": ["s3:Get*","s3:List*"]`

IAM Scenarios

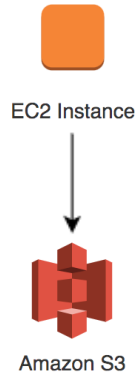
In28 Minutes

Scenario	User/Role	Recommendation
You're the only one in your account	IAM user	Do not use ROOT user
Your team needs access to your AWS account and there is no other identity mechanism	IAM users	Use IAM Groups to manage policies
EC2 instance talks with Amazon S3 or a database	IAM role	
Cross Account Access	IAM role	

IAM Role Use case 1 : EC2 talking with S3

In28 Minutes

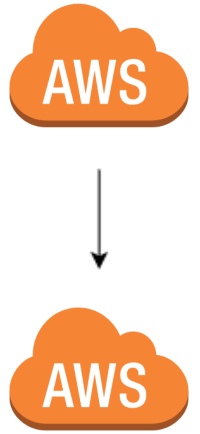
- **1:** Create IAM role with access to S3 bucket
- **2:** Assign IAM role to EC2 instance
- No need to store credentials in config files
- No need for rotation of keys
- **What happens in the background?**
  - **Instance Profile**: A Container (A Box) for an IAM role
    - Used to pass role information to an EC2 instance
    - Creation:
      - AWS Management Console:
        - An instance profile is automatically created when you create a role for EC2 instance
      - From CLI or API
        - Explicitly manage Instance Profiles - CreateInstanceProfile etc
  - (REMEMBER) Instance profile is a simple container for IAM Role



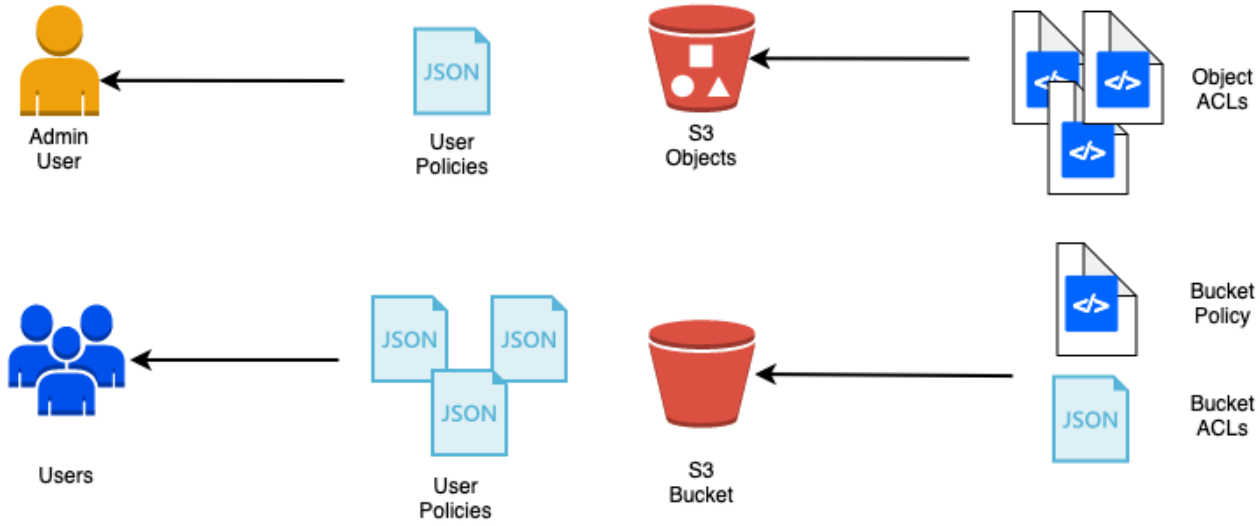
IAM Role Use case 2: Cross Account Access

In28 Minutes

- PROD Account (111111111111)
  - 1: Create IAM role (ProdS3AccessRole) with right permissions
  - 2: Establish trust relationship with DEV AWS account (222222222222)
- DEV Account (222222222222)
  - Grant users (Ops Group) permissions to assume ProdS3AccessRole in PROD Account
    - **1:** Create a customer managed policy ProdS3AccessPolicy allowing access to call STS AssumeRole API for ProdS3AccessRole(arn:aws:iam::111111111111:role/ProdS3AccessRole)
    - **2:** Assign the policy to users (Ops Group)
    - **(Optional) 3:** Enable MFA for assuming the role
- What happens when DEV AWS account Ops user requests access to Prod Account role?
  - 1: Operations user requests access to the role
  - 2: AWS STS AssumeRole API is called to check for access to ProdS3AccessRole role
  - 3: Operations user assumes the role



Identity-based and Resource-based policies



- By default only account owner has access to a S3 bucket
- Access policies enable other users to access S3 buckets and objects:
  - **Identity-based policies** : Attached to an IAM user, group, or role
  - **Resource-based policies and ACLs** : Attached to a resource - S3 buckets, Amazon SQS queues, and AWS KMS keys

Identity-based and Resource-based policies

In28 Minutes

Policy Type	Identity-based	Resource-based
Attached with	IAM user, group, or role	A resource
Type	Managed and Inline	Inline only
Focus	What resource? What actions?	Who? What actions?
Example	Can list S3 buckets with name **XYZ**	Account A can read and modify. Public can read.
Cross-account access	User should switch role	Simpler. User accesses resource directly from his AWS account
Supported by	All services	Subset of services - S3, SQS, SNS, KMS etc
Policy Conditions	When (dates), Where(CIDR blocks),Enforcing MFA	When(dates), Where(CIDR blocks), Is SSL Mandatory?

IAM Best Practices - Recommended by AWS

In28 Minutes

- **Users** – Create individual users
- **Groups** – Manage permissions with groups
- **Permissions** – Grant least privilege
- **Auditing** – Turn on AWS CloudTrail
- **Password** – Configure a strong password policy
- **MFA** – Enable MFA for privileged users
  - (Hardware device - Gemalto, Virtual device - An app on a smart phone)
- **Roles** – Use IAM roles for Amazon EC2 instances
- **Sharing** – Use IAM roles to share access
- **Rotate** – Rotate security credentials regularly
- **Root** – Reduce or remove use of root

AWS IAM

Data Encryption

KMS and Cloud HSM

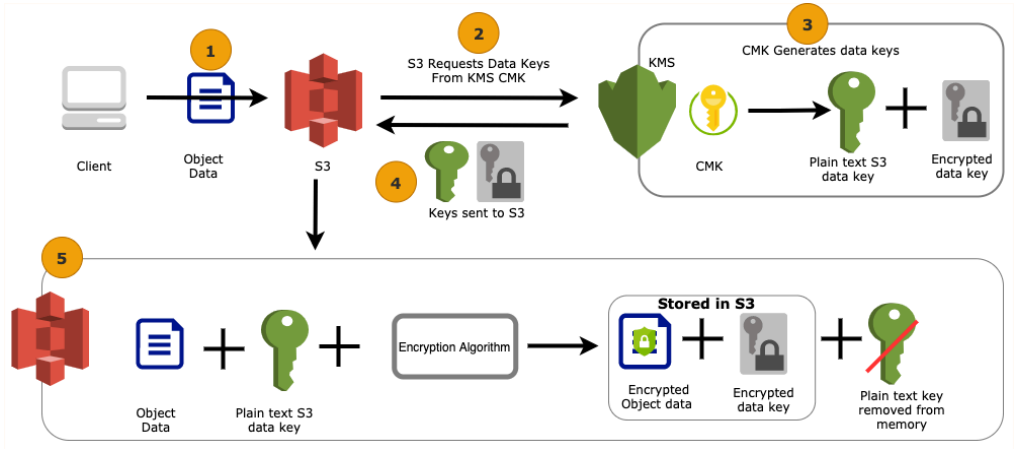
- Generate, store, use and replace your keys(symmetric & asymmetric)
- **KMS**: Multi-tenant Key Management Service
  - **KMS** integrates with all storage and database services in AWS
  - Define key usage permissions (including **cross account** access)
  - **Automatically rotate master keys** once a year
  - **Schedule key deletion** to verify if the key is used
    - Mandatory minimum wait period of 7 days (max-30 days)
- **CloudHSM**: **Dedicated single-tenant** HSM for regulatory compliance
  - AWS **CANNOT access your encryption master keys** in CloudHSM
    - **(Recommendation)** Be ultra safe with your keys. Use two or more HSMs in separate AZs.
    - AWS KMS can use CloudHSM cluster as **"custom key store"** to store the keys:
      - AWS Services can continue to talk to KMS for data encryption
      - (AND) KMS does the necessary integration with CloudHSM cluster
  - **Use Cases**: (Web servers) Offload SSL processing, Certificate Authority etc

AWS KMS

Cloud HSM

KMS - How does encryption and decryption happen?

- **Customer Master Key** (CMK) created in KMS and mapped to S3
- **Encryption Steps**:
  - Data sent to S3
  - S3 receives **data keys** from KMS
  - S3 encrypts data
  - Stores encrypted data & data key
- **Decryption Steps**:
  - S3 sends **encrypted data key** to KMS
  - KMS decrypts using CMK. Returns data key.
  - S3 uses plain text data key to decrypt data
  - Remove data key from memory asap
- Also called **Envelope Encryption**



Customer master keys (CMKs)

In28 Minutes

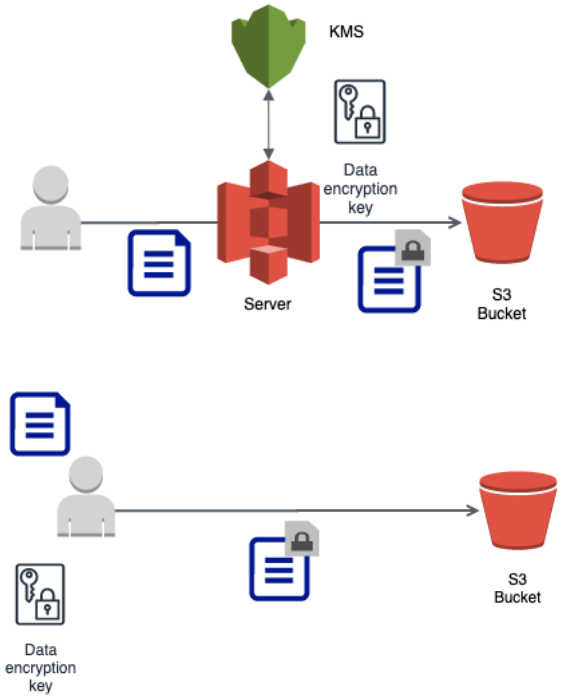
- CMKs are used for encryption, decryption and signing
- 3 Types of CMKs:
  - **Customer managed**: Owned and Managed by Customer
    - Used only for your AWS account
  - **AWS managed**: Managed by AWS on your behalf
    - Used only for your AWS account
  - **AWS owned**: AWS owns and manages them
    - Used in multiple AWS accounts.
    - LIMITED usecases
- **Most Services** support both AWS managed and Customer managed Keys
  - Amazon S3, Amazon DynamoDB, Amazon EBS, Amazon SQS, Amazon SNS
- Few Services support only AWS managed keys
  - Amazon DynamoDB Accelerator (DAX), AWS CodeCommit

AWS KMS

Server Side vs Client Side Encryption - Amazon S3

In28 Minutes

- **Server Side Encryption**: S3 <-> KMS to encrypt data
  - **SSE-S3**: AWS S3 manages its own keys (rotated every month)
    - Request Header - *x-amz-server-side-encryption(AES256)*
  - **SSE-KMS**: Customer manages keys in KMS
    - Request Headers - *x-amz-server-side-encryption(aws:kms)* and *x-amz-server-side-encryption-aws-kms-key-id(ARN for key in KMS)*
  - **SSE-C**: Customer sends key with request (HTTPS mandatory)
    - S3 performs encryption and decryption without storing the key
  - **Use HTTPS endpoints** (secure data in transit)
    - All AWS services (including S3) provides HTTPS endpoints
- **Client Side Encryption**: Client **manages encryption**
  - Client sends encrypted data to AWS service
    - AWS will not be aware of master key or data key
    - AWS service stores data as is
  - Use a client library (Amazon S3 Encryption Client)



KMS with S3 - Use cases

Key Storage	Encryption Location	Requirement	Recommendation
Customer	in S3	You want to manage the keys (including rotation) outside AWS	SSE with Customer-Provided Keys (SSE-C)
KMS	in S3	Easy Management of Keys. Auditing.	SSE with Customer Master Keys (SSE-KMS)
KMS	in S3	You want Encryption but Don't want Management	SSE with Amazon S3-Managed Keys (SSE-S3)
Customer (master key stored within your app)	On Premises		CSE (Amazon S3 encryption client)
KMS	On Premises		CSE (Amazon S3 encryption client)

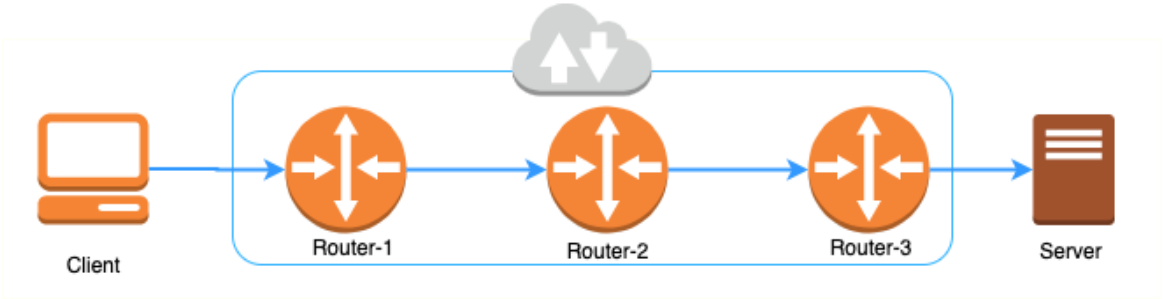
Virtual Private Cloud (VPC)

Amazon VPC (Virtual Private Cloud) and Subnets

In 28 Minutes

- VPC (Virtual Private Cloud) - Your **own isolated network** in AWS cloud
  - Network traffic within a VPC is isolated (not visible) from all other Amazon VPCs
  - You **control all the traffic** coming in and going outside a VPC
  - **(Best Practice)** Create AWS resources **in a VPC**
    - Secure resources from unauthorized access AND
    - Enable secure communication between your cloud resources
    - Each VPC is created in a Region
- **Subnet - Separate public resources from private resources** in a VPC
  - **Create different subnets** for public and private resources
    - Resources in a public subnet **CAN** be accessed from internet
    - Resources in a private subnet **CANNOT** be accessed from internet
    - BUT resources in public subnet can talk to resources in private subnet
    - Each Subnet is created in an Availability Zone
    - VPC - us-east-1 => Subnets - AZs us-east-1a or us-east-1b or ..

Routing on the internet



- You have an IP address of a website you want to visit
- There is **no direct connection** from your computer to the website
- Internet is actually a **set of routers** routing traffic
- Each router has a set of rules that help it decide the path to the destination IP address

Routing inside AWS

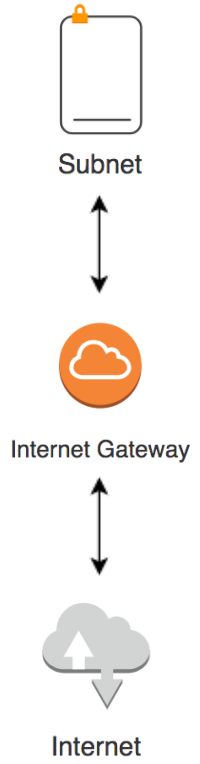
In28 Minutes

Destination	Target
172.31.0.0/16	Local
0.0.0.0/0	igw-1234567

- Route tables associated with VPCs and Subnets are used for Routing
- Route Tables have Routes - **set of rules**
  - Each route or routing rule has a **destination (CIDR Block - range of addresses) and target**
  - **Rule 1** - Route requests to 172.31.0.0/16 (172.31.0.0 to 172.31.255.255) to local resources
  - **Rule 2** - Route all other IP addresses (0.0.0.0/0) to internet (internet gateway)
  - What happens if I search for an address **172.31.0.10**?
    - Two matches - 172.31.0.0/16 (172.31.0.0 to 172.31.255.255) and 0.0.0.0/0
    - **Most specific rule wins** 172.31.0.0/16 is more specific. **Result** : Routing to a local resource
  - What happens if I search for an address **69.209.0.10**?
    - One destination match - 69.208.0.10. **Result** : Routing to internet

Public Subnet vs Private Subnet

Name	Destination	Target	Explanation
RULE 1	172.31.0.0/16	Local	Local routing
RULE 2	0.0.0.0/0	igw-1234567	Internet routing

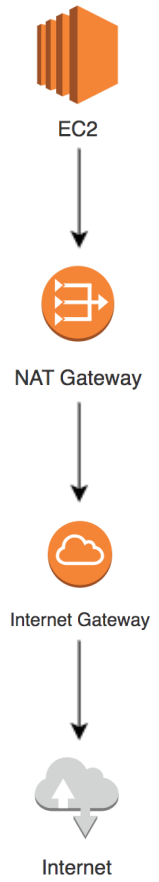


- **Public Subnet**: Communication allowed - Subnet <-> Internet
- An **Internet Gateway** enables internet communication for subnets
  - **Public Subnet**: Subnet having a route to an internet gateway
  - **Private Subnet**: Subnet **DOES NOT** have route to an internet gateway
- **Internet Gateway** sits between subnet (VPC) resources and internet
  - **One to one mapping** with a VPC
  - Supports IPv4 and IPv6
  - Translate private IP address to public IP address and vice-versa

Private Subnet - NAT Devices - Download Patches

In28 Minutes

- Allow internet access from private subnet using NAT Device:
  - **Allow instances in a private subnet to download software patches** while denying inbound traffic from internet
  - Allow instances in a private subnet to **connect privately to other AWS Services** outside the VPC
  - **Three Options**:
    - **NAT Instance**: Install an EC2 instance with specific NAT AMI and configure as a gateway
      - Created in public subnet with **public IP address or Elastic IP**
      - Assigned with Security Group allowing
        - Inbound - HTTP(80) HTTPS(443) from private subnet
        - Outbound - HTTP(80) & HTTPS(443) to internet (0.0.0.0/0)
    - **NAT Gateway**: Managed Service (PREFERRED - No maintenance, more availability & high bandwidth)
      - Created in **PUBLIC subnet** with Elastic IP Address
    - **Egress-Only Internet Gateways**: For IPv6 subnets (NAT Gateway supports **IPv4 ONLY**)
- Private Subnet Route Table should have a rule to direct all outbound traffic (0.0.0.0/0) to the NAT device



Network Access Control List

- Security groups control traffic to a **SPECIFIC resource** in a subnet
- NACL provides **stateless firewall** at subnet level
  - Stop traffic from **even entering the subnet**
- Each subnet **must** be associated with a NACL
  - **Default NACL** allows all inbound and outbound traffic.
  - **Custom created NACL** denies all inbound and outbound traffic by default.
  - Rules have a priority number.
    - Lower number => Higher priority.

Security Group vs NACL

Feature	Security Group	NACL
Level	Assigned to a specific instance(s)/resource(s)	Configured for a subnet. Applies to traffic to all instances in a subnet.
Rules	Allow rules only	Both allow and deny rules
State	Stateful. Return traffic is automatically allowed.	Stateless. You should explicitly allow return traffic.
Evaluation	Traffic allowed if there is a matching rule	Rules are prioritized. Matching rule with highest priority wins.

Private Communication with other VPCs/Resources

In28 Minutes

- **VPC Peering** - Connect VPCs from same or diff. AWS accounts (across regions)
  - Allows private communication between the connected VPCs
  - Peering uses a request/accept protocol (Owner of requesting VPC sends a request)
  - Peering is not transitive. Peer VPCs cannot have overlapping address ranges.
- **VPC Endpoint** - Securely connect your VPC to another service
  - Gateway endpoint: Securely connect to Amazon S3 and DynamoDB
    - Endpoint serves as a target in your route table for traffic
    - Provide access to endpoint (endpoint, identity and resource policies)
  - Interface endpoint: Securely connect to a selected list of AWS services (***https://docs.aws.amazon.com/vpc/latest/privatelink/aws-services-privatelink-support.html***)
    - Powered by PrivateLink (keeps network traffic within AWS network)
    - Needs a elastic network interface (ENI) (entry point for traffic)
  - (Avoid DDoS & MTM attacks) Traffic does NOT go thru internet
  - (Simple) Does NOT need Internet Gateway, VPN or NAT

VPC Flow Logs

In28 Minutes

- **What?**
  - Capture traffic going in and out of your VPC (network interfaces)
- **Why?**
  - Monitor network traffic
  - Troubleshoot connectivity issues (NACL and/or security groups misconfiguration)
- **Can be created for**
  - a VPC
  - a subnet
  - or a network interface (connecting to ELB, RDS, ElastiCache, Redshift etc)
- Publish logs to Amazon CloudWatch Logs or Amazon S3
- **Flow log records** contain ACCEPT or REJECT
  - Is traffic is permitted by security groups or network ACLs?

VPC Flow Logs

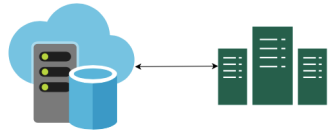
Troubleshoot using VPC Flow Logs

- **Inbound traffic rules** - NACL IN, SG IN, NACL OUT (SG OUT NOT checked)
  - If inbound request is rejected, SG or NACL could be mis-configured
  - If outbound response is rejected, NACL is mis-configured
- **Outbound traffic rules** - SG OUT, NACL OUT, NACL IN (SG IN NOT checked)
  - If outbound request is rejected, SG or NACL could be mis-configured
  - If inbound response is rejected, NACL is mis-configured
- Problem with response => Problem with NACL
- Problem with request could be problems with NACL or SG

Cloud Computing: Public vs Private vs Hybrid clouds

In 28 Minutes

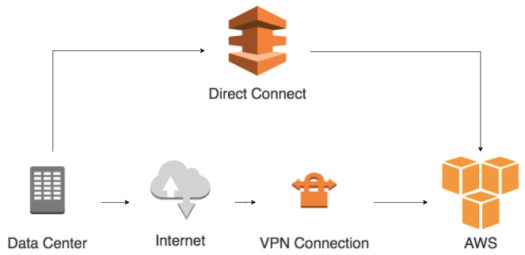
- Cloud Computing
  - **Public Cloud**
    - You host everything in the cloud (You DO NOT need a data center anymore)
      - No Capital Expenditure required
    - Hardware resources are owned by cloud platform
      - Hardware failures and security of the data center are managed by cloud platform
    - Summary: Hardware owned by cloud platform and shared between multiple tenants
      - Tenants: Customers who rent infrastructure (You, Me and other enterprises)
  - **Private Cloud**
    - You host everything in your own data center
      - Needs Capital Expenditure
      - Incur staffing and maintenance expenses for infrastructure
    - Delivers higher level of security and privacy
  - **Hybrid Cloud**:
    - Combination of both (Public & Private)
      - Use Public Cloud for some workloads and Private cloud for others
      - Example: Connecting an on-premise app to a cloud database
    - Provides you with flexibility: Go on-premises or cloud based on specific requirement



AWS and On-Premises - Overview

In28 Minutes

- **AWS Managed VPN**: Tunnels from VPC to on premises
  - Traffic over internet - encrypted using IPsec protocol
  - VPN gateway to connect one VPC to customer network
  - Customer gateway installed in customer network
    - You need a Internet-routable IP address of customer gateway
- **AWS Direct Connect (DX)**: Private dedicated network connection to on premises
  - (Advantage) Reduce your (ISP) bandwidth costs
  - (Advantage) Consistent Network performance (private network)
  - Connection options: Dedicated (1 Gbps or 10 Gbps) or Hosted (Shared 50Mbps to 10 Gbps)
  - (Caution) Establishing DC connection takes a month
  - (Caution) Establish a redundant DC for maximum reliability
  - (Caution) Data is NOT encrypted (Private Connection ONLY)



DevOps

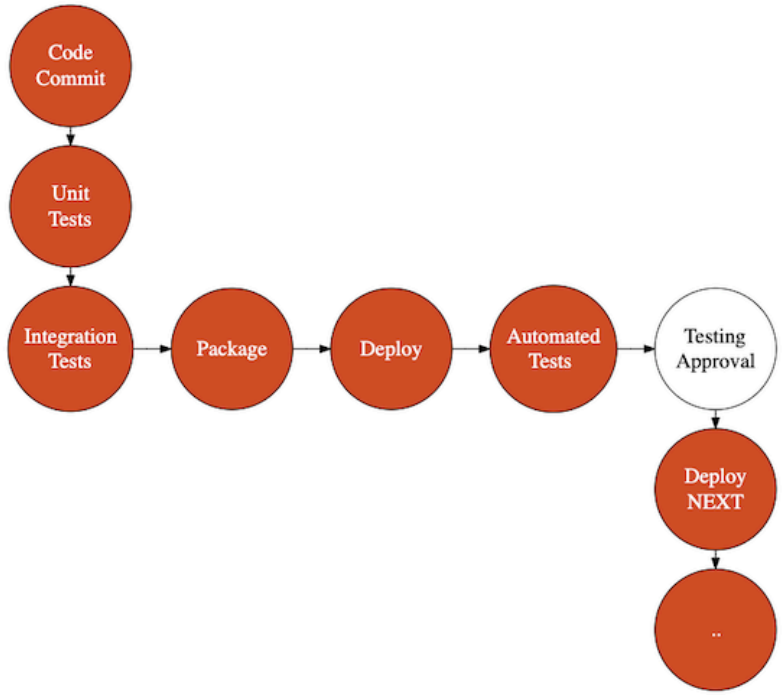
DevOps

In 28 Minutes

- Getting Better at "**Three Elements of Great Software Teams**"
  - **Communication** - Get teams together
  - **Feedback** - Earlier you find a problem, easier it is to fix
  - **Automation** - Automate testing, infrastructure provisioning, deployment, and monitoring
- **Practices**: CI/CD, IaaC, Ops (Monitoring, Tracing, ...)

DevOps - CI, CD

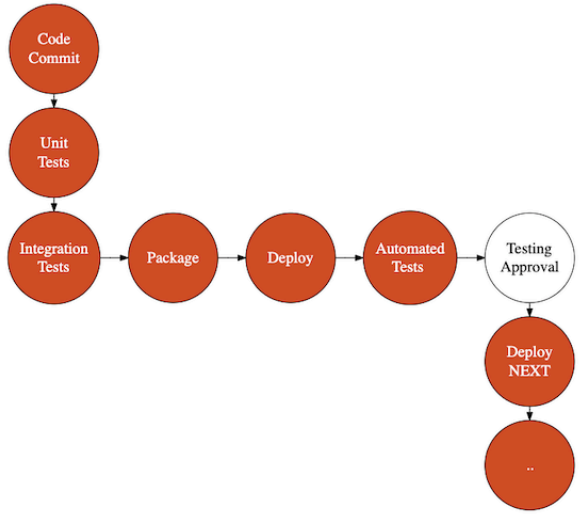
- **Continuous Integration**
  - Continuously run your tests and packaging
- **Continuous Delivery**
  - Continuously deploy to test environments
- **Continuous Deployment**
  - Continuously deploy to production



DevOps - CI, CD Tools in AWS

In28 Minutes

- **AWS CodeCommit**: Fully-featured, private Git repository
  - Similar to Github
- **AWS CodePipeline**: Orchestrate CI/CD pipelines
- **AWS CodeBuild** - Build and Test Code (packages and containers)
- **AWS CodeDeploy** - Automate Deployment(ECS, Lambda etc)
- **Amazon Elastic Container Registry (ECR)**: Store your Docker images



DevOps - CI CD - Recommended Things to Do

In28 Minutes

- **Static Code Analysis**
  - Lint, Sonar
  - Including Static Security Checks (Source Code Security Analyzer software like Veracode or Static Code Analyzer)
- **Runtime Checks**
  - Run Vulnerability Scanners (automated tools that scan web applications for security vulnerabilities)
- **Tests**
  - Unit Tests (JUnit, pytest, Jasmine etc)
  - Integration Tests (Selenium, Robot Framework, Cucumber etc)
  - System Tests (Selenium, Robot Framework, Cucumber etc)
  - Sanity and Regression Tests

DevOps - Infrastructure as Code

- Treat **infrastructure the same way as application code**
  - Track infrastructure **changes over time** (versioning)
  - Bring **repeatability** into your infrastructure
- Two Key Parts
  - **Infrastructure Provisioning**: Provision AWS resources
    - **Cloud neutral**: Terraform (templates), Pulumi (code)
    - **AWS**:
      - **AWS CloudFormation** - Provision Resources using JSON/YAML templates
      - **AWS Cloud Development Kit (AWS CDK)** - Provision resources using popular programming languages
      - **AWS Serverless Application Model (SAM)** - Provision Serverless Resources
  - **Configuration Management**: Install right software and tools on the provisioned resources
    - **Open Source Tools** - Chef, Puppet, Ansible and SaltStack
    - **AWS Service**: OpsWorks (Chef, Puppet in AWS)

AWS CloudFormation - Introduction

In28 Minutes

- Lets consider an example:
  - I would want to create a new VPC and a subnet
  - I want to provision a ELB, ASG with 5 EC2 instances & RDS database
  - I would want to setup the right security groups
- AND I would want to create 4 environments
  - Dev, QA, Stage and Production!
- CloudFormation can help you do all these with a simple (actually NOT so simple) script!
- Advantages (Infrastructure as Code - IAC & CloudFormation) :
  - Automate deployment of AWS resources in a controlled, predictable way
  - Avoid mistakes with manual configuration
  - Think of it as version control for your environments

CloudFormation

AWS CloudFormation

CloudFormation

- Free to use - Pay only for the resources provisioned
  - Get an automated estimate for your configuration
- **Template**: A JSON or YAML defining multiple resources
  - I want a VPC, a subnet, a database and ...
- CloudFormation understands dependencies
  - Creates VPCs first, then subnets and then the database
- (Default) Automatic rollbacks on errors (Easier to retry)
  - If creation of database fails, it would automatic delete the subnet and VPC
- Version control your template file (track changes over time)
- **Stack**: Group of resources created from CF template
- **Change Sets**:
  - To make changes to stack, update the template
  - Change set shows what would change if you execute (Verify and Execute)

AWS CloudFormation Templates - Examples

In 28 Minutes

JSON

```
{
    "Resources" : {
        "MyBucket" : {
            "Type" : "AWS::S3::Bucket"
            "Properties" : {
                "AccessControl" : "PublicRead"
            }
        }
    }
}
```

YAML

```
Resources:
  MyBucket:
    Type: AWS::S3::Bucket
    Properties:
        AccessControl: PublicRead
```

AWS CloudFormation - Important template elements

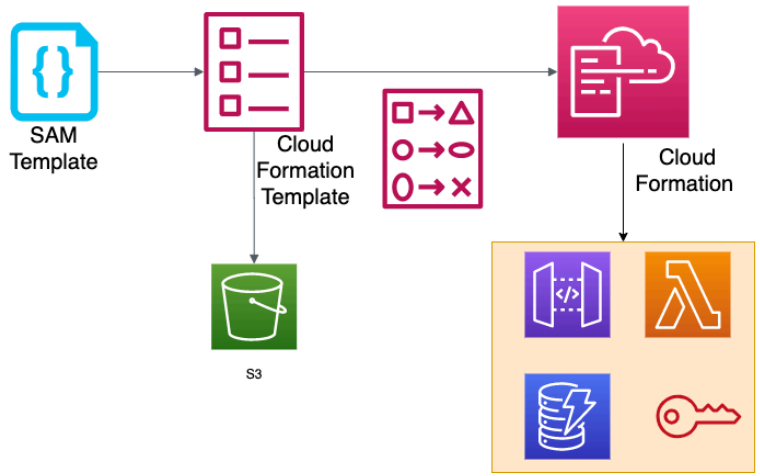
In 28 Minutes

```
{
  "AWSTemplateFormatVersion" : "version date",
  "Description" : "JSON string",
  "Metadata" : {},
  "Parameters" : {},
  "Mappings" : {},
  "Resources" : {},
  "Outputs" : {}
}
```

- **Resources** - What do you want to create?
  - One and only mandatory element
- **Parameters** - Values to pass to your template at runtime
  - Which EC2 instance to create? - ("t2.micro", "m1.small", "m1.large")
- **Mappings** - Key value pairs
  - Example: Configure different values for different regions
- **Outputs** - Return values from execution
  - See them on console and use in automation

Serverless Application Model

- **Serverless Application Model** - Open source framework for building serverless applications
  - Infrastructure as Code (IAC) for Serverless Applications
  - Integrate Serverless Best Practices
    - Tracing(X-Ray), CI/CD(CodeBuild,CodeDeploy,CodePipeline) etc
  - Define YAML file with resources
    - Functions, APIs, Databases..
    - (BEHIND THE SCENES) SAM config => CloudFormation
  - Benefits of SAM:
    - Simple deployment configuration
    - Extends CloudFormation & hides complexity
    - Built-in best practices
    - Local debugging and testing
    - Benefits of IAC(Infrastructure as Code)
      - No Manual Errors, Version Control, Avoid configuration drift



AWS SAM Template - Template Anatomy

In 28 Minutes

```
//Main indicator that it is a serverless template - Mandatory
Transform: AWS::Serverless-2016-10-31

Globals: //Global attributes used across resources
    //set of globals
Description:
    //String
Metadata:
    //template metadata
Parameters:
    //set of parameters
Mappings:
    //set of mappings
Conditions:
    //set of conditions
    //conditionally create resources for different environments
Resources: //Mandatory
    //AWS CloudFormation resources and AWS SAM resources
Outputs:
    //set of outputs
```

AWS SAM - Supported Resources

In28 Minutes

- Application (A container for all resources)
  - AWS::Serverless::Application
- Lambda Functions and Layers
  - AWS::Serverless::Function
  - AWS::Serverless::LayerVersion
- API Gateways
  - AWS::Serverless::Api
  - AWS::Serverless::HttpApi
- DynamoDB Tables
  - AWS::Serverless::SimpleTable
- Step Functions
  - AWS::Serverless::StateMachine
- For other resources, use AWS CloudFormation definition

Example SAM Template

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
Resources:
    CreateThumbnail:
        Type: AWS::Serverless::Function
        Properties:
            Handler: handler
            Runtime: runtime
            Timeout: 60
            Policies: AWSLambdaExecute
            Events:
                CreateThumbnailEvent:
                    Type: S3
                    Properties:
                        Bucket: !Ref SrcBucket
                        Events: s3:ObjectCreated:*
    SrcBucket:
        Type: AWS::S3::Bucket
```

AWS CDK

In 28 Minutes

```
// Create new VPC with 2 Subnets
const vpc = new ec2.Vpc(this, 'VPC', {
  natGateways: 0,
  subnetConfiguration: [{
    cidrMask: 24,
    name: "asterisk",
    subnetType: ec2.SubnetType.PUBLIC
  }]
});

const ec2Instance = new ec2.Instance(this, 'Instance', {
  vpc,
  instanceType: ec2.InstanceType.of(ec2.InstanceClass.T4G, ec2.InstanceSize.MICRO),
  machineImage: ami,
  securityGroup: securityGroup,
  role: role
});
```

- Define and provision cloud infra writing code
  - Supports TypeScript, JavaScript, Python, Java, C# and Go
- Converted to Cloud Formation

Monitoring AWS with Amazon CloudWatch

In 28 Minutes

- **Monitoring and observability** service
- Collects monitoring and operational data in the form of logs, metrics, and events
- Set alarms, visualize logs, take automated actions and troubleshoot issues
- Integrates with more than 70 AWS services:
  - Amazon EC2
  - Amazon DynamoDB
  - Amazon S3
  - Amazon ECS
  - AWS Lambda
  - and ....

Cloudwatch

Amazon CloudWatch Metrics

In28 Minutes

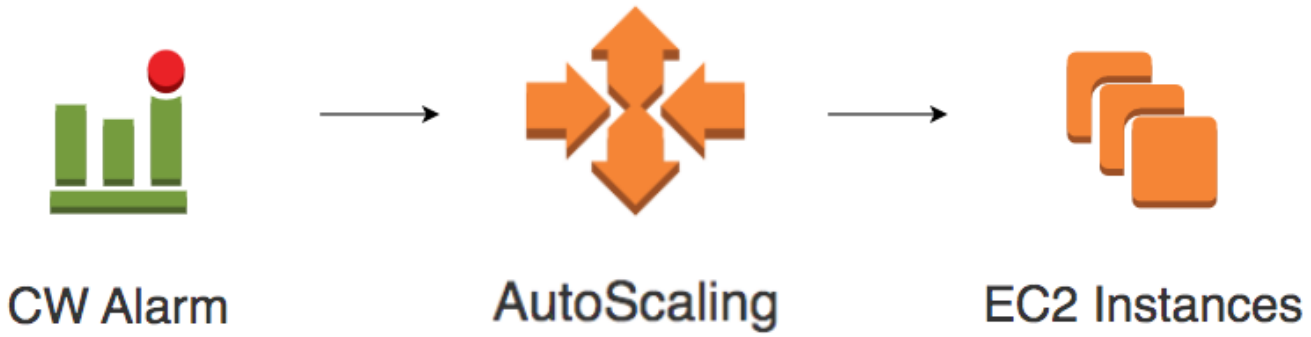
- Amazon CloudWatch Metrics: Most AWS services provide free metrics
  - Enable **detailed monitoring** ($$$) if needed
  - **EC2**: CPUUtilization, NetworkIn, NetworkOut
    - (DEFAULT) EC2 instances collect metrics every 5 minutes.
      - You can increase it to every one minute ($$$)
    - CloudWatch does **NOT** have access to **operating system metrics** like memory consumption
      - Install CloudWatch agent to gather metrics around memory
  - **ELB**: HTTPCode_Target_2XX_Count, HTTPCode_Target_4XX_Count
  - **DynamoDB**: AccountProvisionedReadCapacityUtilization, ConsumedReadCapacityUnits
  - **Lambda**: Throttles, Errors, ConcurrentExecutions, Duration
  - **API Gateway**: 4XXError,5XXError, CacheHitCount, CacheMissCount, Count
    - IntegrationLatency (How long did the backend take to process?)
    - Latency (How long did the total client request to API Gateway take?)
- Metrics exists only in the region in which they are created.

Amazon CloudWatch Logs

- Monitor and troubleshoot using system, application and custom log files
- **Real time application and system monitoring:**
  - Use **CloudWatch Logs Insights** to write queries and get actionable insights
    - Monitor for patterns in your logs and trigger alerts based on them
    - Example : Errors in a specific interval exceed a certain threshold
  - Use **CloudWatch Container Insights** to monitor, troubleshoot and set alarms for your containerized applications - EKS, ECS and Fargate
- **Long term log retention:**
  - Store logs in CloudWatch Logs for as long as you want
    - Default - forever. Configure expiry of your logs at log group level.
  - Or archive logs to S3 bucket
  - Or stream real time to Amazon Elasticsearch Service (Amazon ES) cluster using CloudWatch Logs subscription

Cloudwatch

Amazon CloudWatch Alarms



- **Create alarms based on:**
  - Amazon EC2 instance CPU utilization
  - Amazon SQS queue length
  - Amazon DynamoDB table throughput or
  - Your own custom metrics
- **Take immediate action:**
  - Send a SNS event notification
    - Send an email using SNS
  - Execute an Auto Scaling policy

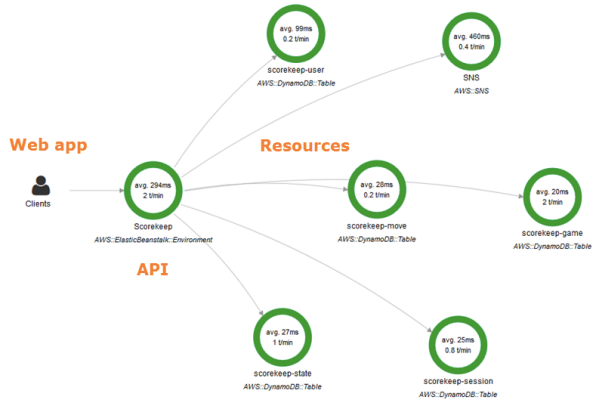
Amazon CloudWatch Events

In28 Minutes

- **Take immediate action** based on **events on AWS resources**
  - Call a AWS Lambda function when an EC2 instance starts
  - Notify an Amazon SNS topic when an Auto Scaling event happens
  - (ADDITIONAL FEATURE) **Schedule events** - Use Unix cron syntax
    - Schedule a call to Lambda function every hour or every minute
    - Send a notification to Amazon SNS topic every 3 hours
- **Example Use Cases:**
  - Send an email after execution of every stage in a pipeline
  - Send an email if an EC2 instance is stopped
- **Example Events:**
  - CodeBuild (Build State-change), CodeDeploy (Deployment State-change)
  - CodeCommit (pullRequestCreated)
  - CodePipeline (Pipeline Execution State Change, Stage Execution State Change)
  - Amazon EC2 State Change Events - stop, start, terminate

X-Ray

- Trace request across microservices/AWS services
  - Analyze, Troubleshoot errors, Solve performance issues
  - Gather tracing information
    - From applications/components/AWS Services
  - Tools to view, filter and gain insights (Ex: Service Map)
- How does Tracing work?
  - **Unique trace ID** assigned to every client request
    - `X-Amzn-Trace-Id:Root=1-5759e988-bd862e3fe`
  - Each service in request chain sends traces to X-Ray with trace ID
    - X-Ray gathers all the information and provides visualization
  - How do you **reduce performance impact** due to tracing?
    - **Sampling** - Only a sub set of requests are sampled (Configurable)
  - How can AWS Services and your applications send tracing info?
    - Step I : Update Application Code Using X-Ray SDK
    - Step II: Use **X-Ray agents** (EASY to use in some services! Ex: AWS Lambda)

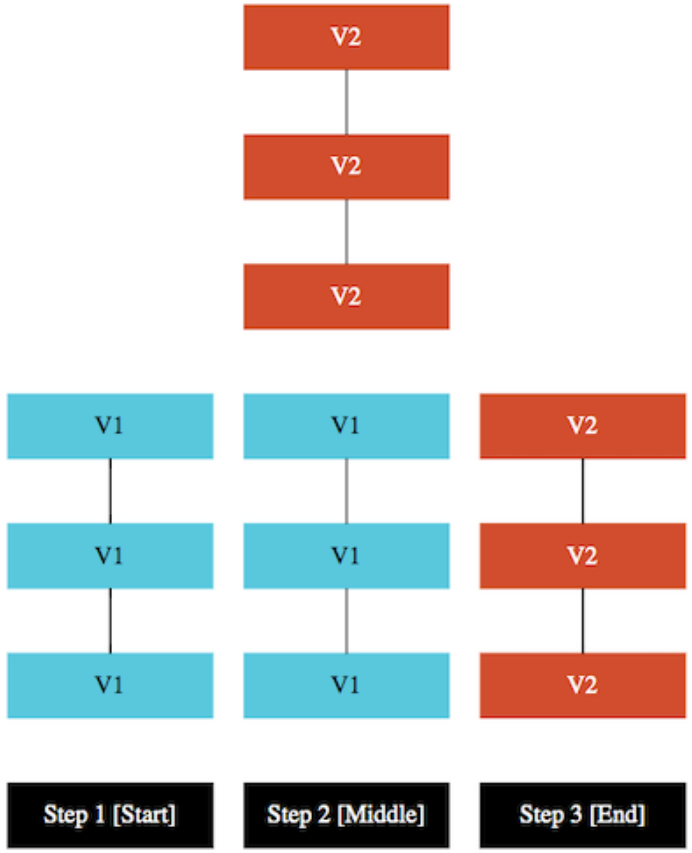


Release Management

Release Management

In28 Minutes

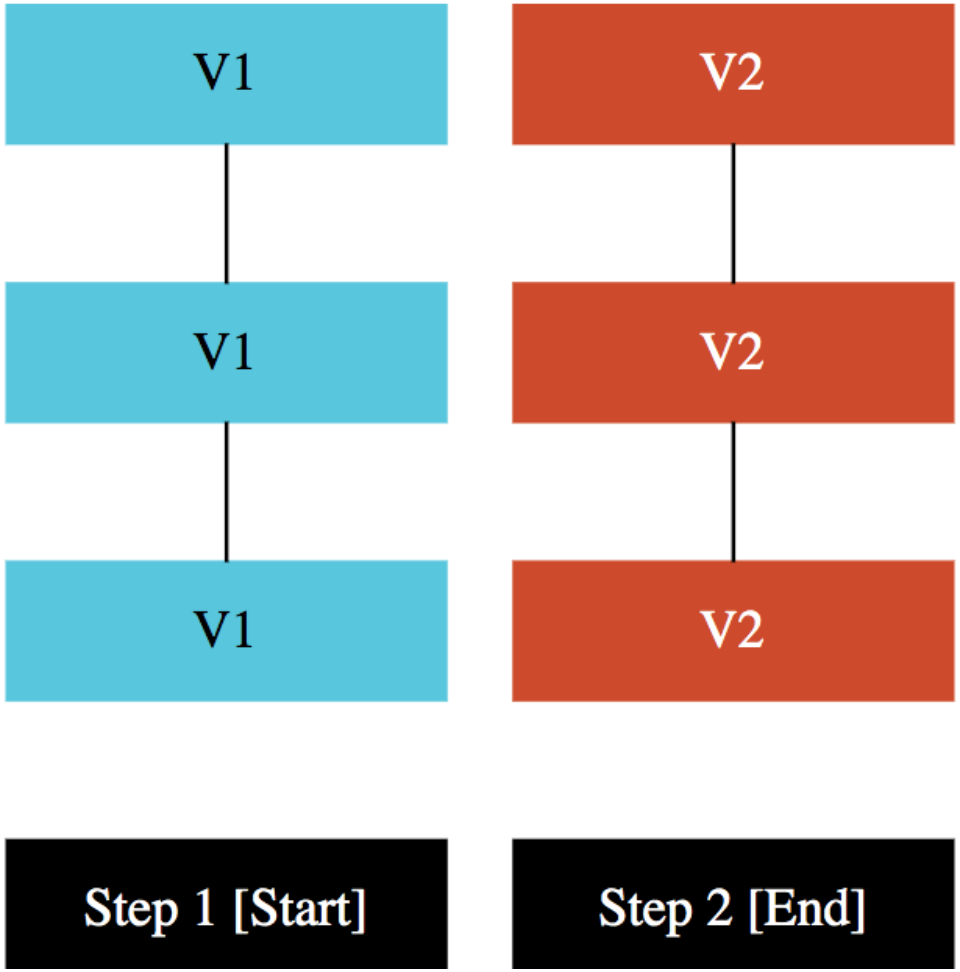
- **Goals**: vary from app to app
  - Zero Downtime
  - Only one version live at a time
  - Minimize Costs (and infrastructure needed)
  - Test with production traffic before going live
- **Best Practices**:
  - Small incremental changes
  - Automation (as much as possible)
  - Handling problems with new releases:
    - Analyze logs and metrics from Monitoring and Logging tools
    - Rollback to previous release and try replicating the problem in other environments



Deployment Approach : Recreate

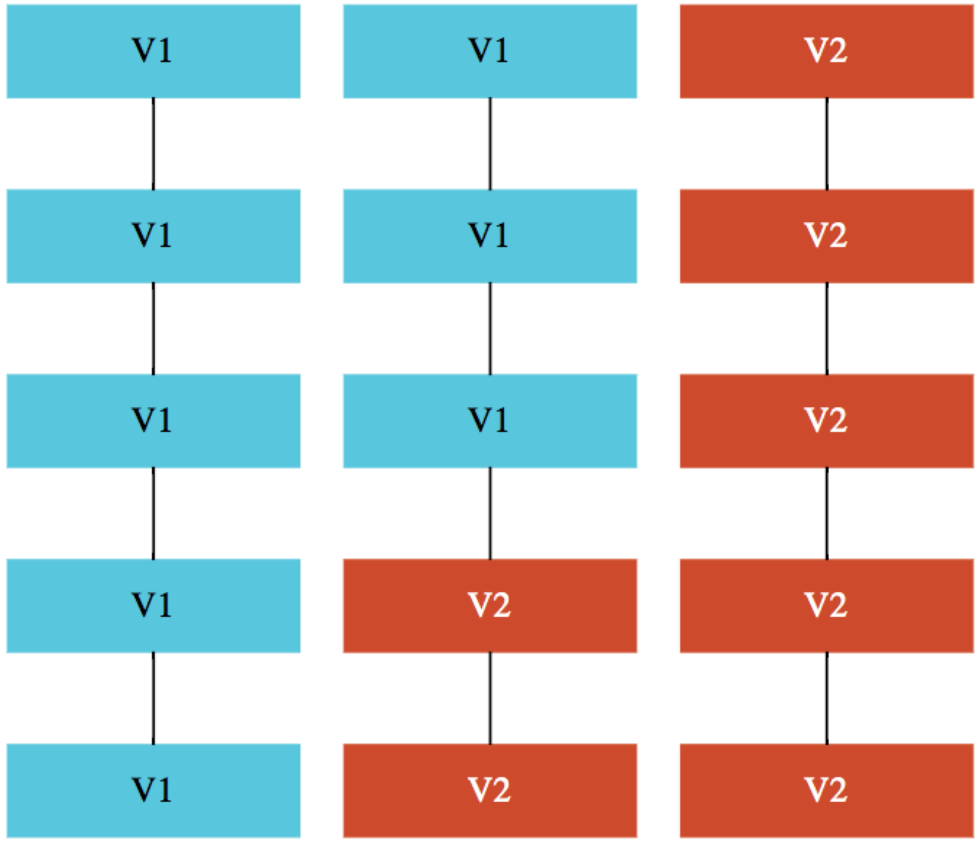
In28 Minutes

- **Approach:**
  - Terminate Version 1
  - Roll out Version 2
- **Characteristics:**
  - App down during the release
  - Rollback needs redeployment
    - AND more downtime
  - Cost effective and Fast
    - BUT disruptive
  - Avoid need for backward compatibility (data and applications)



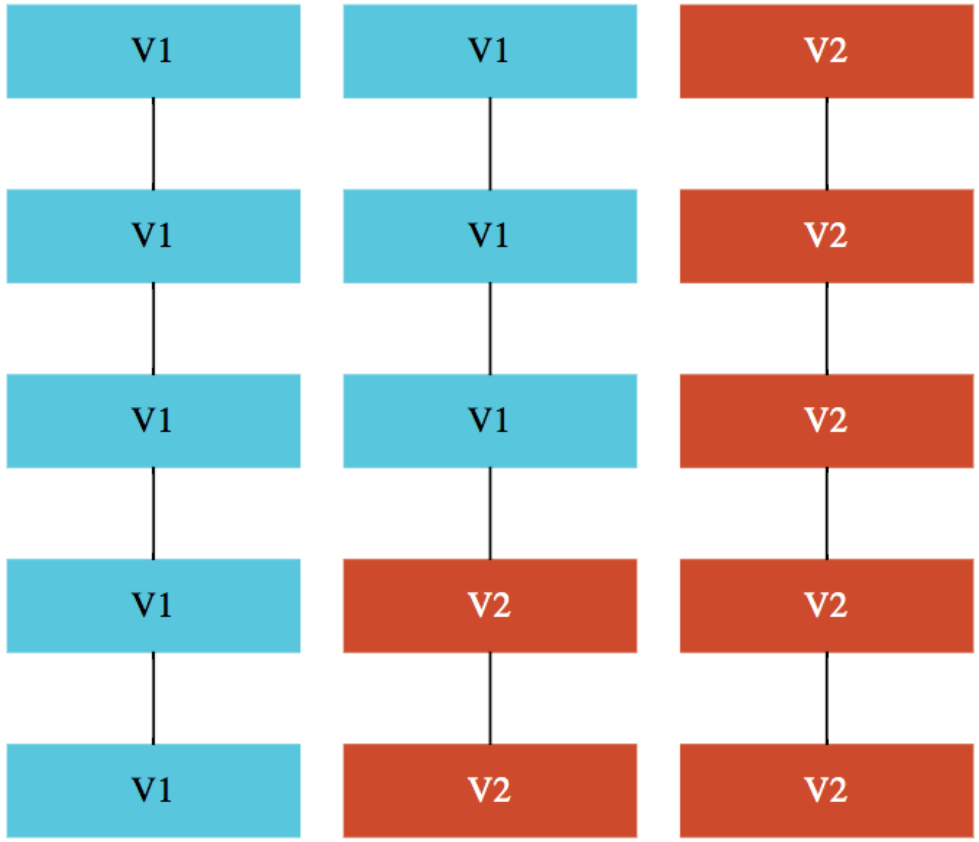
Deployment Approach : Canary

- **Approach:**
  - **Step 1**: V2 rolled out to a subset of instances
  - **Step 2**: On successful testing, V2 rolled out to all instances
    - OR V2 is rolled back in case of failure
- **Characteristics:**
  - Fast
  - Zero downtime
  - No extra infrastructure
  - Minimizes impact to users (in case of release failures)
  - Needs Backward compatibility (data and applications)



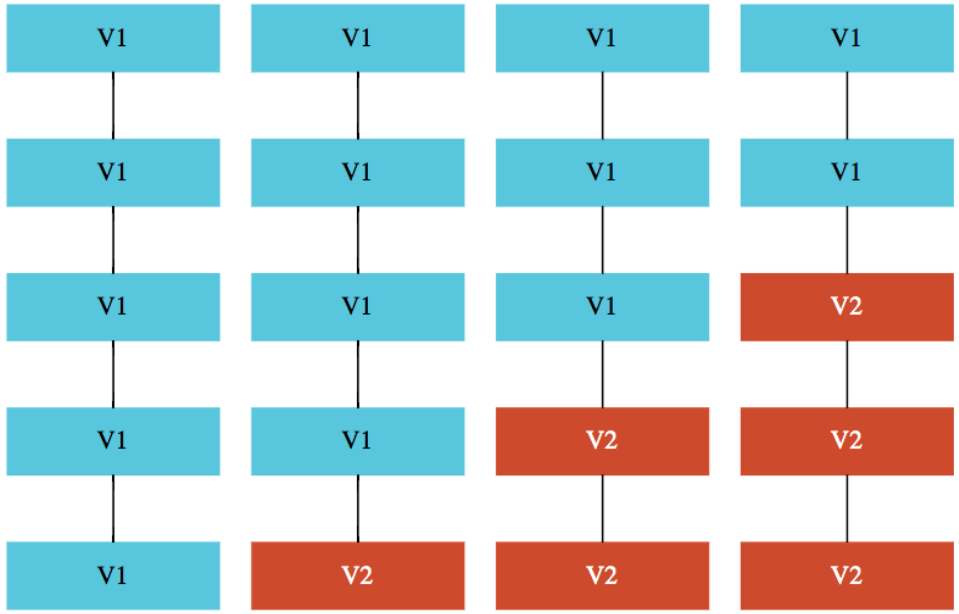
Testing Approach : A/B Testing

- **Use case**: You want to see if users like a feature!
- **Approach:**
  - **Step 1**: V2 (with new feature) rolled out to a subset of users
  - **Step 2**: On successful testing, V2 rolled out to all users
    - OR we go back to V1 in case users don't like the feature!
- **Characteristics:**
  - Gives the ability to test if your users like a feature



Deployment Approach : Rolling

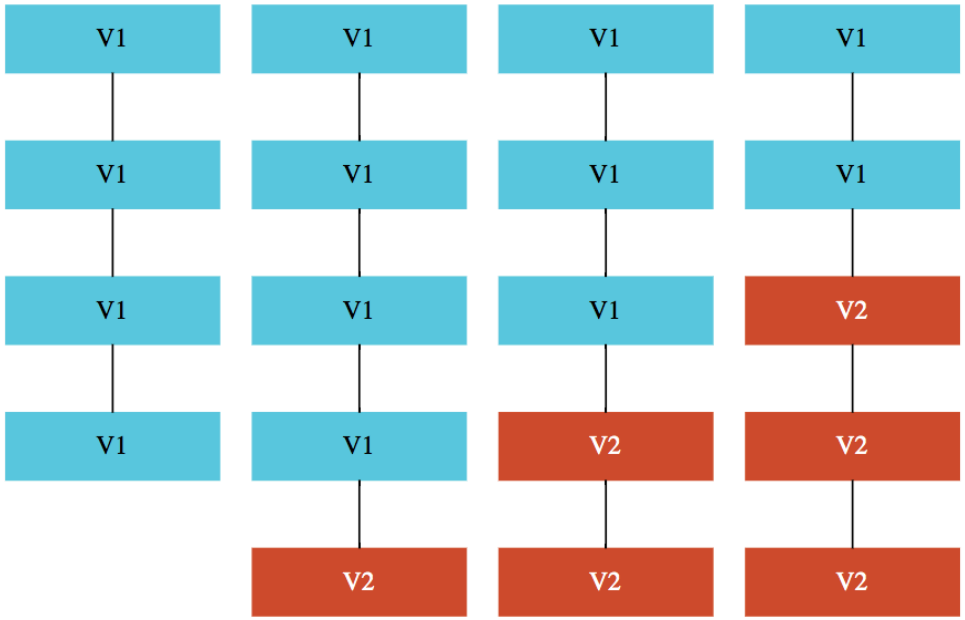
- **Approach:**
  - **Step 1**: V2 rolled out to a percentage of instances (Example window size: 5%)
  - **Step 2..N**: V2 gradually rolled out to rest of the instances (Example: 5% at a time)
- **Characteristics:**
  - Slow
  - Zero downtime
  - Needs automation and additional setup
  - No extra infrastructure
  - Minimizes impact to users (in case of release failures)
  - Needs Backward compatibility (data and applications)



Deployment Approach : Rolling with Additional Batch

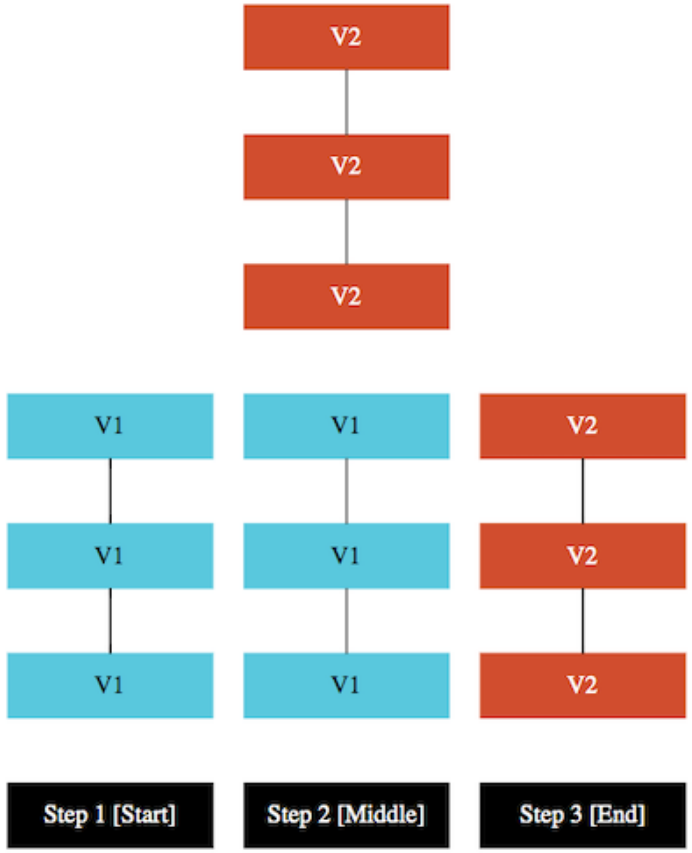
In28 Minutes

- **Approach:**
  - **Step 1**: Additional batch of new instances are created with V2 (Example: 5%)
  - **Step 2..N**: V2 gradually rolled out to the instances batch by batch (Example: 5% at a time)
- **Characteristics:**
  - Same as Rolling Deployment except for:
    - Needs Little bit of extra infrastructure
    - ZERO reduction in number of instances handling user requests



Deployment Approach : Blue Green

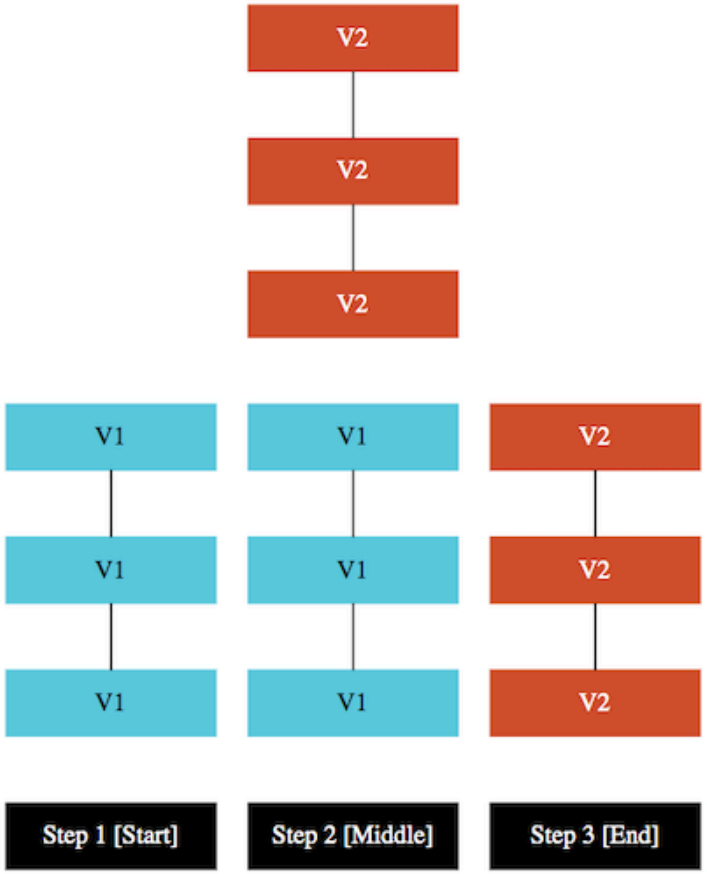
- **Approach:**
  - **Step 1**: V1 is Live
  - **Step 2**: Create (or replicate) a parallel environment with V2
  - **Step 3**: Switch all traffic from V1 to V2 (and remove V1 Environment)
- **Characteristics:**
  - Instant
  - Zero Downtime
  - Easy Rollback
  - Needs additional infra (during the release)
  - ZERO reduction in available capacity
  - Needs Backward compatibility (data and apps)



Testing Approach : Shadow

In28 Minutes

- **Approach:**
  - **Step 1**: V1 is Live
  - **Step 2**: Create (or replicate) a parallel environment with V2
    - Mirror traffic to V1 and V2
  - **Step 3**: Switch all traffic from V1 to V2 (and remove V1 Environment)
- **Characteristics:**
  - Zero production impact: Test V2 with real production traffic before releasing
    - You can also capture and replay live production traffic
  - Complicated : You don't want double payments (might need stubbing)
  - Needs a lot of additional infrastructure



AWS CodeDeploy

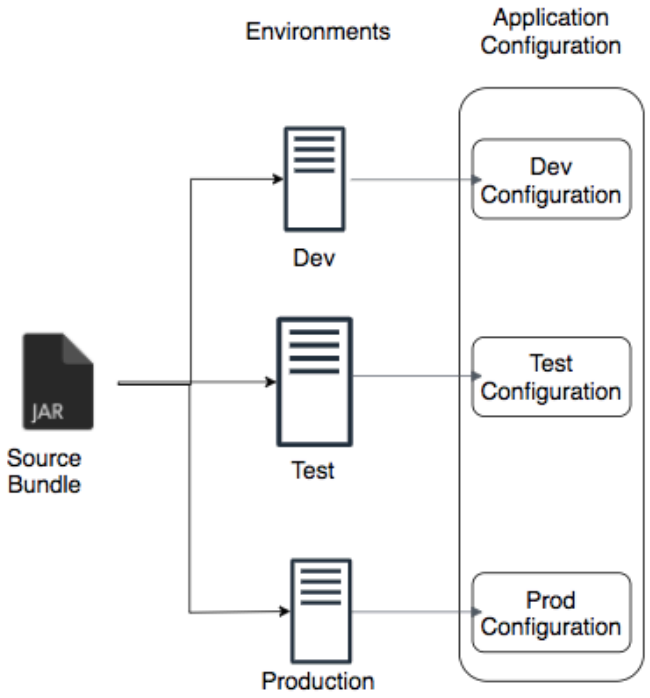
In28 Minutes

- **Automate application deployments** to:
  - Amazon EC2
  - Lambda functions
  - Amazon ECS
  - On premise instances
- Support **variety of deployment strategies**:
  - Canary
  - Rolling (In-place deployment)
  - Rolling with Additional Batch
  - All-at-once
  - Blue/green deployment

Codedeploy

Application Configuration Management

- You want to **connect to a different database in different environments**
  - How do you externalize database configuration from the application?
  - How do you decouple your application from the specific configuration needed in a specific environment?
- **Considerations:**
  - Is the configuration secure?
    - Are the configuration values encrypted?
    - Can you store passwords?
  - How can application retrieve the configured values?
  - What is involved in changing the configuration values?



Environment variables

- **Goal**: Adjust app/function behavior without updating code
- Key value pairs directly associated an environment
- **AWS Lambda**: Configure environment variables and use `process.env.ENV_VAR_NAME` or `System.getenv("ENV_VAR_NAME")` in Lambda function to get the value.
  - (REMEMBER) Environment variables are locked when a Lambda version is published
- **AWS ECS**: Use environment (--env) or environmentFiles (--env-file) in container definition
- **Kubernetes**: Create ConfigMap and Secrets

AWS Systems Manager Parameter Store

In28 Minutes

- Manage **Application Configuration and Secrets**
  - Supports hierarchical structure
  - Maintains history of configuration over a period of time
- **Multi language SDK support** to retrieve configuration:
  - `ssm.get_parameters(Names=['LambdaSecureString'])`
- **Simplified Operations**:
  - Configuration can be changed without releasing a new Lambda version!
  - Monitoring (CloudWatch), Notifications(SNS) and Auditing(AWS CloudTrail)
- **Integrates with**:
  - **AWS KMS** - Encrypt your configuration values
  - **Amazon EC2, Amazon ECS, Amazon EKS, AWS Lambda, ..** - Use configured values from your code
  - **AWS Secrets Manager** - More powerful management for secrets (Automatic Rotation)

AWS Secrets Manager

In28 Minutes

- Service **dedicated to secrets management**
  - Rotate, Manage and retrieve database credentials, API keys, and other secrets for your applications
  - Encrypt your secret data using KMS
  - ($$$) Pay for use (**NOT FREE**)
- **Simplified Operations:**
  - (KEY FEATURE) Rotate secrets automatically without impacting applications
    - Supported for Amazon RDS, Amazon Redshift, and Amazon DocumentDB
  - Configuration can be changed without releasing a new Lambda version!
  - Monitoring (CloudWatch), Notifications(SNS) and Auditing(AWS CloudTrail)
- (RECOMMENDED Workloads) Automatic rotation of secrets for compliance

AWS AppConfig (part of AWS Systems Manager)

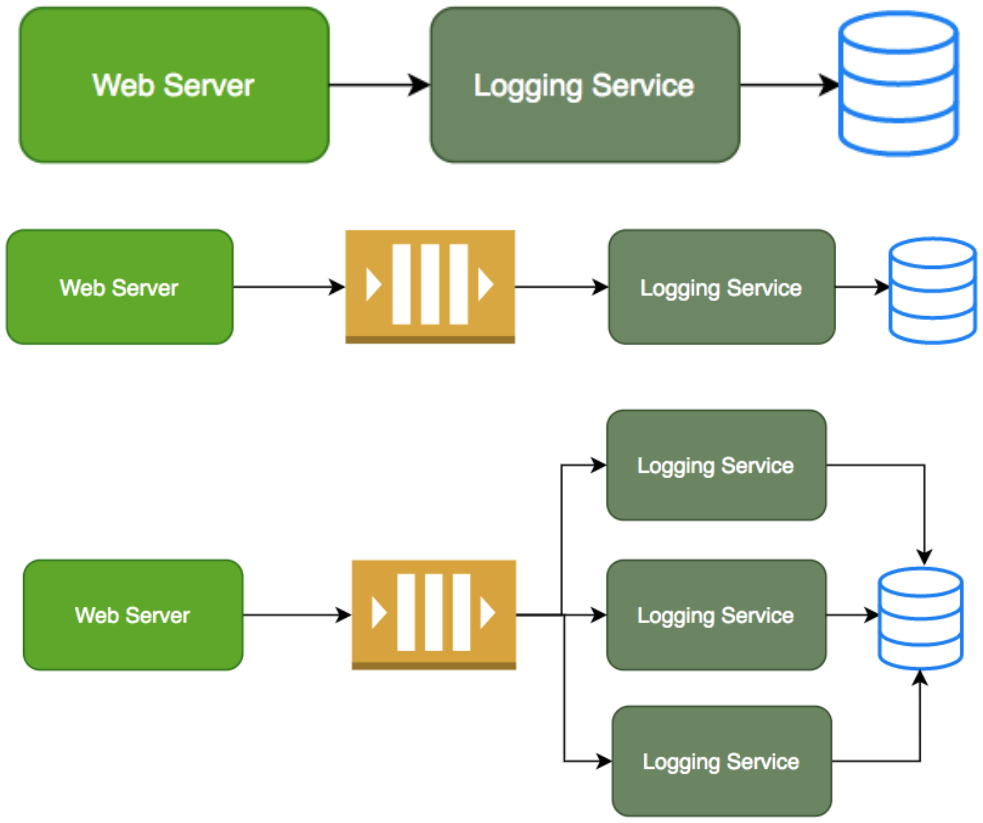
In28 Minutes

- Making configuration changes typically **needs a restart of the application**
  - And what if you want to **control the roll out** of a configuration change (canary approach, ..)?
- **AWS AppConfig**: Create, manage, & quickly deploy application configurations
  - (Feature) Configuration can be stored in Amazon Simple Storage Service (Amazon S3), AWS AppConfig hosted configurations, Parameter Store, ...
  - (Feature) Use a JSON schema or write a Lambda function to validate your configuration before it is deployed
  - (Feature) Control the speed of deployment of the new configuration
    - Monitor deployments and automatically roll back in case of errors (integrates with CloudWatch alarms)
- **Advantages**:
  - Reduce errors in configuration changes
  - Deploy changes across a set of targets quickly
  - Update applications without interruptions
  - Control deployment of changes across your application

Decoupling Applications with SQS and SNS

Synchronous vs Asynchronous Communication

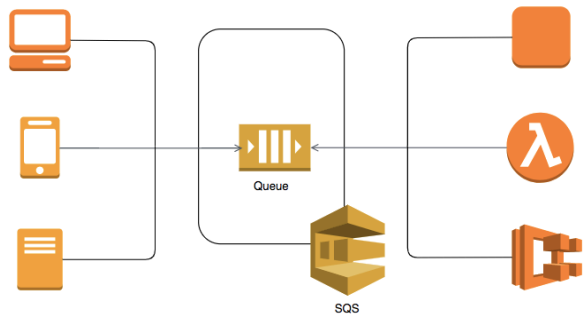
- **Synchronous Communication:**
  - What if your logging service goes down?
    - Will you applications go down too?
  - What if there is high load?
    - Log Service unable to handle and goes down
- **Asynchronous Communication:**
  - Create a queue or a topic
    - Your applications put the logs on the queue
    - Picked up when the logging service is ready
  - Good example of decoupling!
  - (Possible) Multiple logging service instances reading from the queue!



Asynchronous Communication - Pull Model - SQS

In28 Minutes

- Producers put messages. Consumers poll on queue.
  - Only one of the consumers will successfully process a message
- **Advantages**:
  - Scalability: Scale consumer instances under high load
  - Availability: Producer up even if a consumer is down
  - Reliability: Work is not lost due to insufficient resources
  - Decoupling: Make changes to consumers without effect on producers worrying about them
- **Features**:
  - Reliable, scalable, fully-managed message queuing service
  - High availability
  - Unlimited scaling
    - Auto scale to process billions of messages per day
  - Low cost (Pay for use)



Standard and FIFO Queues

In28 Minutes

- **Standard Queue**
  - Unlimited throughput
  - BUT NO guarantee of ordering (Best-Effort Ordering)
  - and NO guarantee of exactly-once processing
    - Guarantees at-least-once delivery (some messages can be processed twice)

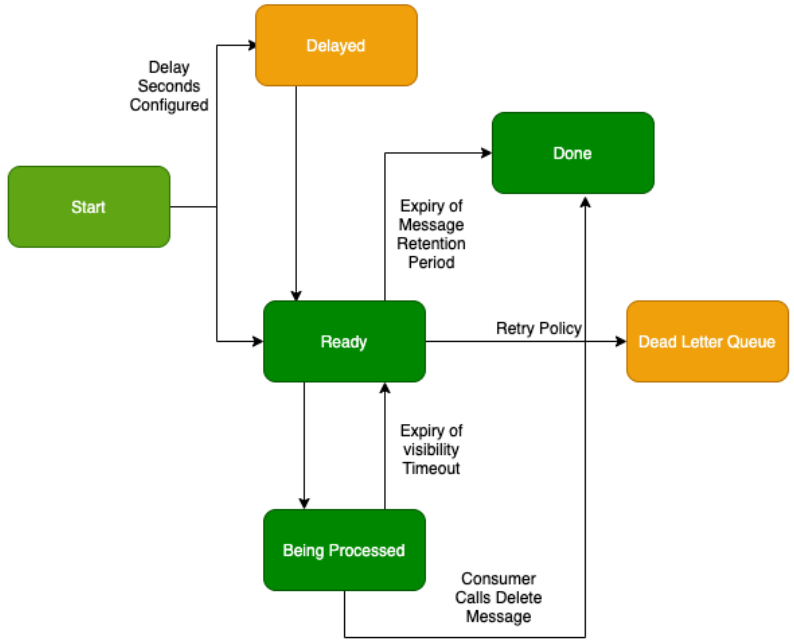
Amazon SQS

- **FIFO (first-in-first-out) Queue**
  - First-In-First-out Delivery
  - Exactly-Once Processing
  - **BUT** throughput is lower
    - Upto 300 messages per second (300 send, receive, or delete operations per second)
    - If you batch 10 messages per operation (maximum), up to 3,000 messages per second
- **Choose**
  - Standard SQS queue if throughput is important
  - FIFO Queue if order of events is important

Sending and receiving a SQS Message - Best case scenario

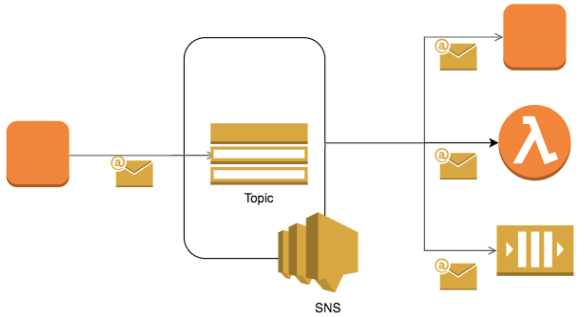
In28 Minutes

- **1:** Producer places message on queue
  - Receives globally unique message ID ABCDEFGHIJ (used to track the message)
- **2:** Consumer polls for messages
  - Receives the message ABCDEFGHIJ along with a receipt handle XYZ
- **3:** Message remains in the queue while the consumer processes the message
  - Other consumers will not receive ABCDEFGHIJ even if they poll for messages
- **4:** Consumer processes the message
  - Calls delete message (using receipt handle XYZ)
  - Message is removed from the queue



Asynchronous Communication - Push Model - SNS

- **How does it work** (Publish-Subscribe(pub-sub))?
  - 1: Create an SNS Topic
  - 2: Subscribers can register for a Topic
  - 3: When an SNS Topic receives an event notification (from publisher), it is broadcast to all Subscribers
  - (Advantage) Decoupling: Producers don't care about Consumers
  - (Advantage) Availability: Producer up even if subscriber is down
- **Use Cases**: Monitoring Apps, workflow systems
- Provides mobile and enterprise messaging services
  - Push notifications to Apple, Android, FireOS, Windows devices
  - Send SMS to mobile users and Emails
- REMEMBER : SNS does not need SQS or a Queue

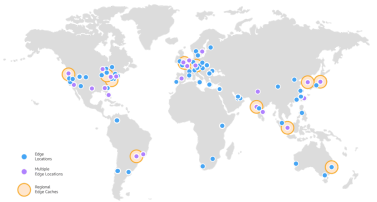


Routing and Content Delivery

Amazon CloudFront - Content Delivery Network

In 28 Minutes

- **Deliver content to your global audience**:
  - AWS provides 200+ edge locations around the world
  - Provides high availability and low latency
  - Serve users from nearest edge location (based on user location)
  - If content is not available at the edge location, it is retrieved from the origin server and cached
- **Use Cases**: Static web apps, Downloads (media/software)
- **Content Source**: S3, EC2, ELB or External Websites
- **Integrates with**: AWS Shield (Avoid DDoS attacks)
  - AWS WAF (protect from SQL injection, cross-site scripting)
- **Cost Benefits**: Zero cost for transfer from S3 to CloudFront
  - Reduce compute workload for your EC2 instances



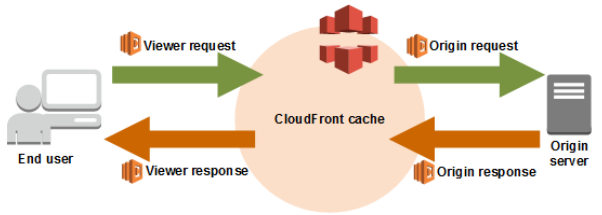
Recommended Architecture - Static Content

- (NOT RECOMMENDED) Serving static content from EC2
- Store static content in S3
- Distribute it to edge locations around the world using CloudFront
- **Advantages**:
  - Pay for use
  - Low latency
  - Simple Caching (with TTL)
  - Reduce load on your compute instances (for example, EC2)

AWS Lambda@Edge

In28 Minutes

- **Run Lambda functions to customize CloudFront content**
  - (RESTRICTION) ONLY Python or Node JS supported
- Lambda functions can be run at **different points in processing** a request in CloudFront :
  - After CF receives a request from a viewer (Viewer Request)
  - Before CF forwards the request to Origin (Origin Request)
  - After CF receives response from Origin (Origin Response)
  - Before sending response to the viewer (Viewer Response)
- Use Cases:
  - **A/B Testing** - URL rewrite to different versions of a site
  - **Multi device support** - Based on User-Agent header, send pictures of different resolution
  - **Generate new HTTP responses** - redirect unauthenticated users to Login page



https://docs.aws.amazon.com/lambda/latest/dg/lambda-edge.html

Route 53 = Domain Registrar + DNS (Domain Name Server)

In28 Minutes

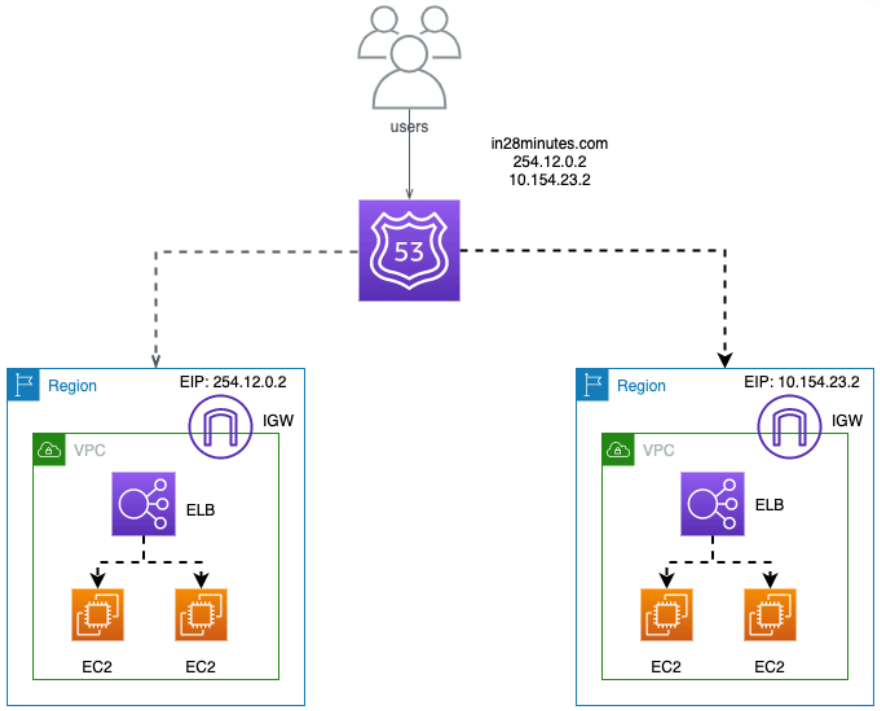
- What would be the steps in **setting up a website with a domain name** (for example, in28minutes.com)?
  - **Step I** : Buy the domain name in28minutes.com (Domain Registrar)
  - **Step II** : Setup your website content (Website Hosting)
  - **Step III** : Route requests to in28minutes.com to the my website host server (DNS)

Route53

- **Route 53 = Domain Registrar + DNS**
  - **Domain Registrar** - Buy your domain name
  - **DNS** - Setup your DNS routing for in28minutes.com
    - Configure Records for routing traffic for in28minutes.com
      - Route api.in28minutes.com to the IP address of api server
      - Route static.in28minutes.com to the IP address of http server
      - Route email (ranga@in28minutes.com) to the mail server(mail.in28minutes.com)
      - Each record is associated with a TTL (Time To Live) - How long is your mapping cached at the routers and the client?

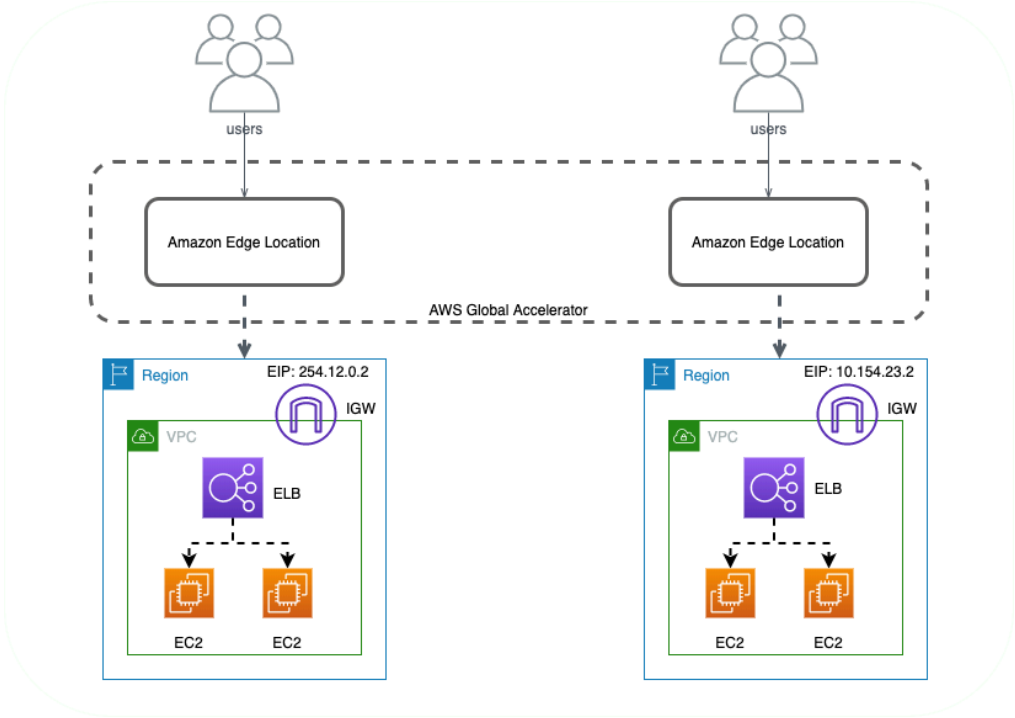
Need for AWS Global Accelerator

- Cached DNS answers
  - clients might cache DNS answers causing a delay in propagation of configuration updates
- High latency
  - users connect to the region over the internet



AWS Global Accelerator

- Directs traffic to optimal endpoints over the AWS global network
- Global Accelerator provides you with two static IP addresses
- Static IP addresses are anycast from the AWS edge network
  - Distribute traffic across multiple endpoint resources in multiple AWS Regions
- Works with Network Load Balancers, Application Load Balancers, EC2 Instances, and Elastic IP addresses



Management and Governance

AWS CloudTrail and AWS Config

In28 Minutes

Solution	Description
AWS CloudTrail	Track events, API calls, changes made to your AWS resources. Who (made the request), What (action, parameters, end result) and When? **Multi Region Trail** - One trail for all AWS regions vs **Single Region Trail** - Only events from one region
AWS Config	**Auditing**: Complete inventory of your AWS resources **Resource history and change tracking** - Find how a resource was configured at any point in time **Governance** - Customize Config Rules for specific resources or for entire AWS account and Continuously evaluate compliance against desired configuration
AWS Config vs AWS CloudTrail	AWS Config - What did my AWS resource look like? AWS CloudTrail - Who made an API call to modify this resource?

AWS Organizations

- **AWS Organizations**: Simple management for multiple AWS accounts
  - Organize accounts into Organizational Units (OU)
  - Consolidated bill for AWS accounts
  - Centralized management for AWS Config Rules
  - Send AWS CloudTrail data to one S3 bucket (across accounts)
  - AWS Firewall Manager to manage firewall rules (WAF, Shield and Security Groups)
- Use **Service control policies(SCP)** to define cross account restrictions
  - Require Amazon EC2 instances to use a specific type
  - Require MFA to stop an Amazon EC2 instance
  - Require a tag upon resource creation
- Use **AWS Resource Access Manager** to share AWS resources:
  - Share AWS Transit Gateways, Subnets, ... with other AWS accounts or your AWS Organization (Optimize costs)

AWS Trusted Advisor

- Cost optimization, performance, security & fault tolerance recommendations

- **4 FREE Checks**:
  - Service limits (usage > 80%)
  - Security groups having unrestricted access (0.0.0.0/0)
  - Proper use of IAM
  - MFA on Root Account
- Enable **Business or Enterprise AWS support plan** for over 50 checks
  - Cost Optimization: Unused resources, Other opportunities (ex: reserved instances)
  - Security : Settings to make your AWS solution more secure (ex: security group)
  - Fault Tolerance: Redundancy improvements, over-utilized resources
  - Performance: Improve speed and responsiveness of your AWS solutions
  - Service Limits: Is your usage is more than 80% of service limits?

Cost Management

Consumption-based vs Fixed-price Pricing Models

In 28 Minutes

- **Consumption-based** - You are billed for only what you use
  - Example: Serverless FaaS - You pay for no of invocations!
- **Fixed-price** - You are billed for instances irrespective of whether they are used or not
  - **Example**: You provision a VM instance
    - You pay for its lifetime irrespective of whether you use it or NOT
  - **Example**: You provision a Kubernetes cluster
    - You are billed irrespective of whether you use it or not

Expenditure Models: CapEx vs OpEx

In28 Minutes

- **Capital Expenditure (CapEx)**: Money spent to buy infrastructure
  - Additional cost to maintain infrastructure with time
  - You might need a team to manage the infrastructure
  - **Example**: Deploying your own data center with physical servers
  - **Example**: Purchasing Reservations
  - **Example**: Leasing Software
- **Operational Expenditure (OpEx)**: Money spent to use a service or a product
  - **Zero upfront costs**
  - You Pay for services as you use them (Pay-as-you-go model)
  - **Example**: Provisioning VMs as you need them
  - **Example**: Using Serverless FaaS and paying for invocations

How is Cost Decided?

In28 Minutes

Factor	Details
Resource type & cnfgn	How much memory? How much CPU?
Usage meters	How long was your VM running for? How many invocations of an Serverless FaaS function?
Which Region?	Price varies from Region to Region
Data transfer outside cloud	Migrating data from on-premises to cloud is typically free Migrating data from cloud to On-Premises is NOT free
Data transfer within cloud	Within the same zone - free Between different zones in same region might not be free Between regions is almost always NOT free
Reserved or Not	Some services offer reservations ahead of time
Recommended	Prefer private IP address to public IP addresses. Maximize traffic that stays with an AZ (at least with a Region)

Billing and Cost Management Services/Tools

In 28 Minutes

Service	Description
AWS Billing and Cost Management	Pay your AWS bill, monitor your usage **Cost Explorer** - View your AWS cost data as a graph (Filter by Region, AZ, tags etc. See future cost projection.) **AWS Budgets** - Create a budget (Create alerts (SNS)) Recommendation: Enable Cost allocation tags. Helps you categorize your resource costs in Cost Management.
AWS Compute Optimizer	Recommends compute optimizations to reduce costs (Ex: Right-sizing - EC2 instance type, Auto Scaling group configuration)
AWS Pricing Calculator (NEW)	Estimate cost of your architecture solution
AWS Simple Monthly Calculator (OLD)	Estimate charges for AWS services
TCO - Total Cost of Ownership Calculator (OLD)	Compare Cost of running applications in AWS vs On Premise

Total Cost of Ownership(TCO)

In28 Minutes

- **Total Cost of Ownership(TCO)** includes:
  - Infrastructure Costs
    - Procuring Servers, Databases, Storage, Routers ..
    - Infrastructure maintenance costs
  - Licensing Costs (Software + Hardware)
  - Networking Costs (Connection cost + Data Ingress + Data Egress)
  - Personnel Costs (Dev + Test + Ops + Business + ..)
  - Other Costs:
    - Penalties for missed SLAs or Compliance needs
    - Third Party APIs
    - Electricity costs
- **When designing solutions (or migrating to cloud)**, ensure that you take Total Cost of Ownership(TCO) into account!
  - Compare Apples to Apples!

Managing Costs - Best Practices

In 28 Minutes

- **Group resources** based on cost ownership
  - Tags etc.
- **Regular cost reviews** (at least weekly)
  - CapEx (Ahead of time planning) -> OpEx (regular reviews)
- **Estimate costs** before you deploy (Pricing Calculator)
- Use **Cost Management features**
  - Budgets and Budgets alerts etc.
- **Others**:
  - Stop Resources when you don't need them
  - Use Managed Services (PaaS >>> IaaS)
  - Reserve compute for 1 or 3 years
  - Use Spot instances for fault tolerant non-critical workloads
  - Involve all teams - executive, management, business, technology & finance

Security

Amazon Cognito

In28 Minutes

- Add authentication & authorization to your mobile & web apps
  - **Integrate with web identity providers** (ex: Google, Facebook)
  - Add **multi-factor authentication (MFA)**, phone and email verification
  - Sync user data across devices, platforms, and applications
  - **User Pools**: Create **your own secure and scalable user directory**
    - Create **sign-up (or registration) pages**
    - Customizable **web UI to sign in users** (with option to social sign-in)
    - Integrates with Application Load Balancer and API Gateway
    - Provides triggers to customize workflow - **Pre Authentication Lambda Trigger**, **Pre Sign-up Lambda Trigger**,**Post Confirmation Lambda Trigger** etc
  - **Identity pools**: Provide **access to AWS resources** to **your users**
    - Integrate with your own user pool or OpenID Connect provider (Amazon, Apple, Facebook, Google+, Twitter) or SAML identity providers (Corporate)
    - Allows **multiple** authentication (identity) providers

AWS Shield

In 28 Minutes

- **Shields from Distributed Denial of Service** (DDoS) attacks
  - Disrupt normal traffic of a server by overwhelming it with a flood of Internet traffic
  - Protect Amazon Route 53, CloudFront, AWS Global Accelerator, EC2 instances and Elastic Load Balancers (ELB)
- AWS Shield Standard is automatically enabled (ZERO COST)
  - Protection against common infrastructure (layer 3 and 4) DDoS attacks
- Enable AWS Shield Advanced ($$$) for Enhanced Protection:
  - 24x7 access to the AWS DDoS Response Team (DRT)
  - Protects your AWS bill from usage spikes as a result of a DDoS attack
- **RECOMMENDED:** Protect any web application (from Amazon S3 or external) from DDoS by putting Amazon CloudFront enabled with AWS Shield in front of it

AWS Shield

AWS WAF - Web Application Firewall

In28 Minutes

- **AWS WAF** protect your web applications from **OWASP Top 10 exploits, CVE and a lot more**!
  - **OWASP (Open Web Application Security Project) Top 10**
    - List of broadly agreed "**most critical security risks to web applications**"
    - Examples : SQL injection, cross-site scripting etc
  - **Common Vulnerabilities and Exposures (CVE)** is a list of information-security vulnerabilities and exposures
- Can be deployed on Amazon CloudFront, Application Load Balancer, Amazon API Gateway
- Customize rules & trigger realtime alerts (CloudWatch Alarms)
- **Web traffic filtering** : block attacks
  - Filter traffic based on IP addresses, geo locations, HTTP headers and body (block attacks from specific user-agents, bad bots, or content scrapers)

AWS WAF

Other Important Security Services

In28 Minutes

Solution	Description
Amazon Macie	Fully managed data security and privacy service Uses machine learning to identify sensitive data in Amazon S3 (Recommendation) When migrating data to AWS use S3 for staging and Run Macie
Amazon GuardDuty	Continuously monitor AWS environment for suspicious activity (Intelligent Threat Detection) Analyze AWS CloudTrail events, VPC Flow Logs etc
Amazon Detective	Analyze, and quickly identify the root cause of potential security issues. Collects log data from your AWS resources and uses machine learning enabling you to perform more efficient security investigations.
Certificate Manager	Provision, manage, deploy, and renew SSL/TLS certificates on the AWS platform
Penetration Testing	Testing application security by simulating an attack

Digital Transformation

What has changed in last decade or so?

In 28 Minutes

- How consumers make purchase decisions? (**Social**)
- How we do things? (**Mobile**)
- How much data we have? (**Big Data**)
  - How much intelligence we can get? (**AI/ML**)
- How much access startups have to technology at scale? (**Cloud**)

Enterprises have to adapt (or get disrupted)

In 28 Minutes

- **Enterprises can ADAPT** by:
  - Providing awesome (omni-channel - social, mobile) customer experiences
  - Getting intelligence from data (Big Data, AI/ML)
    - Example: Personalize consumer offerings
  - Enabling themselves to make changes faster
    - Cultural change from "traditional Datacenter, SDLC, manual IT Ops" to "Cloud, Containers, DevOps/SRE, Automation"
- **Digital Transformation**: Using modern technologies to create (or modify) business processes & customer experiences by innovating with technology and team culture
  - Focus on WHY (NOT HOW)
    - Increase pace of change
    - Revenue Growth
    - Cost Savings
    - Higher customer engagement/retention

Cloud - Enabler for Digital Transformation

In 28 Minutes

- Cloud can **ENABLE** Digital Transformations
  - Lower cost
  - Reduced responsibilities
  - Higher capabilities
  - Increased speed to market
- **BUT needs a change** in skills, mindset and culture
  - Modern Architectures (Microservices, Serverless, Containers, Kubernetes)
  - More Agile Processes (DevOps)
  - Right Talent
  - Right Culture (of data driven experimentation and innovation)

Cloud Mindset

In28 Minutes

Factor	Data Center	Cloud
Infrastructure	Buy	Rent
Planning	Ahead of time	Provision when you need it
Deployment	VMs	PaaS or Containers or Serverless
Team	Specialized skills	T-shaped skills
Releases	Manual	CI/CD with flexible release options (Canary, A/B Testing, ...)
Infrastructure Creation	Manual	Infrastructure as Code
Attitude	Avoid Failures	Move Fast by Reducing Cost of Failure (Automation of testing, releases, infrastructure creation and monitoring)

Cloud Migration Approaches

In28 Minutes

- You have a **combination of following options**:
  - **Rehosting** ("lift and shift")
  - **Replatforming**
    - Few adjustments to suit the cloud
    - Example: Containerizing
  - **Repurchasing**
    - Move to a new, cloud-native product
      - Move to a new database
  - **Refactoring**
    - Example: Serverless Computing
    - Most expensive
  - **Retiring**
    - End of service
  - **Retaining**
    - Do NOT move to Cloud
    - Stay on-premises

Cloud Migration - AWS Managed Services

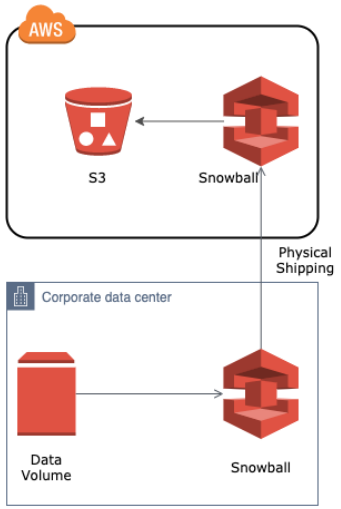
In28 Minutes

Service	Description
AWS Migration Hub	Single place to track application migrations across multiple AWS and partner solutions
AWS Application Discovery Service	Discover on-premises application inventory and dependencies
AWS Control Tower	Easiest way to set up and govern a secure multi-account AWS environment (creates your landing zone using AWS Organizations)
AWS Application Migration Service (MGN)	Lift and shift applications to AWS
AWS Mainframe Modernization	Migrate Mainframe Applications to AWS

AWS Snowball and AWS Snowmobile

In28 Minutes

- Transfer terabytes or petabytes of data from on-premises
  - 100TB (80 TB usable) per appliance with automatic encryption (KMS)
  - Simple Process: Request for Snowball, Copy data and Ship it back
  - Manage jobs with AWS Snowball console
- Current versions of AWS Snowball use Snowball Edge devices
  - Provide both compute and storage(Storage or Compute or GPU Optimized)
  - Pre-process data (using Lambda functions)
- Choose Snowball if direct transfer takes over a week
  - 5TB can be transferred on 100Mbps line in a week at 80% utilization
    - S3 Transfer Acceleration - Basic option to transfer less data (upto a few terabytes) into S3 Uses Amazon CloudFront's Edge Locations.
    - AWS DataSync - 10x faster (100s of TB) data transfers from/to AWS over internet or AWS Direct Connect (multiple destinations, built-in retry).
  - Use **Snowmobile** Trucks (100PB per truck) for dozen petabytes to exabytes



Data Migration Alternatives

In28 Minutes

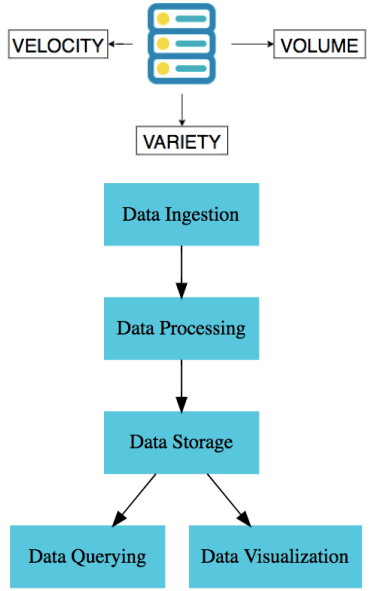
Solution	Description
AWS Data Pipeline	Process and move data (ETL) between S3, RDS, DynamoDB, EMR, On-premise data Create complex data processing workloads (NOT for streaming data!)
AWS Database Migration Service	Migrate databases to AWS while keeping source databases operational (AFTER MIGRATION) Keep databases in sync and pick right moment to switch (Use case) Consolidate databases into one, Disaster Recovery, Standby Database Preferred for homogeneous migrations
AWS Schema Conversion Tool	Migrate database schema (views, stored procedures ...) to compatible targets Features: SCT assessment report, Update source code (update embedded SQL in code), Fan-in (multiple sources - single target),Fan-out (single source - multiple targets) Preferred for schema conversion, large data warehouse workloads (RedShift)
AWS DataSync	Transfer from on-premise file storage to S3, EFS or FSx for Windows (Integration) Migrate data using DataSync and use AWS Storage Gateway for ongoing updates from on-premises applications

Data Analytics

Big Data - Terminology and Evolution

In28 Minutes

- **3Vs** of Big Data
  - **Volume**: Terabytes to Petabytes to Exabytes
  - **Variety**: Structured, Semi structured, Unstructured
  - **Velocity**: Batch, Streaming ..
- **Terminology**: Data warehouse vs Data lake
  - **Data warehouse**: PBs of Storage + Compute (Typically)
    - Data stored in a format ready for specific analysis! (processed data)
      - Examples: Teradata, BigQuery(GCP), Redshift(AWS), Azure Synapse Analytics
    - Typically uses specialized hardware
  - **Data lake**: Typically retains all raw data (compressed)
    - Typically object storage is used as data lake
      - Amazon S3, Google Cloud Storage, Azure Data Lake Storage Gen2 etc..
    - Flexibility while saving cost
    - Perform ad-hoc analysis on demand
    - Analytics & intelligence services (even data warehouses) can directly read from data lake
      - Azure Synapse Analytics, BigQuery(GCP), Redshift Spectrum(AWS), Amazon Athena etc..



Big Data & Datawarehousing in AWS

In28 Minutes

Service	Scenario
Amazon Redshift	Run complex queries (SQL) against data warehouse - housing structured and unstructured data pulled in from a variety of sources
Amazon EMR	Managed Hadoop. Large scale data processing with high customization (machine learning, graph analytics) Important tools in Hadoop ecosystem are natively supported (Pig, Hive, Spark or Presto)
Amazon S3	Can be used as a Data Lake
Lake Formation	Makes it easy to set up a secure data lake
Amazon Redshift Spectrum	Run queries directly against S3 without worrying about loading entire data from S3 into a data warehouse. Scale compute and storage independently.
Amazon Athena	Quick ad-hoc queries without provisioning a compute cluster (serverless) Amazon Redshift Spectrum is recommended if you are executing queries frequently against structured data

Streaming Data - Simple Solutions

- How to process continuous streaming data from application logs or user actions from social media applications?
  - Characteristics of streaming data: Continuously generated, Small pieces of data which are Sequenced - mostly associated with time
- Option: **S3 Notifications**
  - Send notifications to SNS, SQS, trigger lambdas on
    - creation, deletion or update of an S3 object
  - Setup at bucket level (Use prefix and suffix)
  - Cost efficient for simple use cases (S3 notification -> Lambda)
    - Almost negligible cost (storage for file + invocation)
- Option: **DynamoDB Streams** (DynamoDB > Stream > Lambda)
  - Events from DynamoDB buffered in a stream (real-time sequenced)
  - Can be enabled or disabled
  - Use case - Send email when user registers

Amazon Kinesis

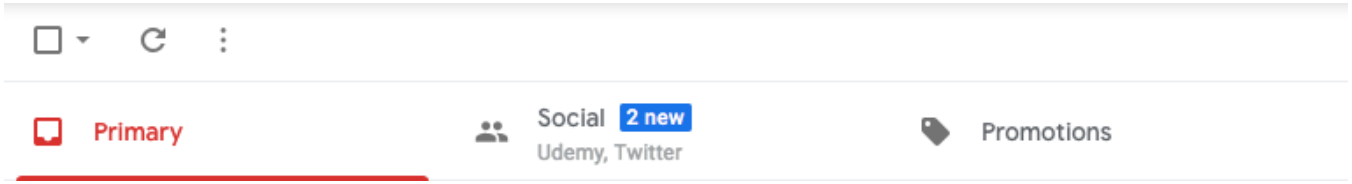
In28 Minutes

- Handle streaming data (NOT recommended for ETL Batch Jobs)
- **Amazon Kinesis Data Streams** (Alternative for Kafka)
  - (ALTERNATIVE)AWS MSK - Fully managed service for Apache Kafka
  - Limitless Real time stream processing with Sub second latency
    - Supports multiple clients (Each client can track their stream position)
    - Retain and replay data (max 7 days & default 1 day)
- **Amazon Kinesis Firehose**: Data ingestion for streaming data
  - Receive > Process (transform - Lambda, compress, encrypt) > Store (S3, Elasticsearch, Redshift and Splunk)
    - Use existing analytics tools based on S3, Redshift and Elasticsearch
- **Amazon Kinesis Analytics**: Continuously analyze streaming data
  - Run SQL queries and write Java apps (find active users in last 5 minutes)
- **Amazon Kinesis Video Streams**: Monitor video streams from web-cams
  - Integrate with machine learning to get intelligence (Examples: traffic lights, shopping malls)

Machine Learning

Artificial Intelligence - All around you

In 28 Minutes



- Self-driving cars
- Spam Filters
- Email Classification
- Fraud Detection

What is AI? (Oxford Dictionary)

In 28 Minutes

The theory and development of computer systems able to perform tasks normally requiring human intelligence, such as visual perception, speech recognition, decision-making, and translation between languages

Understanding Types of AI

In 28 Minutes

- **Strong artificial intelligence (or general AI)**: Intelligence of machine = Intelligence of human
  - A machine that can solve problems, learn, and plan for the future
  - An expert at everything (including learning to play all sports and games!)
  - Learns like a child, building on it's own experiences
  - We are far away from achieving this! (Estimates: few decades to never)
- **Narrow AI (or weak AI)**: Focuses on specific task
  - Examples: Self-driving cars and virtual assistants
  - **Machine learning**: Learn from data (examples)

Tags:

Water 100% confidence Sky 100% confidence
Lake 95% confidence Outdoor 95% confidence
Skyscraper 89% confidence Reflection 61% confidence
Overlooking 33% confidence Day 12% confidence

Description:

a city skyline with water 27% confidence

Racy Content: Adult Content:

False 75% confidence False 78% confidence

Exploring Machine Learning vs Traditional Programming

In28 Minutes

- **Traditional Programming**: Based on Rules
  - IF this DO that
  - Example: Predict price of a home
    - Design an algorithm taking all factors into consideration:
      - Location, Home size, Age, Condition, Market, Economy etc
- **Machine Learning**: Learning from Examples (NOT Rules)
  - Give millions of examples
  - Create a Model
  - Use the model to make predictions!
- **Challenges**:
  - No of examples needed
  - Availability of skilled personnel
  - Complexity in implementing MLOps

Home size (Square Yds)	Age	Condition (1-10)	Price $$$
300	10	5	XYZ
200	15	9	ABC
250	1	10	DEF
150	2	34	GHI

Machine Learning - 3 Approaches

In 28 Minutes

- **Use Pre-Trained Models**
  - Get intelligence from text, images, audio, video
  - Amazon Comprehend, Amazon Rekognition, ...
- **Build simple models**: Without needing data scientists
  - Limited/no-code experience
  - Example: Amazon SageMaker Auto ML
- **Build complex models**: Using data scientists and team
  - Build Your Own ML Models from ZERO (code-experienced)
  - Example: Amazon SageMaker

Tags:

Water 100% confidence Sky 100% confidence
Lake 95% confidence Outdoor 95% confidence
Skyscraper 89% confidence Reflection 61% confidence
Overlooking 33% confidence Day 12% confidence

Description:

a city skyline with water 27% confidence

Racy Content: False 75% confidence

Adult Content: False 78% confidence

Pre-Trained Models in AWS

- **Amazon Comprehend** - Analyze Unstructured Text
- **Amazon Rekognition** - Search and Analyze Images and Videos
- **Amazon Transcribe** - Powerful Speech Recognition
- **Amazon Polly** - Turn Text into Lifelike Speech
- **Amazon Translate** - Powerful Neural Machine Translation
- **Amazon Personalize** - Amazon Personalize helps you easily add real-time recommendations to your apps
- **Amazon Fraud Detector** - Detect more online fraud faster using machine learning

Tags:

Water 100% confidence Sky 100% confidence
Lake 95% confidence Outdoor 95% confidence
Skyscraper 89% confidence Reflection 61% confidence
Overlooking 33% confidence Day 12% confidence

Description:

a city skyline with water 27% confidence

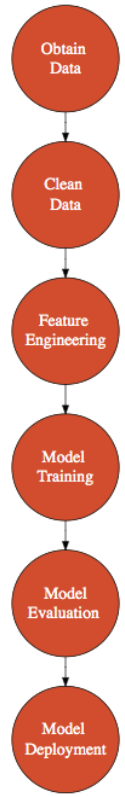
Racy Content: False 75% confidence

Adult Content: False 78% confidence

Creating Machine Learning Models - Steps

In28 Minutes

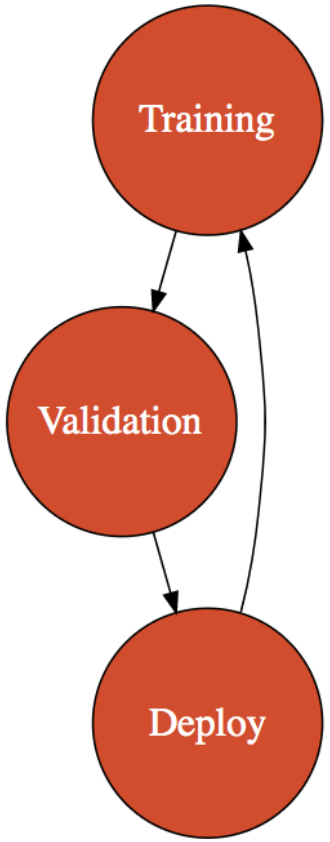
- **1:** Obtain Data
- **2:** Clean Data
- **3:** Feature Engineering: Identify Features and Label
- **4:** Create a Model using the Dataset and the ML algorithm
- **5:** Evaluate the accuracy of the model
- **6:** Deploy the model for use



Amazon SageMaker

In 28 Minutes

- **Amazon SageMaker**: Simplifies creation of your models
  - Manage data, code, compute, models etc
  - Prepare data
  - Train models
  - Publish models
  - Monitor models
- **Multiple options to create models**
  - **AutoML**: Build custom models with minimum ML expertise
  - **Build Your Own Models**: Data Scientists
    - Support for deep-learning frameworks such as TensorFlow, Apache MXNet, PyTorch, and more (use them within the built-in containers)
    - Data and compute management, pipelines



Well Architected Framework

Well Architected Framework

In28 Minutes

- Helps cloud architects **build application infrastructure** that is:
  - Secure
  - High-performing
  - Resilient and
  - Efficient
- **Six Pillars**
  - Operational Excellence
  - Security
  - Reliability
  - Performance Efficiency
  - Cost Optimization
  - Sustainability

Operational Excellence Pillar

In28 Minutes

- **Avoid/Minimize** effort and problems with:
  - Provisioning servers, Deployment, Monitoring and Support
- **Recommendations**:
  - **Use Managed Services**: No worry about managing servers, availability etc
  - **Go serverless**: Prefer Lambda to EC2!
  - **Use Infrastructure As Code**: Terraform, Pulumi, Cloud Formation
  - **Implement CI/CD** to find problems early: CodePipeline, CodeBuild, CodeDeploy
  - Perform frequent, small reversible changes
- **Recommended Approach**:
  - **Prepare for failure**: Game days, Disaster recovery exercises
  - **Operate**: Gather Data and Metrics
    - CloudWatch (Logs agent), Config, Config Rules, CloudTrail, VPC Flow Logs and X-Ray (tracing)
  - **Evolve**: Get intelligence (Ex:Use Amazon Elasticsearch to analyze your logs)


AWS Lambda


CloudFormation


Codepipeline


AWS Config


Cloudwatch

Security Pillar

- **Principle of least privilege for least time**
  - Use temporary credentials when possible (IAM roles, Instance profiles)
  - Enforce MFA and strong password practices
  - Rotate credentials regularly
- **Security in Depth** - Apply security in all layers
  - VPCs and Private Subnets (Security Groups and Network Access Control List)
  - Use hardened EC2 AMIs(golden image) - Automate patches (OS, Software..)
  - Use CloudFront with AWS Shield for DDoS mitigation
  - Use WAF with CloudFront and ALB (Protect web apps from XSS, SQL injection etc)
  - **Use Infrastructure As Code** (Automate provisioning infra that adheres to security policies)

Security Pillar - 2 - Data

- Protect **Data at Rest**
  - Enable encryption - KMS & Cloud HSM (Rotate encryption keys)
  - Enable versioning (when available)
- Protect **Data in Transit**
  - Data coming in and going out of AWS
  - By default, all AWS API use HTTPS/SSL
  - You can also choose to perform client side encryption for additional security
  - Ensure your data stays in AWS network when possible (VPC Endpoints and AWS PrivateLink)

Security Pillar - 3 - Active Monitoring

In 28 Minutes

- **Detect Threats**: Actively monitor for security issues
  - Monitor CloudWatch Logs
  - Use Amazon GuardDuty to detect threats and continuously monitor for malicious behavior
  - Use AWS Organization to centralize security policies for multiple AWS accounts

Reliability Pillar

In28 Minutes

- **Reliability**: Ability to recover from infra and app issues
  - Adapt to changing demands in load
- **Best Practices**
  - **Automate recovery from failure**
    - Health checks and Auto scaling
    - Managed services like RDS can automatically switch to standby
  - **Scale horizontally** (Reduces impact of single failure)
  - **Maintain Redundancy**
    - Multiple Direct Connect connections
    - Multiple Regions and Availability Zones
  - **Prefer serverless** architectures
  - Prefer **loosely coupled** architectures: SQS, SNS
  - **Adhere to** Distributed System Best Practices
    - Use Amazon API Gateway for throttling requests
    - AWS SDK provides retry with exponential backoff


AWS Lambda


Amazon SQS


Amazon SNS


API Gateway


AutoScaling

Prefer Loosely Coupled Architectures

In28 Minutes

- **ELB**
  - Works in tandem with AWS auto scaling
- **Amazon SQS**
  - Polling mechanism
- **Amazon SNS**
  - Publish subscribe pattern
  - Bulk notifications and Mobile push support
- **Amazon Kinesis**
  - Handle event streams
  - Multiple clients
  - Each client can track their stream position

Troubleshooting on AWS - Quick Review

In 28 Minutes

Option	Details	When to Use
Amazon S3 Server Access Logs	S3 data request details - request type, the resources requested, and the date and time of request	Troubleshoot bucket access issues and data requests
Amazon ELB Access Logs	Client's IP address, latencies, and server responses	Analyze traffic patterns and troubleshoot network issues
Amazon VPC Flow Logs	Monitor network traffic	Troubleshoot network connectivity and security issues

Troubleshooting on AWS - Quick Review - 2

In 28 Minutes

Option	Details	When to Use
Amazon CloudWatch	Monitor metrics from AWS resources	Monitoring
Amazon CloudWatch Logs	Store and Analyze log data from Amazon EC2 instances and on-premises servers	Debugging application issues and Monitoring
AWS Config	AWS resource inventory. History. Rules.	Inventory and History
Amazon CloudTrail	History of AWS API calls made via AWS Management Console, AWS CLI, AWS SDKs etc.	Auditing and troubleshooting. Determine who did what, when, and from where.

Performance Efficiency Pillar: Meet needs with min. resources

In 28 Minutes

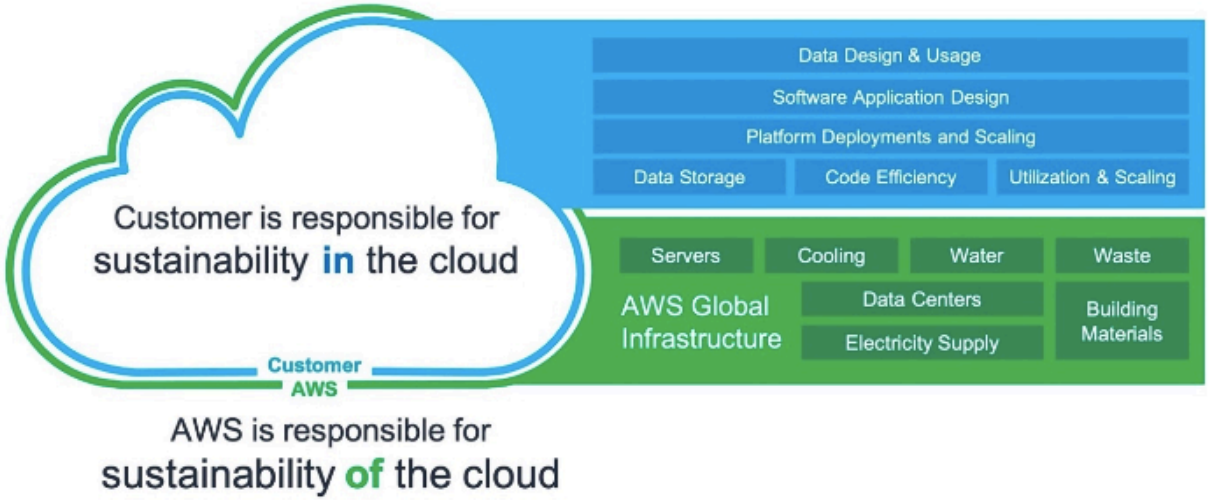
- Continue being efficient as demand and technology evolves
- **Best Practices**:
  - Use Managed Services (Avoid Undifferentiated Heavy Lifting)
  - Go Serverless (Lower transactional costs and less operational burden)
  - Experiment (Cloud makes it easy to experiment)
  - Monitor Performance (Trigger CloudWatch alarms - Perform actions with SQS and Lambda)
- **Choose the right solution**:
  - **Compute**: EC2 instances vs Lambda vs Containers
  - **Storage**: Block, File, Object
  - **Database**: RDS vs DynamoDB vs RedShift ..
  - **Caching**: ElastiCache vs CloudFront vs DAX vs Read Replicas
  - **Network**: CloudFront, Global Accelerator, Route 53, Placement Groups, VPC endpoints, Direct Connect
  - **Use product specific features**: Enhanced Networking, S3 Transfer Acceleration, EBS optimized instances

Cost Optimization Pillar: Run systems at lowest cost

In 28 Minutes

- **Best Practices**
  - **Match supply and demand**
    - Implement Auto Scaling
    - Stop Dev/Test resources when you don't need them
    - Go Serverless
  - **Track your expenditure** (Use tags on resources)
    - Cost Explorer to track and analyze your spend
    - AWS Budgets to trigger alerts
- **Choose Cost-Effective Solutions**
  - **Right-Sizing** : Analyze 5 large servers vs 10 small servers
    - Use CloudWatch (monitoring) and Trusted Advisor (recommendations) to right size your resources
  - Email server vs Managed email service (charged per email)
  - On-Demand vs Reserved vs Spot instances
  - **Avoid expensive software** : MySQL vs Aurora vs Oracle
  - Optimize data transfer costs using AWS Direct Connect and Amazon CloudFront

Sustainability Pillar: Focus on environmental sustainability



- **Sustainability** - "development that meets the needs of the present without compromising the ability of future generations to meet their own needs."
  - **Shared Responsibility**
    - AWS is responsible for sustainability **of the cloud**
      - Efficient power and cooling technology, operate energy efficient server populations, achieve high server utilization rates, ...
    - Customers are responsible for sustainability **in the cloud**
      - Choosing efficient programming language, Using modern algorithms, Efficient data storage, Right sizing infrastructure ...

Quick Review

Compute and Containers

In28 Minutes

Service	Description
EC2	Virtual Servers in the Cloud
Elastic Beanstalk	PaaS to Run and Manage Web Apps
Elastic Container Service	Highly secure, reliable, and scalable way to run containers
Elastic Kubernetes Service	The most trusted way to start, run, and scale Kubernetes
Lambda	Run Code without Thinking about Servers
Elastic Container Registry	Fully-managed Docker container registry

Storage

Service	Description
EBS	Block Storage for EC2 instances
EFS	Managed File Storage for EC2
FSx	Fully managed third-party file systems optimized for a variety of workloads
S3	Scalable Storage in the Cloud
S3 Glacier	Archive Storage in the Cloud
Storage Gateway	Hybrid Storage Integration

Database

Service	Description
RDS	Managed Relational Database Service
DynamoDB	Managed NoSQL Database
Amazon DocumentDB	Fully-managed MongoDB-compatible database service
ElastiCache	In-Memory Cache
Amazon Keyspaces	Serverless Cassandra-compatible database
Neptune	Fast, reliable graph database built for the cloud

DevOps

In 28 Minutes

Service	Description
CodeCommit	Store Code in Private Git Repositories
CodeBuild	Build and Test Code
CodeDeploy	Automate Code Deployments
CodePipeline	Release Software using Continuous Delivery
CloudFormation	Create and Manage Resources with Templates
CDK	Create and Manage Resources with Code
SAM	Create and Manage Serverless Resources
OpsWorks	Configuration Management with Chef and Puppet
CloudWatch	Monitor Resources and Applications
X-Ray	Analyze and Debug Your Applications

Analytics

Service	Description
Amazon Redshift	Fast, Simple, Cost-Effective Data Warehousing
EMR	Managed Hadoop Framework
Kinesis	Work with Real-Time Streaming Data
MSK	Fully managed, highly available, and secure service for Apache Kafka
Athena	Query Data in S3 using SQL
AWS Lake Formation	Makes it easy to set up a secure data lake

Networking & Content Delivery

Service	Description
VPC	Isolated Cloud Resources
API Gateway	Build, Deploy and Manage APIs
Direct Connect	Dedicated Network Connection to AWS
CloudFront	Global Content Delivery Network
Route 53	Scalable DNS and Domain Name Registration
Global Accelerator	Improve your application's availability and performance using the AWS Global Network

Security, Identity, & Compliance

In28 Minutes

Service	Description
IAM	Manage access to AWS resources
Cognito	Implement Authentication and Authorization for Your Web and Mobile Apps
Certificate Manager	Provision, Manage, and Deploy SSL/TLS Certificates
Key Management Service	Securely Generate and Manage AWS Encryption Keys
CloudHSM	Managed Hardware Security Modules in the Cloud
GuardDuty	Intelligent Threat Detection to Protect Your AWS Accounts and Workloads
Amazon Macie	Amazon Macie classifies and secures your business-critical content.
Secrets Manager	Easily rotate, manage, and retrieve secrets throughout their lifecycle
WAF & Shield	Protects Against DDoS Attacks and Malicious Web Traffic
Detective	Investigate and analyze potential security issues

Management & Governance

Service	Description
CloudTrail	Track User Activity and API Usage
Config	Track Resource Inventory and Changes
AWS Organizations	Central governance and management across AWS accounts.
Trusted Advisor	Optimize Performance and Security
AWS Well-Architected Tool	Use AWS Well-Architected Tool to learn best practices, measure, and improve your workloads
AWS AppConfig	With AWS AppConfig, make updates to application configurations at runtime.
AWS Compute Optimizer	Recommend optimal AWS Compute resources for your workloads
Control Tower	The easiest way to set up and govern a secure, compliant multi-account environment

Get Ready

Recommended Resources

Title	Link
AWS Architecture Center	*https://aws.amazon.com/architecture/*
AWS FAQs	*https://aws.amazon.com/faqs/* (EC2, S3, VPC, RDS, SQS etc)

You are all set!

Let's clap for you!

- You have a lot of patience! Congratulations
- You have put your best foot forward to do well at your AWS interview
- Make sure you prepare well and
- Good Luck!

Do Not Forget!

In 28 Minutes

- Recommend the course to your friends!
  - Do not forget to review!
- Your Success = My Success
  - Share your success story with me on LinkedIn (Ranga Karanam)
  - Share your success story and lessons learnt in Q&A with other learners!

What Next?

FASTEST ROADMAPS

in28minutes.com

In28 Minutes

- Google Cloud Certifications
- Azure Certifications
- AWS Certifications
- DevOps
- Java Full Stack
- Java Microservices