AWS CERTIFIED

SOLUTIONS ARCHITECT

#NOTES

- Scheduled scaling policy → Auto Scaling group
- STS (Security Token Service)
- check PuTTY (PEM → PPK)
- Amazon Aurora / Redshift *Check
- AWS Kinesis *Check
- AWS Lambda *Check - OLAP / OLPT
- AWS Redshift *Check
- AWS DynamoDB - Redshift Cluster
- AWS Active Directory - AWS X-Ray
- CloudWatch + ELB + SNS
- AWS MQ / SQS / SNS

THE WELL ARCHITECTED FRAMEWORK

- RELIABILITY
- PERFORMANCE EFFICIENCY
- SECURITY
- COST OPTIMIZATION
- OPERATIONAL EXCELLENCE

SOLUTIONS ARCHITECT

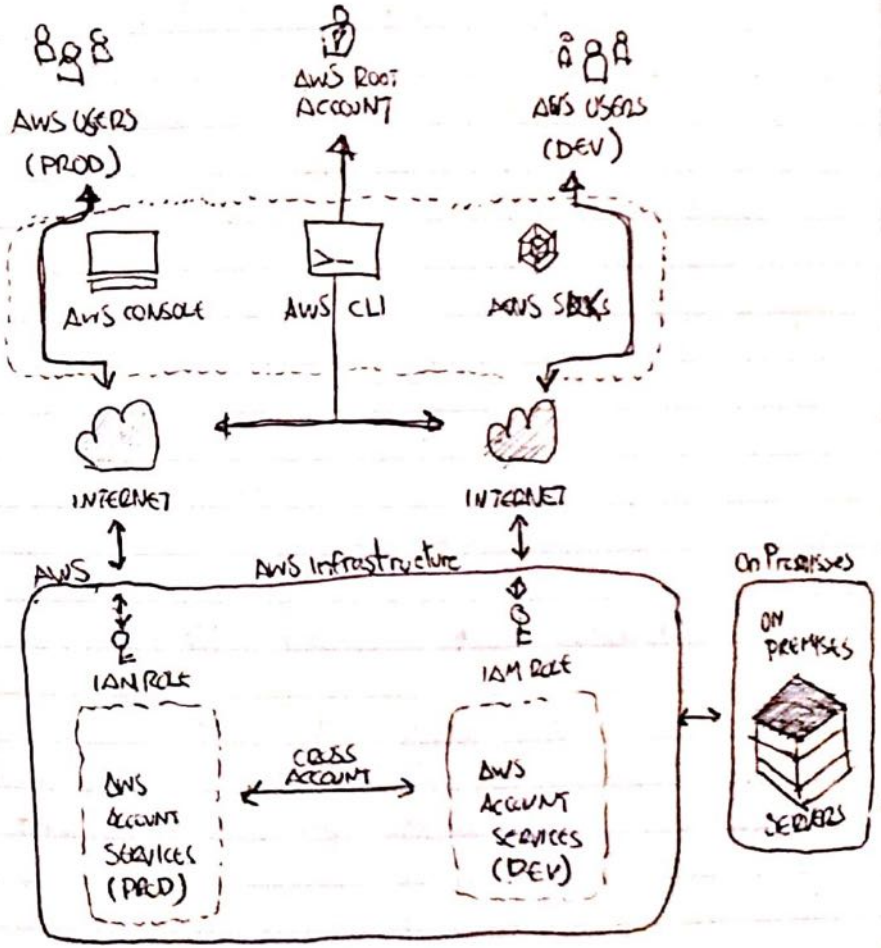
- Responsible for interpreting customer requirements and defining a solution following architectural best practices. and design principles

- IT Professional responsible of overseeing a company's cloud computing strategy

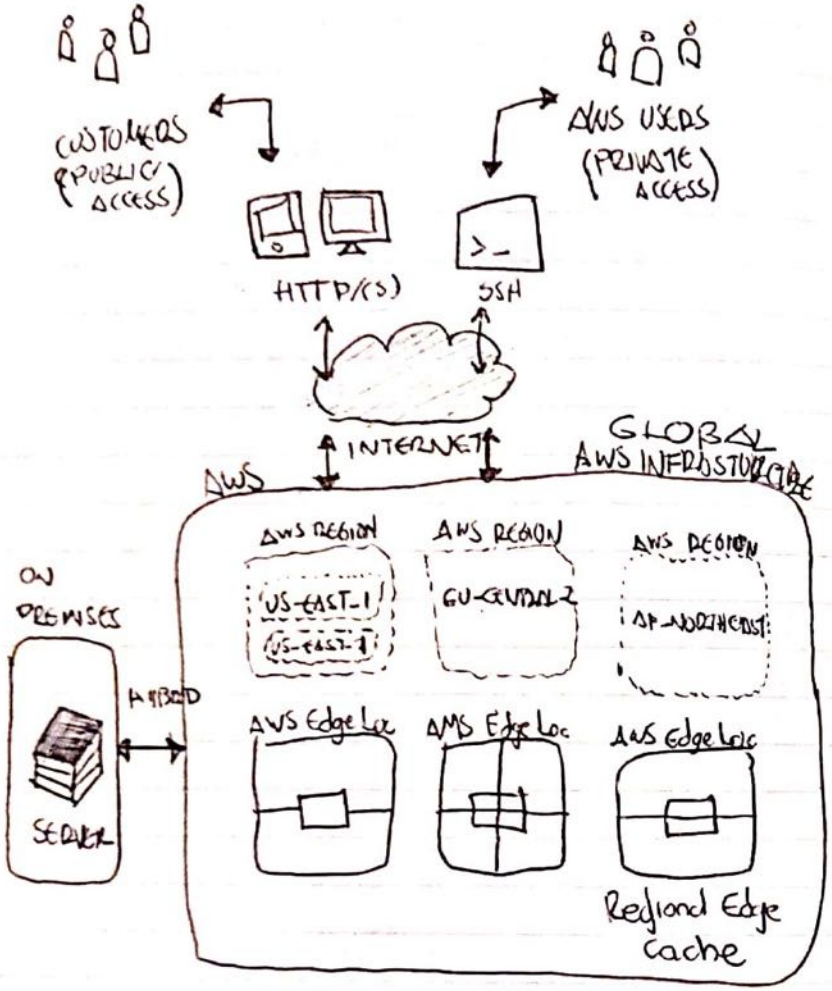
AWS WELL ARCHITECTED FRAMEWORK

- RELIABILITY
- COST OPTIMIZATION
- OPERATIONAL EXCELLENCE
- SECURITY
- PERFORMANCE EFFICIENCY

ACCOUNT & SERVICES LAYER

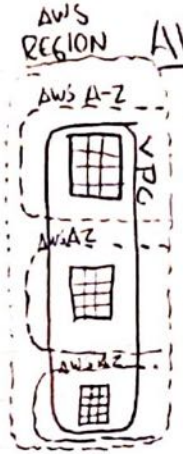


AWS PHYSICAL & NETWORKING LAYER



AWS ACCOUNT CONNECTION TOOLS

- Console: Web based UI. API Calls
- CLI: Text based. API Calls; API Key Conf.
- SDK: Service based. API Calls: API Ky Conf



AWS A-Z

AWS REGION
us-east-1

AWS A-Z
us-east-1a
us-east-1b
us-east-1c

AWS EDGE LOCATIONS

- Points of Presence
  - ↳ Route 53 (DNS)
  - ↳ CloudFront (CDN)
  - ↳ Shield (DDoS)
  - ↳ WAF (Firewall)
  - ↳ SS (Transfer)
  - ↳ Lambda@Edge (Serverless)
  - ↳ API Gateway (Endpoints)

AWS REGIONAL EDGE CACHE

(REC) are used by CloudFront to offload your origin by caching content that has been ejected from an Edge Location.

PLP: PRINCIPLE OF LEAST PRIVILEDGE

Security best practice by only allowing for users to have the exact amount of access they need for the job

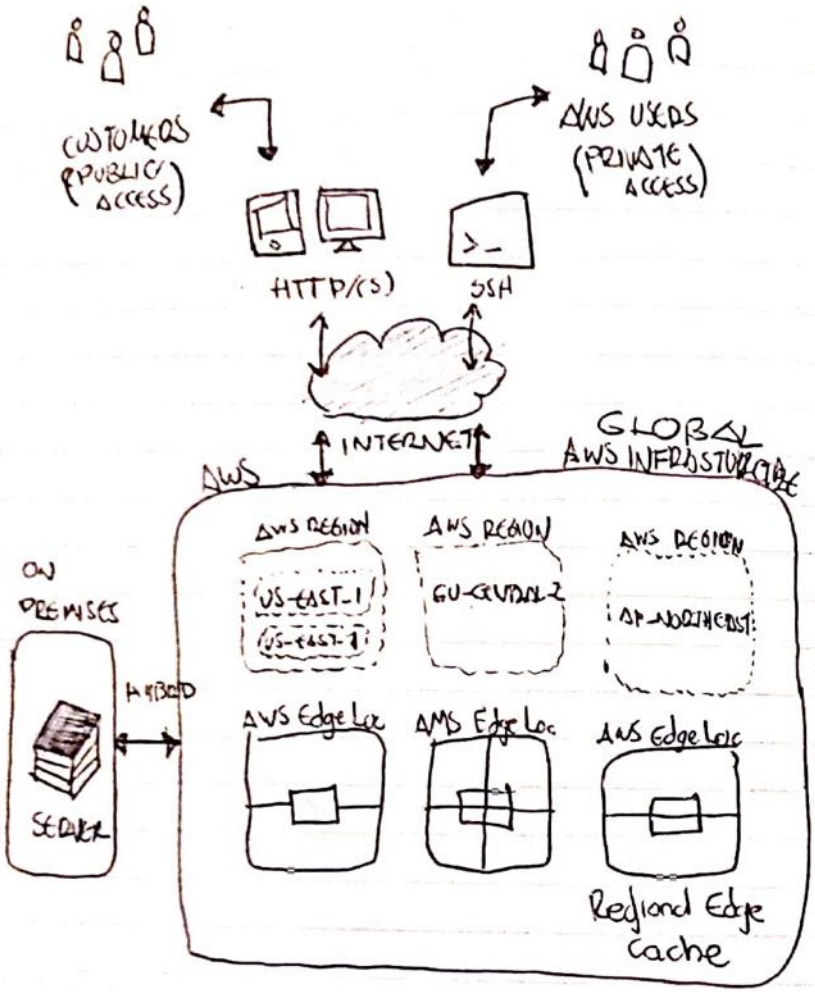
CSA: CERTIFIED SOLUTIONS ARCHITECT

AWS CSA TERMINOLOGY

- **High Availability**: Refers to architecture that continues to remain available to uses in the event of a component or system failure. Multi A-Z archs. allow high availability in the event of A-Z outage.

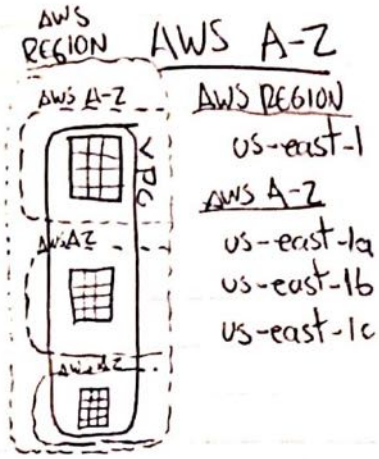
- **Fault Tolerant**: Architecture not only remains available during an outage, but also soffer no degradation in performance. F-T archs usually require redundancy and should

AWS PHYSICAL & NETWORKING LAYER



AWS ACCOUNT CONNECTION TOOLS

- Console: Web based UI, API Calls
- CLI: Text based. API Calls; API Key Conf.
- SDK: Service based. API Calls: API Ky Conf



AWS A-Z

AWS REGION
us-east-1

AWS A-Z
us-east-1a
us-east-1b
us-east-1c

AWS EDGE LOCATIONS

- Points of Presence
  - ↳ Route 53 (DNS)
  - ↳ CloudFront (CDN)
  - ↳ Shield (DDoS)
  - ↳ WAF (Firewall)
  - ↳ SS (Transfer)
  - ↳ Lambda@Edge (Serverless)
  - ↳ API Gateway (Endpoints)

AWS REGIONAL EDGE CACHE

(REC) are used by CloudFront to offload your origin by caching content that has been ejected from an Edge Location.

PLP: PRINCIPLE OF LEAST PRIVILEDGE

Security best practice by only allowing for users to have the exact amount of access they need for the job

CSA: CERTIFIED SOLUTIONS ARCHITECT

AWS CSA TERMINOLOGY

- High Availability: Refers to architecture that continues to remain available to users in the event of a component or system failure. Multi A-Z archs. allow high availability in the event of A-Z outage.

- Fault Tolerant: Architecture not only remains available during an outage, but also suffer no degradation in performance. F-T archs usually require redundancy and should

be traded with cost concerns.

- Scalability: Refers to the ability to easily increase size and capacity in a cost effective way. Scaling can be vertical (capacity on a single instance) or horizontal (+ or -, number of instances)

- Elasticity: Refers to the ease of a system's ability to change and adapt. Ex. Automatic scaling in or out, updating firewalls, remapp IP

- Cost Efficient: Trade-offs required to make a system as inexpensive as possible while meeting functional / business requirements.

- Secure: Refers to following proper security guidelines and practices to secure a system at every layer.

AWS SHARED RESPONSIBILITY MODEL

- AWS (Security **of** the cloud):
  - ↳ Responsible of global infrastructure
  - ↳ Facilities, Physical access, Storage
  - ↳ Network, Virtualization

- Customer (Security **in** the cloud)
  - ↳ Responsible of your Virtual env, data, apps
  - ↳ AMIs, OSes, Apps, SGs, WAFs, Data(in transit/rest)
  - ↳ Credentials, Policies, Intrusion, (detection/prevention)

AWS Identity & Access Mgmt

- IAM is used to create users and assign roles to those users
- Groups inherit policies to users
- USERS, ROLES, POLICIES, GROUPS, API Keys
- IAM can use a 3rd party provider, LDAP such as Active Directory

  - ↳ Delete Root access Keys
  - ↳ Activate MFA
  - ↳ Strong Password Policy
  - ↳ Credential Report

  IAM is global but policies can condition groups or roles for A-Z restriction

- Policy is a JSON document that formally states one or more permission.
  - ▶ Deny > Allow for some permissions.
  - Can create custom permissions policies.
  - Conditions must be met in orther to access the resource.

- User By default, users have implicit "deny" on all AWS resources, on creation. Permanent
  - → ACCESS KEYS
  - → ACCESS SECRET KEYS

  Inline policies are special permissions

- ROLE IAM Entity. Temporary added to certain users credentials. Federated users (LDAP/Active Dir) 15min-12hours. Another entity "assumes"

- **Role** Permissions Policy + Trust Policy
  (Grants Permission) (Which Entity)
  An EC2 instance can have only one role at a time

IAM Role Entities

- AWS Service
- AWS Account
- Web Identity
- SAML 2.0 Identification (federation)

API KEYS: Used to sign programmatically requests in/of AWS resources. These are permanent Should be rotated. Access keys come from a service called STS. (Simple Token Service) 15mn ↔ ~~12~~hrs.
STS is for temp) credentials from IAM

IDENTITY
FEDERATION: LDAP? Active Directory? AWS SAML is an Id federation service.
- SAML (Security Assertion Markup Lang)
- Amazon Cognito. / OpenID
- Web Identity Federation

AWS ORGANIZATIONS

- IAM Policy Management
  - Account Group
  - Create Accounts Programmatically
- Consolidate Billing
  - Manage Payment Methods

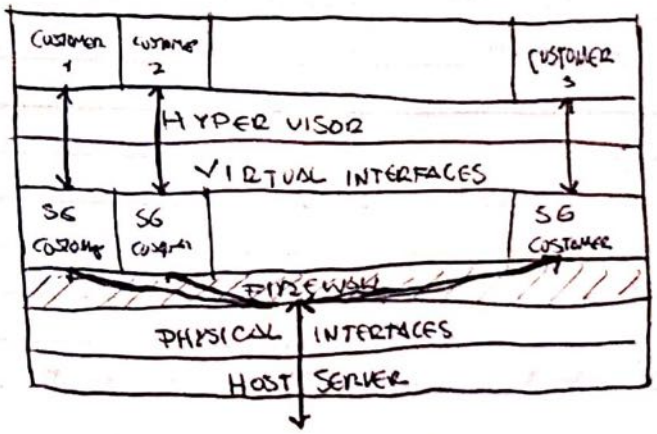
AWS EC2 INSTANCES

- **AMI** : AMAZON MACHINE IMAGE
  - ↳ OPERATING SYSTEM
  - ↳ SOFWARE PACKAGES
  - ↳ (ROOT STORAGE

- **EC2**:
  - AMAZON EC2 is scalable virtual servers in the cloud
  - EC2 VS is known as "instance", type & sizes
  - Linux or Windows

  - Designed to mimic on prem config, with the ~~the~~ ability to be commissioned and decommissioned on demand for easy scalability and elasticity
  - EC2 are comprised of
    - AMI : Amazon Machine Image
    - ~~Type~~ Instance Type: Size and Hardware
    - Network Interface : Public, Private, Elastic IP
    - Storage : Hard drive

- **Hypervisor**



- **VIRTUALIZATION**: The process of creating a "virtual" version of something rather of the version of that thing. In AWS EC2, virtualization is create a "portion" of a server
  - Virtualization for EC2 is run using Xen Hypervisor
  - Physical servers and Xen is handled by AWS

- **LINUX AMI VIRTUALIZATION:**
  - ↳ HVM AMIs (Hardware)
  - ↳ PV AMIs (Paravirtual)

- **EC2 Instance Type**

TYPE	CATEGORY	DESCRIPTION	USE CASE
M5	GENERAL	ENHANCED COMPUTE, MEM	MID DB
C5	COMPUTE	ADVANCED CPU	MODELING
H1	STORAGE	LOCAL HDD	MAP REDUCE
R4	MEMORY	MORE RAM PER $	IN MEMORY CACHE
X1	MEMORY	TERAS OF RAM / SSD	IN MEM DB
I3	IO	LOCAL SSD, HIGH IOPS	NOSQL DB
G3	GPU GRAPHICS	GPUS VIDEO	3D RENDER
P3	GPU COMPUTE	GPUS TENSORS	ML, AI, DS
F1	ACCELERATED	FPGA, CUSTOM	GENOMICS
T2	BURSTABLE	SHARED, LOW COST	WEB

- **EC2 IP ADDRESSES (EIP)**

- PRIVATE IP
  - ↳ All EC2 have or private IP
  - ↳ Private IP is used in the VPC / Subnet

- Public IP
  - ↳ Enable or Autoassign a public IP (IPv4)
  - ↳ Necesary if you want the EC2 to have direct connection to any resource in the open internet

- Elastic IP
  - ↳ A static IPv4 designed for dynamic cloud computing
  - ↳ EIP can be persistent or static.
  - ↳ Ideal for Route53 (DNS config) ideal for masking Instance Failure
  - ↳ Attach / Detach from EC2 instances.

- Monitoring: Enable CloudWatch

- Tenancy: Shared, Dedicated, Dedicated Host, Used when there are some business or legal compliance rules.

- User Data: Section to include custom scripts.

curl http://169.254.169.254/latest/user-data
/meta-data

To get info about the EC2 Instance

- Storage
  - ↳ Instance Store: HDD, Bootvolumes, Secondary
  - ↳ EBS: Outside EC2, Bootvolumes, over the network
  - ↳ EFS: Multy instance File systems.
  - ↳ S3: Simple Storage

- EBS Elastic Block Storage
  - ↳ EBS are persistent, beyond the EC2 instance
  - ↳ EBS is a NAS, can be attached to various EC2 is
  - ↳ Snapshots are backups
  - ↳ 16 Tebibites max size
  - ↳ IOPS:
    - AWS measures IOPS in 256 kB chunks
  - ↳ The type of EBS volume greatly influences I/O
  - ↳ Out of a 1000 EBS, 1 or 2 will fail in a year (.02%-.01% failure rate)
  - ↳ EBS is lazy loading, so to avoid degradation the volume should be initialized, meaning all data should be read beforehand, to populate it,
    5 IOPS per GiB
  - (SSD)
    - General Purpose: Low IOPS, 1GiB - 16TiB
    - Provisioned IOPS: High IOPS, ≤80,000, 4GiB - 16TiB
  - (HDD)
    - Throughput Optimized (st1): 500 Mb/s, 500GiB - 16TiB
    - Cold HDD (sc1): No boot vol, 250 Mb/s, 500GiB - 16TiB

- Instance Store
  - ↳ Physically attached, very high IOPS (3M Read, 1.4M Write)
  - ↳ Ephemeral data, lost on restart
  - ↳ Suitable for swap, caches, replicated data
- Tags
  Suitable to categorize and manage instances

- EBS Volume Types

- Security Group: A set of firewall rules to allow or deny traffic to your instance.
  - ↳ Associate a keypair (PKI) public, private to enable ssh connections

- EBS Snapshots:
  - ↳ Point-in-time backups of EBS volumes, stored in S3
  - ↳ Incremental in nature
  - ↳ Snapshots can be used to create AMIs
  - ↳ Frequent snapshots increases data durability, so highly recommended
  - ↳ Stop writes issuing snapshot command for data consistency
    - fsfreeze
    - Stop instance if boot volume
  - ↳ A snapshot only stores changes since most recent snapshot, thus reducing cost.
  - ↳ However, if old snapshot deleted, blocks required to store other snapshots are retained

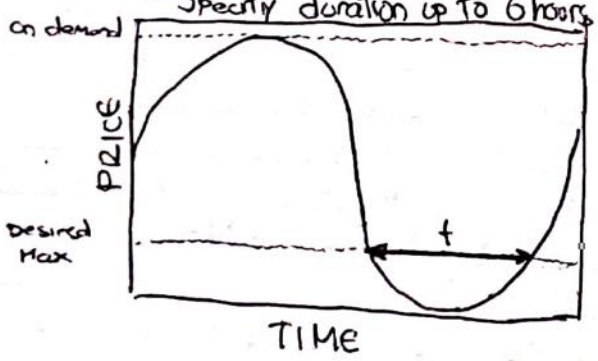
- EC2 Placement Groups
  - ↳ Used to reduce latency.
  - Cluster Placement Groups
    - Cluster of instances in same availability zone
    - Used for extremely low latency
    - AWS attempts to place as close as possible, maximize network throughput (up to 25Gbps). Use ENA (Network Adapter)
  - Spread Placement Group
    - Places instances on distinct host hardware

• EC2 Purchasing Options

- On Demand
  - ↳ Any instance type per hour/per second
  - ↳ The most expensive, most flexible

- Reserved
  - ↳ Allows to reserve for a set time period of 1 or 3 years
  - ↳ Significant price discount
    - Standard (up to ~75 discount)
    - Convertible (up to ~54 discount)
    - Scheduled (1 yr, 5-10% discount)

- Spot
  - ↳ AWS sells unused capacity for short amounts of time at a great substantial discount (90%)
  - ↳ Specify maximum price
  - ↳ Instances launch when spot price is <= thanmax
  - ↳ No charge if terminated in first hour, by AWS
  - ↳ Automatically terminate when spot price exceeds max
  - ↳ Spot Blocks
    - Specify duration up to 6 hours (30-50)% discount



• EFS (Elastic File System)

- ↳ NFS (Network File System) NFSv4
- ↳ Shared file system
- ↳ Storage capacity is elastic (Petabyte scale)
- ↳ Fully managed, only compatible with Linux
- ↳ Can be accessed by one or more EC2 instances
- ↳ EFS can be mounted in a on-premises server (EFS File Sync)
- ↳ Low latency, Petabyte scale
- ↳ Burst performance of 100 Mb/s
- ↳ POSIX permissions
- ↳ VPC, IAM
- ↳ Encrypt with KMS

AWS Virtual Private Cloud

• VPC: Is a virtual network that resembles a traditional physical network, operated in a traditional datacenter
- ↳ Private (No IGW) Subnets
- ↳ Public (IGW) Subnets
- ↳ VPC is housed within chosen AWS Region
- ↳ Ability to extend on-premises
- ↳ AWS Provides an internal DNS server, so you can have custom names for instances
  - Launch instances in subnet
  - Custom CIDR notation
    - Classless Inter Domain Range
  - Configure routes between subnets via route tables
  - Launch an Internet Gateway (IGW)
  - Layered network resources
  - Extend on prem network VPN/VPG Ipsec VPN

- Layered Security:
  - Instance level security Groups (firewall instance)
  - Network ACL (firewall network)

- VPC Limits
  - ↳ 5 VPC per Region
  - ↳ 5 IGW per Region (1 IGW per VPC)
  - ↳ 50 Customer gateways per Region
  - ↳ 50 VPN connections per Region
  - ↳ 200 Route tables per Region / 50 entries/table
  - ↳ 5 EIP (Elastic IPs)
  - ↳ 500 SG per VPC
  - ↳ 50 Rules per SG
  - ↳ 5 Security Groups per NI (Network Interface)

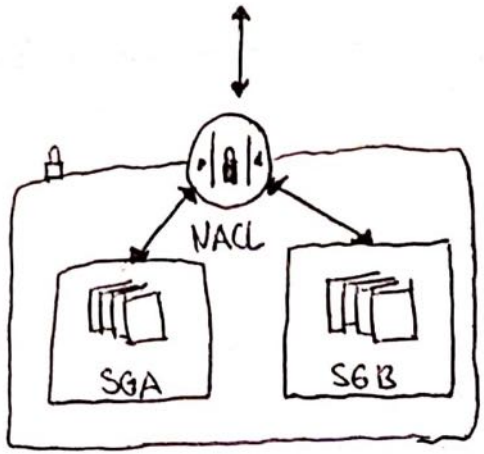
- VPC Network Routing
  - ↳ Every subnet in AWS VPC reserves 5 Private IPs from the CIDR notation for internal use - first 4 and the last 1

    10.0.0.0./16 - 65536
    10.0.0.0/24 - 254
    10.0.0.0/31 - 1
    10.0.0.0/0 - Entire Internet

  - ↳ IGW is a managed VPC component that allows communication between instances in your VPC and the Internet
  - ↳ 1IGW <> 1 VPC

- Route Table: A set of rules, called routes, that are used to determine where network traffic is directed
  - ↳ Destination: CIDR block range of the target
  - ↳ Target: A name identifier of where data is being routed to
  - ↳ By default, all subnets traffic is allowed to each other available subnet, within your VPC which is called local route
  - ↳ Local route is not modified
  - ↳ Unlike IGW, multiple "active" route tables in VPC
  - ↳ Non deletable if has dependencies

- Security
  - ↳ NACL (Network Access Control List)
    - Support all "allow" / "deny" rules for traffic flow
    - Stateless, so return traffic must be allowed through outbound rule
    - Process rules in number order
    - Rules evaluated in order, lowest rule number to highest rule number
      - if traffic is denied at a lower rule no. and allowed at a high rule no, Allow rule will be ignored and traffic denied.
    - Last rule in NACL is "catch all" deny rule
      - Unless stated, all is denied
  - ↳ NACL is optional layer of security for VPC and acts as a firewall of one or more subnets
  - ↳ Increment in order of 10s to be prone to change.



• Security Groups

↳ Similar to NACLs allow/deny traffic
↳ Instance level
↳ Allow/Deny rules work ~~to~~ different ACLs:
- SG support only allow rules
- Stateful: Return traffic requests are allowed
- All rules evaluated before allow

HIGH AVAILABILITY

• ELB (Elastic Load Balancer)

↳ Load Balancing is a practice to distribute incoming traffic.
↳ Cross-zone balancing
↳ Can be paired with Auto Scaling Group
↳ Own DNS record
↳ Can be public-facing or internal
↳ Health checks make ELB stop sending traffic to unhealthy instances
↳ Can reduce compute power using a SSL

• Classic ELB

- Simple balancing
- Routed evenly
- TCP, HTTP, HTTPS, SSL

OSI MODEL

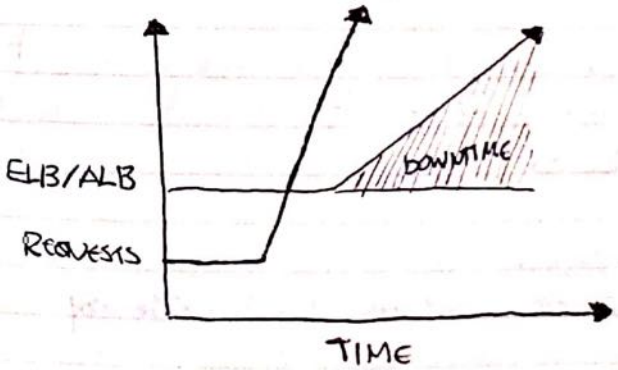
7	Application
6	Presentation
5	Session
4	Transport
3	Network
2	Datalink
1	Physical

• Application Load Balancer

- Content based rules
  - Host based
  - Path based
- Can balance multiple ports
- ECS, EKS, HTTPS, WebSockets Access Logs, Sticky Sessions, WAF

• Network Load Balancer

- Extreme Performance
- Doesn't need to scale to handle large traffic spikes
- Layer 4 TCP Load Balancing
- Static/Elastic IP Address per AZ
- IP Address ~~per~~ as Targets
- No SSL offloading



- SERVING TRAFFIC TO <> FROM PRIVATE SUBNET

- BASTION HOST
  - ↳ EC2 Instance that lives in a public subnet but used as gateway for traffic destined for instances in private subnets
  - ↳ "CRITICAL STRONG POINT"
  - ↳ Should have increased and extremely tighten security
  - ↳ "SSH" Access point into internal network without a VPN

- NAT GATEWAY:
  - ↳ NETWORK ADDRESS TRANSLATION
  - ↳ Designed to provide EC2 Instances with internet (Instances that live in a private subnet)
  - ↳ Will prevent for any host outside of the VPC from initiating a connection with instances associated with the NAT
  - ↳ Will allow traffic flow (incoming) if request initiated in an instance in a private subnet
  - ↳ A NAT is needed bc instances launched in a private subnet can't connect to the Internet
  - ↳ Must be launched in a public subnet
  - ↳ Must be part of the private subnets route tables

- NAT INSTANCES:
  - ↳ EC2 Instance used as NAT Gateway
  - ↳

- VPC ENDPOINTS
  - ↳ To connect VPC (Public/Private) with AWS Services
  - ↳ Gateway Endpoints
  - ↳ S3, DynamoDB
  - ↳ Better option than NAT Gateways or SSL endpoints
  - ↳ Must update route tables
  - Gateway Endpoint: (S3, DynamoDB)
  - Interface Endpoint: (CloudWatch Logs, KMS, CodeBuild, Kinesis, Service catalog)

- AUTOSCALING
  - ↳ Is a feature in AWS that automates the process of provisioning or terminating EC2 instances (on-demand) to be available to your application
  - ↳ Increase or decrease will be based on CloudWatch metrics
    - CPU Utilization (DEFAULT)
    - SIMULTANEOUS USERS (CUSTOM METRIC)
- AUTO SCALING COMPONENTS

  - LAUNCH CONFIGURATION
    - ↳ EC2 Template used by ASG to provision additional instances (AMI, Type, User-data, Storage)

  - AUTO SCALING GROUP
    - ↳ Rules that govern if/when an EC2 instance is provisioned/terminated
    - ↳ MIN/MAX allows instances
    - ↳ VPC & AZs to launch instances

↳ If provisioned should receive traffic from ELB
↳ Scaling policies (Cloudwatch metric)
↳ SNS notifications

• CLOUD WATCH ALARMS
↳ Metrics selected indicate load on instances
- CPU
- Latency
↳ Alarms are triggered when exeded threshold
↳ Alarms trigger ASG

• STATELESS ARCHITECTURE
↳ Store state information off instance
↳ No-SQL database (Dynamo DB & Redis)
↳ Shared filesystem
↳ "Stick sessions" from ELB

• HIGH AVAILABILITY: Refers to architectures that remain highly available to users despite a component or systems failure. Usually, HA in AWS refer to a Multi-AZ architecture to protect applications from an outage

• FAULT TOLERANCE: Refers to architectures that not only don't suffer from applications or systems failure but suffer no degradation in performance from an outage. Usually FT refers to redundancy and it should be traded off with cost concerns

• SCALABILITY: Refers to the ability to increase size and capacity in a cost-effective way to (usually based on demand) to meet business and operational requirements. Vertical (Capacity) and Horizontal (Number)

• ELASTICITY: Refers to the ability of a system to adapt and change. IE. Automatic Scaling, IP Remapping

• COST EFFICIENT: Refers to the trade-offs required to make a system as inexpensive as possible while meeting all Functional requirements.

• SECURE: Refers to following the guidelines and proper practices to secure a system at every layer.

ROUTE 53 (DNS)

• ROUTE 53: Is a DNS (Domain Name System Server) hosting solution provided by AWS
- Domain Registration;
- (Domain Name System Service) Service:
- Health Checking:
↳ Commonly used with ELB
↳ Can be used to manage internal servers in VPCs
↳ Latency, Geo, Basic, and failover routing policies
↳ Failover S3 bucket
- Route Policies
  - Single
  - Weighted
  - Latency
  - Failover
  - Geolocation
  - Multivalue

- R53 Record sets
  - Record Type
    - A - AAAA IPv4, IPv6
    - CNAME - MX Host, Mail Exchange
  - Alias
    - ELB - Elastick Beanstalk
    - CloudFront - S3
  - Routing Policy
    - Simple: 1 Endpoint
    - Weighted: Multiple endpoints (manual LB)
    - Latency: User based latency
    - Failover: "Secondary"/"Primary" failure
    - Geolocation: Geofencing

- R53 S3 Failover
  - S3 bucket can be used as failover
  - Extremelly reliable backup solution

- CLOUDFRONT CONTENT DELIVERY NETWORK
  - Edge location is one of over 100 AWS data centers geographically distributed across the world
  - R53, CloudFront, WAF, Shield, Lambda @ Edge
  - Retrives contents from origins
    - Define contents for origins (static/dynamic)
    - Integrates with R53
    - Cache Behavior
      - > PATH pattern > Query String
      - > Min, Max, Default TTL > HTTP Methods

- CloudFront Security
  - SSL Certificates
  - End-to-End HTTPs
  - AWS WAF
  - Private Content
    - S3 (OAI) (ORIGIN ACCESS IDENTITY)
    - SIGNED URL or SIGNED COOKIES
      - > EXPIRATION
      - > RESTRICT IP
      - > TRUSTED SIGNERS (KEY PAIRS)
  - Geo Restriction

- CloudFront Consideration
  - Regional caches
  - Point entire domain to CF to speed dynamic content
  - Performance can be affected by:
    - File size/type
    - Remoting request from edge to origin
    - Query strings (reduce cache "hits")
  - Performance can be increased by:
    - Longer cache periods

DATABASES

- Relational (SQL) Databases
  - Structured Data (schema)
  - Data is normalized and spread across tables
  - SQL Query lang
  - ACID (Atomicity, Consistency, Isolation, Durability)
  - Vertical Scaling, Read Replicas
  - OLTP / OLAP
  - Oracle, SQL Server, MySQL, PostresSQL

- **No SQL Databases:**
  - Semi-structured
  - Flat table structure
  - API Queries
  - High Transaction Rates
  - Horizontal Scaling
  - Column, Key-Value, Document, Graph
  - MongoDB, CouchDB, HBase, Cassandra, Riak

RDS
RedShift | DynamoDB ElastiCache Neptune HBase EMR

- **RDS (Relational Database Service)**
  - Fully managed SQL database service
    - Not access to OS
    - Same way to connect to an onprem db
    - Provision/resize hardware on demand
    - Multi A/Z option (backups)
    - Read Replicas (MySQL, PostgreSQL, Aurora)
    - OLTP

[MySQL, MariaDB, PostgreSQL, Oracle, MSSQL Server]

- **Aurora**
  - MySQL or PostgresSQL compatible
  - 5x faster MySQL, 3x faster PostgresSQL
  - Continuous backup S3
  - Up to 15 read replicas accross 3 AZs
  - Lag of single digit milliseconds
  - Backtrack in seconds
  - Multi-master option

- **Aurora Serverless**
  - Autoscaling
  - No management of instances or clusters
  - Scale to zero
  - Pay as you go (ACU, Storage, I/O)

- Automatic minor updates
- Multi A/Z in one click
- Server/Platform
- P.I.T snapshots

- **RDS Backups**
  - Automated PiT (Point in Time) backups
  - Backups are deleted when DB deleted
  - Period max auto backups 35 days
  - Snapshots can be copied to other regions DRP
  - RDS Encryption - snapshots automatically encrypted

- **RDS Read Replicas**
  - Asynchronous copies, read only
  - Multiple read replicas from primary
  - Cross-region supported
  - MySQL, MariaDB, Postgre SQL, Aurora.
  - Can promote read replica to primary

- **RDS Multi-AZ Failover**
  - Synchronous replicates data to standby db in different AZ but same region
  - In the event of outage / failover, AWS will switch DNS record from the primary to the secondary
  - Primary must be launched into a "subnet group"
  - RDS backups taken against 2nd (standby) to avoid degradation

- **DYNAMO DB**
  - ↳ Serverless, No SQL Database, Fully managed
  - ↳ Similar to MongoDB
  - ↳ Schemaless, key-value store
  - ↳ Specify required throughput, AWS does the rest
  - ↳ Service manages all provisioning
  - ↳ Fully distributed, scales automatically
  - ↳ Built as fault tolerant highly available service
  - ↳ Integrates with EMR (Elastic Map Reduce)
  - ↳ (IoT, Gaming, Mobile, Fast I/O) Sessions

- **NEPTUNE**
  - ↳ Fully managed, No SQL, Graph db service
  - ↳ Open-source APIs
    - TinkerPop Gremlin
    - RDF SPARQL
  - ↳ Highly Available
  - ↳ Storage up to 64 Tb, up to 15 Read Replicas
  - ↳ SSE, Server Side Encryption
  - ↳ (Social Network, Knowledge Graph, Fraud Detection)

- **ELASTICACHE**
  - ↳ In Memory Cache Engine & Data Store
  - ↳ Improve db performance by caching results
  - ↳ Provisions nodes clusters for scale
  - ↳ Allows managing web sessions,
    Cache Hit / Cache Miss
  - ↳ **Memcached**
    - Simple Model
    - Easy to scale
    - Multithreaded
  - ↳ **Redis**
    - Complex data types
    - Multy-AZ failover
    - Data persistance
    - Snapshots for backup and restore
    - Encryption (HIPPA)

- **REDSHIFT**
  - ↳ Petabyte-Scale Data Warehouse, OLAP
  - ↳ OnLine Analytics Processing
  - ↳ Fully managed and scalable
    - Snapshots
    - Encryption
    - SQL (PostgreSQL compatible)
    - Low cost (~$1,000 Tb/year)
  - ↳ MPP (Massive Parallel Processing)
  - ↳ (Big Data, Analythics, BI, Tableau, Quicksight)
  - ↳ Unstructured data stored in S3
  - ↳ Query exabytes of data in few minutes
  - ↳ Forked from PostgreSQL
- **SPECTRUM** - Queries data in S3 so you can query exabytes instead of petabytes

S3 (SIMPLE STORAGE SERVICE)

- **S3** : Object storage service. Serverless purpose for designing a HA, FT, secure architecture.
  - Bulk storage (unlimited)
  - Various storage classes (optimize cost vs durability)
  - Objects versioning
  - Access restriction via S3 bucket policy
  - Object mgmt via lifecycle policy

- Hosting static files & websites
- Origin to CloudFront CDN
- File shares / backups 4 hybrid networks (AWS Storage Gateway)

↳ Objects stay within AWS region and get replicated / synced across AZs

↳ Always create S3 Bucket in region that makes sense for purpose
- Content to customers
- Sharing data to EC2

↳ Read consistency rules
- All regions support read-after-write consistency PUT of new objects to S3
- All regions use eventual consistency for PUTS overwriting existing objects and DELETES of objects

- CAP Theorem
  ↳ Consistency, Availability, Partition

• S3 COMPONENTS

• BUCKETS : Main storage containers, contain group of information, sub name spaces (folders)
- Unique name across AWS
- DNS-compliant
- No uppercase or underscores
- Lowercase letters, numbers, hyphens
- Only 100 buckets per account (soft limit)
- Bucket ownership can not be transfered

• OBJECTS : Objects are files stored in S3
- Include metadata (set key-pair names)
- Contain information specified to user
- Must be assigned storage class, determines availability, durability, cost
- All objects are private by default
- Objects can
  > Be as small as 0 bytes and as big as 5TB
  > Have multiple versions
  > Be publicly available via URL
  > Automatically switch to storage class (lifecycle)
  > Encrypted
  > Organized
- SSE (Server Side Encryption) -SSE-S3 (AWS) -SSE-C (Customer) -SSE-KMS (AWS)
- Client Side Encryption (CSE)
- SSL terminated endpoints for the API

• FOLDERS
- Folders for simplicity
- Only as a means of grouping
- AWS S3 using key-name prefixes
- AWS S3 has a flat structure, no hierarchy

• S3 FEATURES

• VERSIONING
↳ Feature to manage and store all old/new/deleted versions of an object
- Disabled by default
- Once enabled, you can suspend but not fully disable.
- Can be enabled only on the bucket level and applies to ALL object

- Cross Region Replication
- Version can be used with lifecycle policies

• STORAGE CLASSES
- Storage cost, Object Availability, Frequency of Access

> Standard
- General, all-purpose storage
- 99.999999999% (eleven nines) durability
- 99.99% (four nines) availability
- Most expensive

> Infrequent Access (S3-IA)
- Not Frequent access, but readyliy available
- 99.999999999% durability
- 99.9% accessibility / availability
- Less expensive than standard
- 30 day minimum

> One Zone Infrequent Access (S3 One Zone-IA)
- Non-critical, reproducible objs
- 99.999999999% durability
- 99.5% availability
- Less expensive S3-IA
- 30 day min

> GLACIER
- Long term archival storage
- Several hours to retrieve objs
- 99.999999999% durability
- Cheapest ($.004/6b/m)
- 90 day min

access
storage

• LIFECYCLE POLICIES
↳ Set of rules to automate object migration of an objects store class

• S3 EVENTS
↳ Event notifications, allow automated communication between S3 and other AWS services
↳ Can be sent to
- SNS Topic
- Lambda
- SQS Queue

• STATIC WEB HOSTING
↳ Provides an option to low cost host a website (static), highly reliable, service
↳ When enabled, a unique URL (endpoint) will be made available to serve
- HTML - CSS - JS
↳ Can be mapped to a R53 domain CNAME Record
↳ CORS Rules apply to access resources from other domains

• PERMISSIONS
↳ All bucket and objects are private by default
↳ Resource based policies or IAM user policy.
↳ Resource based policies:

> BUCKET POLICIES
- Policies only attached to bucket (No IAM user)
- Permission / policy to ALL objects
- Specifies actions allowed or denied for particular user

> S3 ACL (Access Control List)
- Grant access to users on other AWS account
- Both bucket & objects have ACLs
- Object ACL allow us to share S3 object with public via URL link

> IAM
- On a user level, not bucket

- GLACIAR
  - ↳ Works with S3 or as a standalone service
  - ↳ Archival service, archives, not buckets
  - → Used for data not accessed frequently
  - → "Check out"/"Check in" jobs can take hours (3-5)
  - → AWS S3 Lifecycle policies for easy archiving
  - → Very inexpensive and cost effective archival solution
  - → 99.999999999% - durability
  - → SSL/TLS endpoint
  - → Encryption at rest Rest
  - → Vault Lock Feature
    - Unable to change contents
    - For a period of time

VAULT

  - → Data Retrival
    - Expedited (1-5 min)
    - Standard (3-5 hrs)
    - Bulk (5-12 hrs)
  - → Enable Event Notifications to notify job complete
- TRANSFERING DATA TO S3
  - ↳ Single Operation Upload: Traditional upload
  - → Multi-part Upload:
    - When file is > 100 Mb ⇒ 5 Gb
    - Parallel threads to upload chunks
    - Can be used for files up to 5 Tb
  - → Transfer Acceleration
    - Upload through CloudFront Edge
    - Change endpoint S3 → CF CDN
  - → Snowball
    - Petabyte service data transportation
    - Up to 80 Tb per device

→ Snowball Edge
- Snowball plus compute capacity
- Up to 100 TB per device
- Can be clustered
- S3 API Interface
- Lambda Functions

>100 days over network 1Gbps

→ Snowmobile
- Exabyte scale
- 100 PB per Snowmobile

>20 years over network 1Gbps

→ Storage Gateway
- Connect datacenters to cloud based storage
- VM WARE or Hyper-V
- Encryption at rest - in transit
- Store local data at S3
- Volume Gateway
  - > EBS Snapshots
  - > Gateway Cached Volumes
    - Create storage volumes and mount them as iSCSI devices on the on-prem servers
    - Gateway will store data written to this volume in S3 and will cache frequently accessed data on-premise in the storage device
  - > Gateway Stored Volumes
    - Store all data locally (on premise) in storage volume
    - Gateway will periodically take snapshot of the data as incremental backups and stores them on S3

- File Gateway
  - Local NFS
  - Objects are stored in S3
- Tape Gateway
  - Emulates industry-standard iSCSI-based virtual tape library
  - Common backup applications (Veeam, Veritas Arcserve, Dell)

HYBRID ENVIRONMENTS

- VPN (VIRTUAL PRIVATE NETWORK)
  - A VPN enables the ability to extend a network from one geographical location to another geolocation
  - Traffic traverses the internet
  - To extend On Prem <> Cloud
  - Allows an AWS EC2 instance to communicate internally without the need of a public IP address and IGW
  - Provides additional level of security by ensuring all traffic is encrypted
  - VPN connection has two parallels routes (IPsec) tunnels, for redundancy
  - Only one VPG (Virtual Private Gateway) can be attached to a VPC
  - A VPC can have a VPG and IGW at the same time
- IPsec (Internet Protocol Security)
  - Suite of secure network protocols for IPv4
  - Provides Mutual Auth
  - Provides Encryption - negotiation of keys
- CUSTOMER GATEWAY
  - Physical device that acts as a "connector" to the VPN
  - Needs a public IP address
  - Static Routing: Manual definition of routes
  - Dynamic Routing: Routes automatically propagates
    - Border Gateway Protocol (BGP)
    - Autonomous System Number (ASN)
- VPG (VIRTUAL PRIVATE GATEWAY)
  - "Connector" to the VPC side of the VPN
  - Connected to the VPC
  - Target for Route Tables
  - CloudHub: Multiple customer networks
- ROUTER
  - Route Tables are actually part of "router" assigned to your VPC, despite AWS dispersed users of actually having and managing a physical router.
  - When setting a VPN, the route table must include routes for the on prem server.
  - Choose route propagation for BGP to automatically add routes
- AWS DIRECT CONNECT
  - Connects your customer WAN to AWS, ie Fiber Cable connection.
  - 1Gbps, 10Gbps (Verizon, ISP)
  - Access to Public Service Endpoints in all regions

↳ Reduce network cost
↳ Increased network consistency
↳ Dedicated private network connection on-premise
↳ Direct Connect Location
- PVI (Private Virtual Interface)
> Not a ISP
> Internal IPs
- PVI (Public Virtual Interface)
> Connect to public AWS endpoints

↳ DIRECT CONNECT GATEWAY
↳ Connect to any VPC in your account in any region
↳ VPC cannot have overlapping CIDR blocks
↳ Communication traverses AWS backbone

• VPC Peering
↳ VPC are isolated by default
↳ VPC peering allows 2 VPC to connect and share resources
↳ Used to extend your private network from a VPC to or subnet to another VPC
↳ To share internal resources
↳ Different regions is called Inter-Region VPC Peering
↳ CIDR Blocks cannot overlap

SIMPLE NOTIFICATION SERVICE

↳ SNS coordinates and manages the sending and delivery to subscribers of messages
↳ Cloudwatch + SNS, Mobile Push Notifications
↳ TOPIC
- Object to which you public messages
- Subscribers subscribe to recive msgs

↳ Subscribers
- Endpoint message is sent
- Available endpoints:
> HTTP > Email JSON
> HTTPS > SQS
> Email > (iOS/Android/Amazon/Microsoft)
> Lambda > SMS

↳ PUBLISHERS
- The entity that triggers
> Application > Cloudwatch Alarm
> S3 Event

SIMPLE QUEUE SERVICE

↳ DECOUPLED ARCHITECTURES
• TIGHTLY COUPLED SYSTEMS
> Depends upon each other
> If a component fails, all components fail
• LOOSELY COUPLED / DECOUPLED SYSTEM:
> Multiple components can process info without being connected
> Components are not connected, - if 1 fails the rest continue to operate (FT/HA)
• AWS Services to decoupled systems
SWF - Simple WorkFlow
SQS - Simple Queue Service

• SQS
↳ Simple Queue provides ability to have hosted / HA queues that can be used for messa between servers
↳ SQS is used to create decoupled app environments

↳ Msgs are retrived through polling
↳ Long Polling
- (1-20) seconds
- Waits until message is available before sending a response, and will return all msgs from all SQS services
- Long Polling reduces API calls
↳ Short Polling
- Immediate
- Will not return all possible messages in a poll
- Increases API requests
↳ Each msg is up to 256 KB of text.
↳ Larger msgs use S3
↳ Standard Queue: Guarantee delivery not order
↳ FIFO Queue Order & delivery 300 Tps
↳ SQS Workflow
- Generally a "worker" will "poll" a queue to retrieve waiting msgs for processing
- Auto Scaling can be applied to a queue based on size so if a component gets an increase in demand, the number of workers instances can increase

SNS → Push / SQS ← Poll

Message Queue (AWS MQ)

↳ A managed service, compatible with Apache MQ, an open source message broker
↳ Compatible with JMS API or protocols such as AMQP, MQTT, OpenWire, STOMP
↳ Functions as Queue or Topic
↳ One to One, One to Many
• MQ Components
↳ Broker
- Single Instance
- HA Pair (active/standby)
- Managed Updates
↳ Active MQ Console

Simple Work Flow

↳ Fully managed orchestration service
↳ A SWF allows an architecture developer to ~~independent~~ implement distributed, async apps as a work flow
↳ A workflow coordinates and manages execution of activities that run async in various machines
↳ SWF has constant execution
↳ Order guaranteed, no duplicate tasks
↳ SWF is primarily ~~on~~ on API
↳ workflows can last up to a year
↳ Components SWF
- Starter: Starts the workflow
- Workflow (Decider): A sequence of steps
- Activities: Single steps
- Tasks
  - Activity Tasks: Worker to perform function
  - Decision Tasks: State of workflow
- Worker: Responsible of task execution

API GATEWAY

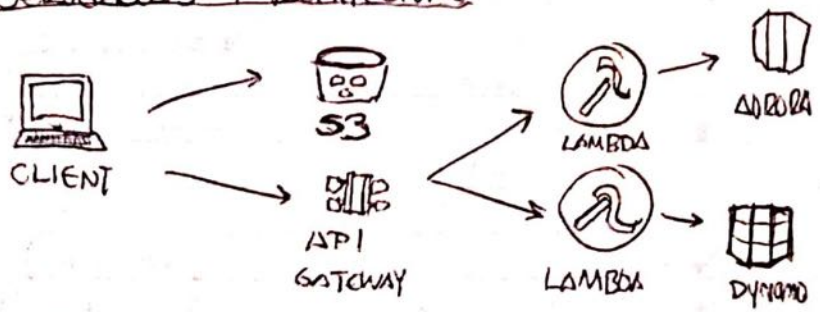
- Backend language agnostic
- Define a collection of methods and their results
- Alternatively you can expose the API through an EC2 instance or a Lambda
- Internet <-> CloudFront <-> API Gateway -> EC2 / Lambda / other Endpoint
- Fully managed service to create and manage your own APIs
- Acts as a "front door" for the app, allowing access to data/logic/functionality in backend
- Features
  - RESTful APIs
  - Deploy to 'stage' (dev, beta, prod)
  - Create API versioning
  - Roll back previous API versions
  - Custom domain names
  - Create and manage API keys and meter usage through CloudFunction
  - Set throttling rules based on the number of request per second
  - Set Signature v4 and authorize API call
- Benefits
  - Ability to cache API response
  - DDoS protection via CloudFront
  - SDK generation for iOS, Android, JS
  - Supports Swagger
  - Req, res transformation

SERVICE ORIENTED ARCHITECTURE IN AWS

- Serverless Computing

- MICROSERVICES:
  - Each microservice is a self contained unit of functionality on independent resources
  - Microservices communicate with each other using APIs
  - Decouple Microservices using Queues

- MICROSERVICES CONSTRUCTS
  - Lambda Functions
  - Messaging Services (SQS, MQ)
  - API Gateway
  - Containers (ECS, EKS)
  - Load Balancing and Autoscaling
- SERVERLESS ARCHITECTURE



- LAMBDA
  - You pay for what you use
  - No config or balancing or optimize cost
  - "Serverless" computing platform

- Lambda lets you configure amount of resources (memory, CPU) available to function
- Integrates with many other resources
- Langs:
  - Node.js
  - Java
  - C#
  - Python
  - Go
- Lambda over EC2?
  - Changes to AWS S3 bucket
  - Updates to DynamoDB table
  - Custom events from apps or devices
  - Kinesis Streams

CLOUDWATCH ALARMS

- METRICS
  - Used to monitor services
  - Monitor environment by config and viewing metrics
  - EC2-per-instance metrics:
    - CPU utilization
    - CPU credit usage
  - S3 metrics
    - # of objects
    - Bucket Size Bytes
  - ELB Metrics
    - Request Count
    - Unhealthy Host Count
  - EC2 standard monitoring is 5mn (dev)
  - EC2 standard monitoring is 1mn (prod)
  - Can be triggered (SNS) based on thresholds
  - Auto Scaling heavily uses Cloudwatch to trigger provision instances from ASG (thresholds)

- LOGS
  - Centralized repositories
  - Log Streams (EC2 Cloudwatch Logs Agents)
  - Metrics can be made from Log Streams

- EVENTS
  - Create rules to respond to events in Event Streams
    - Trigger Lambda function when EC2 instance enters running state
  - Schedule Events

- MONITORING
  - System Status Check
    - Things that happen outside control
    - Loss of network connection
    - Loss of system power
    - Software issues / hardware issues on physical
    - Solution: Stopping and restarting will fix issue
  - Instance Status Check
    - Things that we do control
    - Failed system status checks
    - Misconfigured networking or startup conf
    - Exhausted mem
    - Corrupted file system
    - Incompatible kernel
    - Solution: Reboot, or dedicated system conf issue
  - Default CloudWatch metrics
    - CPU Utilization
    - Network I/O
    - CPU CreditBalance
    - CPU Credit Usage

→ OS Level metrics (Third party script by AWS)
- Mem utilization, usage, available
- Disk swap utilization
- Disk space utilization, usage, available

• ALARMS
→ Select a metric and set a threshold to fire a CloudWatch Metric.
→ Alarms can be used to trigger other events in AWS like publishing a SNS topic or triggering auto scaling

CLOUDWATCH TRAILS

CLOUDTRAIL

→ Logs the API usage made to AWS
→ Command Line, SDK, Console, Logs All
→ Logs are placed in designated S3 Bucket
→ Help address security concerns, by allowing to view what actions are users in AWS acont are performing
→ CloudTrail can be big (very)
→ Global or Per[redacted] Region
→ Send CloudTrail Events to CloudWatch Logs
→ Define custom metrics & alarms
- SG and NACL Changes
- VPC changes
- Failed Console Login Attempts
- Failed API Authorizations
- IAM Changes

VPC LOGS AND ACCESS LOGS

• FLOW LOGS
→ Allow to collect information about IP traffic going to and from the network if in the VPC
→ Stored in a group log in CloudWatch
→ Can be created on a specific VPC, Subnet, NI
→ FL created on a VPC or Subnet will include all NIs
→ Each NI will have its own unique log stream
→ Can optionally accept traffic, reject, or all traffic

[SOURCE IP] [DEST IP] [SOURCE PORT] [DEST PORT]

• ACCESS LOGS
→ ELB
→ S3
→ CloudFront

CLOUDFORMATION

→ IaaC: Infrastructure as Code
→ Uses templates to create stacks (JSON/YAML)
→ Use same deploy template to update or replicate copies of that architecture
→ Saves time - Avoid manually creation of arch
→ Version control of infrastructure
→ CRUD Stack (Nested Stack)

• TEMPLATE
- Resources: Services to deploy
- Parameters: Variables in template
- Mappings: Lookup Table, AMIs
- Conditions. Define ifs and thens
- Outputs: Info returned from stack

- HELPER SCRIPTS
  - cfn-init: packages, users, groups, files, commands
  - cfn-signal: user with wait conditions
  - cfn-hup: in place instance update of packages

ELASTIC CONTAINER SERVICES

→ Docker service managed in AWS
→ Easily create and manage a fleet of Docker containers on a cluster of EC2
  - Higher Instance Utilization
  - Consistency between env
  - Microservices
  - Continues Deployment
  - LB
  - Autoscaling

- ECS Components
  ↳ Docker images
  → ECR (Registry)
  - Task Definition
    - Images Docker
    - CPU, Mem
    - Launch Type: EC2 or Fargate (managed, serverless)
  → EC2 Cluster
    - EC2 Container Instances
      > EC2 agent

ECS ≈ KUBERNETES

- EKS, ECS for Kubernetes
  → Managed availability and scalability of K8s control plane nodes
    - Start/stops containers
    - Schedules containers
    - Replaces unhealthy nodes

ELASTIC BEANSTALK

→ Web based or worker based (SQS)
→ PaaS
→ Easiest way to deploy and automate
  - Load Balancing
  - Autoscaling
  - Monitoring
  - Platform Mgmt
  - Code Deployment
  - Java
  - PHP
  - C#
  - Node.js
  - Ruby
  - Python
  - Go
  - Docker
→ Servers
  - Apache - Nginx - Passenger - IIS
→ Deployment Options
  - In Place (Rolling) - Blue-Green
  - In Place (Rolling): Updates environment
  - Blue-Green: Deploys to new Env and falls back to old Env if anything fails.

AWS KINESIS (ANALYTICS)

→ Manage service that provides the ability to multiple producers of data to send their data to Kinesis simultaneously into a stream to consumers to read.
→ KINESIS VIDEO STREAM
  - Stream video to AWS
  - Real-time or batch video processing and analytics
→ KINESIS DATA STREAM
  - Inject data from many sources
  - RT processing applications
  - Kinesis connector - EMR
  - Data Processed in sequence
  - SSE

KINESIS FIRE HOSE
- Load streaming data into S3, Redshift, Elastic search, Splunk

KINESIS DATA ANALYTICS
- Run SQL queries against Data Streams or Firehose
- Send results to output DataStreams of Firehose

KINESIS DATA STREAMS COMPONENTS

- Stream: Contains one or more Shards
- Shard: (Processing Power)
  - 1Mb/sec data input and 2Mb/s data output
  - Distribute data to shards using Partition Keys

- Producers: (Data Creator)
  - Send data to Kinesis
  - Build producers to continuously input data to Kinesis
  - Can include (not limited to)
    - IoT
    - Mobile phones
  - Ability to have multiple (thousands) of prodocers and scale based on need
    - More data, More shards
    - Shards ~ 2Mb of read data, 1Mb of write data / second
  - Kinesis Data Streams API
  - Kinesis Producer Library (Java)
  - Kinesis Agent (Linux)

- Consumers
  - App that consume the data of kinesis stream data
  - Concurrently, multiple consumers multiple same data
  - Kinesis Consumer Library (Java)
    - Java Lib (EC2) - Wrapper for other lang
    - Lauces consumer per shard
    - Autoscaling
  - Lambda can read stream data
  - Kinesis Connector for EMR

BENEFITS
- Real-time processing
- Parallel processing
- Durable: Replicated in 3 datacenters, 24hrs 7 d max
- Scale: From mb to Tb

WHEN TO USE

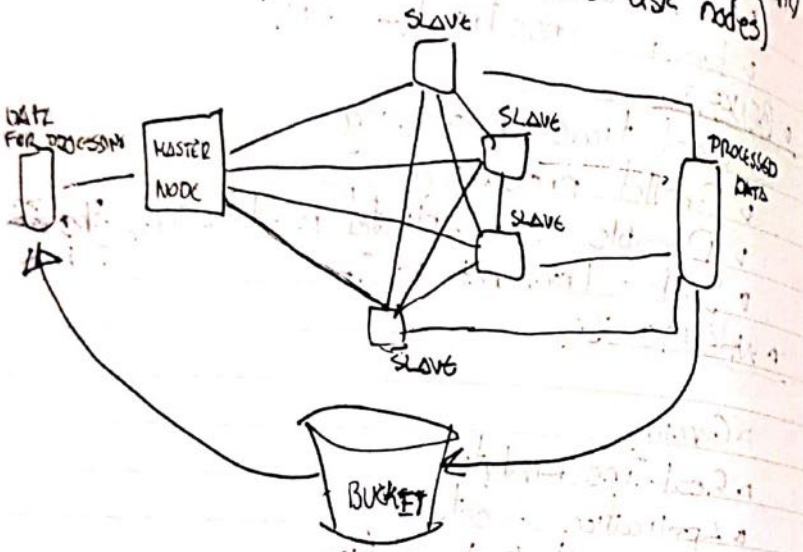
- Gaming
- Real-Time Analytics
- Application Alerts
- Log/Event Data Collection
- Mobile Data Collection

ELASTIC MAP REDUCE

- Big data, managed service for hadoop clusters or Spark on Aws
- Processing of data in large batches
- EMR is used to analyze and process vast amounts of data

• EMR ADVANTAGES

- Storage is S3
  - Load into HDFS or keep in S3 (EMRFS)
- Transient clusters
- Spot instances
- Bootstrapping
- Preconfigured app frameworks (Hadoop, Spark)
- Scalability (add/remove core task nodes)



• EMR MASTER NODE:

- Responsible for coordinating components and distribution of data and tasks among slaves for processing
- Tracks status and monitors health checks
- SSH is permitted to run interactive nodes jobs

• EMR SLAVE NODES

- CORE NODES:
  - Run tasks and store data in HDFS
  - "Heavy lifting" of data
- TASK NODE
  - Only run tasks
  - Optional

• EMR MAP PHASE

- Mapping splits data (defines processing of data for each split)
- Block size is for HDFS is 128MB, optimum
- Larger instance size, ~~[illegible]~~ the more chunks can be processed
- If there are more chunks than nodes/mappers chunks will queue for processing

• EMR REDUCE PHASE

- Reducing aggregates data into single output file
- Reduced data needs to be stored (S3)
- Data processed is not persistent

AWS WELL ARCHITECTED FRAMEWORK

• OPERATIONAL EXCELLENCE //OPERATIONAL EXCELLENCE

- The ability to run and monitor systems to deliver business value and to continually improve supporting processes and procedures
  - Operations as code
  - Annotate documentation
  - Frequent, small, reversible changes
  - Refine operations procedures frequently
  - Anticipate failure
  - Learn from operational failure

- Prepare
  - CloudFormation
  - AWS Config

- CloudWatch
- Cloud Trail
- Flow Logs

• AWS Config
↳ Visibility into resources configuration
- Detailed config
▵ Snapshots of resources inventory
▵ Deliver Report to S3
- Relationships between resources
- Historical timeline
- Integrates with CloudTrail
↳ Notifications
- Resourse is CRUD
- Config Stream to SNS Topic
↳ Rules
- Evaluates compliance
- Managed Rules
- Custom Rules

↳ OPERATE
- Cloud Watch
- Cloud Trail
- Flow Logs
- X-Ray (Microservices, Optimization)
↳ EVOLVE
- Elastic search
- Code Build
- Code Commit
- Code Deploy
- Code Pipeline
- Code Star
- ~~X~~ Ray

• AWS Elasticsearch
↳ Open source distributed search engine
- Log Analytics
- Full-text Search
- OpsInt
↳ Data sources
- Logstash
- Kloudwatch Logs
- Kinesis Firehose
↳ Kibana Dashboards

// RELIABILITY

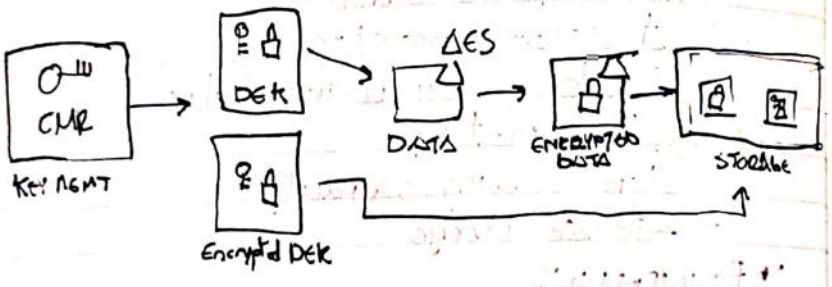
• RELIABILITY
↳ "The ability to recover from failure and migrate disruptions"
- Test recovery procedures
- Automate recovery from failure
- Scale horizontally
- Stop guessing capacity.
- Automate change
↳ FOUNDATIONS
- IAM : Access Control
- VPC : Isolated Network
- Trusted Advisor : Support, Service Limit, security, optimization
- AWS Shield : DDoS Protection
↳ CHANGE MANAGEMENT
- Cloud Watch : Control Access
- Config : Configuration Awareness
- Cloud Trail : Audit AWS APIs
- Auto Scaling : Demand Mgmt

FAILURE MANAGEMENT

- CloudFormation: IaaC
- S3: Durable Backups
- Glacier: Durable Archives
- KMS: Reliable key Mgmt

AWS KMS

- Managed Encryption Key Mgmt service
- Master keys never leave AWS KMS
- IAM Permissions, Key permissions
- Unique data key for each object
- EBS, S3, RDS, Redshift, Elastic Transcoder, EMR, SQS, Kinesis, Workmail

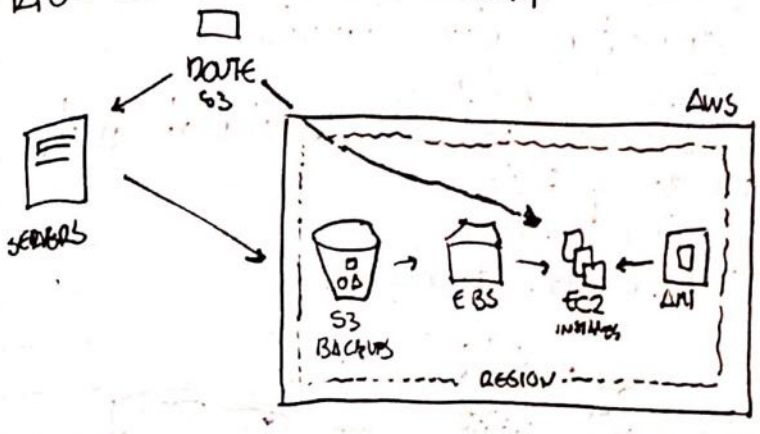


DISASTER RECOVERY STRATEGY

- RTO (Recovery Time Objective): How long to recover
- RPO (Recovery Point Objective): How much data is lost

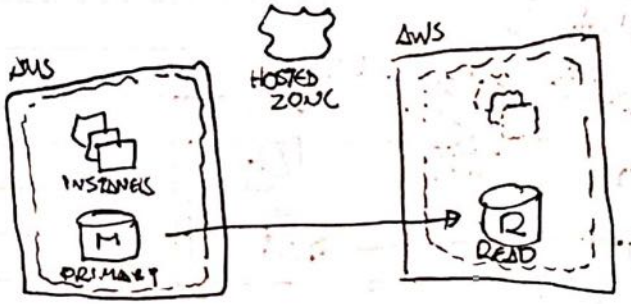
BACKUP AND RESTORE

- Backup data to AWS
- Have AMIs in region recovery
- CloudFormation templates standing by
- In Case of Disaster
  - Spin up AMIs
  - Restore backups
  - Modify DNS to point new instances
- RTO: Time to launch instances, restore data, update DNS
- RPO: Data since last backup



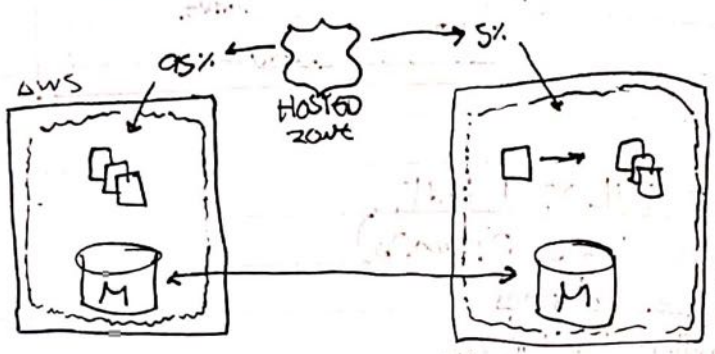
PILOT LIGHT

- CRR (RDS, DynamoDB)
- Instances Stopped
- Smaller DB instance
- In Case of Disaster
  - Start instances
  - Scale up DB, Promote to Primary
  - Modify DNS, Route 53 Failover
- RTO - Time to startup instances and scale
- RPO - Replication lag



→ LOW CAPACITY STANDBY

- CRR, Similar to Pilot Light
- Some capacity running 24/7
- Continuous testing with trick traffic
- Multi-Master Option (Aurora)
- In Case of Disaster
  - ▷ Scale up / Autoscale to full prod capacity
  - ▷ Route 53 Failover
- RTO: Time to scale
- RPO: Replication lag only



→ MULTI SITE ACTIVE ACTIVE

- CRR
- Full capacity 24/7, Two Regions
- Multi-master Option
- In Case of Disaster
  - ▷ Route 53 Failover
- RTO: Time to fail over
- RPO: Replication lag

//SECURITY

• SECURITY

→ "The ability to protect information, systems and assets while delivering business value through risk assessments and migration schemes"

- Implement a strong Identity Foundation
- Enable Traceability
- Apply security at every layer
- Automate security
- Protect data in transit and at rest
- Prepare for security events

"Security is a Shared Responsability"

→ IAM (IDENTITY AND ACCESS MANAGEMENT)

- IAM: Access Control
- Organizations: Central Manage Accounts
- MFA Token: Identity Authentication
- Temp Creds: Limited Life Credentials

→ DETECTIVE CONTROLS

- CloudTrail: API Access Logs
- CloudWatch: Log Metric Filters
- Config: Resource Inventory
- Guard Duty: Threat Detection

• AWS Guard Duty:

→ Intelligent Threat Detection:

- Threat Intelligence Feeds
- Machine Learning
- Managed Rule Sets
- Continuous Monitoring
  - CloudTrail Events
  - VPC Flow Logs
  - DNS Logs
- Can trigger Lambdas

↳ INFRASTRUCTURE PROTECTION

- VPC: Isolated Virtual Networks
- Inspector: Vulnerability Detector
- Shield: DDoS Protection
- WAF: Firewall Application

• AWS Inspector
  ↳ Agent based security scans
  ↳ Rules Knowledgebase
    ▷ Common vulns
    ▷ Best Practices
    ▷ CIS Benchmarks
    ▷ RIBA (Runtime Behavior Port/Protocols Analytics)

• AWS SHIELD:
  ↳ Automatically enabled
  ↳ Standard (Network / Transport Layer)
    - SYN Floods. UDP Floods
    - Reflection attacks
    - Malformed TCP
    - Automatic Protection 98% of 3/4 attacks
  ↳ Advanced
    - Advanced DDoS Protection
    - Larger Attacks
    - 24/7 Response team
    - Cost Protection

• AWS WAF
  ↳ Application Layer Protection
    - SQL Injections
    - XSS
    - Bad Bots
    - High Request Rates
    - Large Request Size
    - Blacklist/Whitelist IPs
  ↳ Associate with
    - CloudFront
    - ALB

↳ DATA PROTECTION

- AWS MACIE: Data Security Automation
- S3: Object Encryption
- EBS: Block Encryption
- KMS: Encryption Key Management

• AWS MACIE
  ↳ Machine learning data classification
  ↳ Data stored in S3
  ↳ Risky or suspicious activity
  ↳ Alerts
    - High Risk Data Events
    - Credentials in Source Code
    - Unencrypted backups
    - Signs of potential risk

↳ INCIDENT RESPONSE

- CloudFormation: Infrastructure as Code
- IAM: Response Team Authorization.

• EFFICIENCY // PERFORMANCE EFFICIENCY
  ↳ "The ability to use computing resources efficiently to meet system requirements and to maintain that efficiency as demand changes and technologies evolve"
    - Democratize advanced technologies
    - Go global in minutes
    - Use serverless architectures
    - Experiments more often
    - Mechanical sympathy

↳ SELECTION

- Compute: Auto scaling
- Storage: S3, EBS
- Database: DynamoDB, RDS
- Network: VPC, Route 53, Direct Connect

→ REVIEW
- AWS Blog and News Update

→ MONITORING
- CloudWatch: Metrics, Alarms, Notifications
- Lambda: Automated Actions

→ CACHING / TRADEOFFS
- CloudFront: Global Caching
- ElasticCache: Request Offloading
- Snowball: Data Migration
- RDS: Read Replicas

// COST OPTIMIZATION

• COST OPTIMIZATION
→ "The ability to avoid or eliminate unneeded cost or suboptimal resources"
- Adopt a consumption model
- Measure overall efficiency
- Stop spending money on data centers
- Analyze and attribute expenditure
- Use managed services

→ COST EFFECTIVE RESOURCES
- Instances: Reserved and Spot
- Tags: Cost Allocation

→ MATCHED SUPPLY & DEMAND
- Auto scaling ⟷ CloudWatch
  ▸ Scale when Demand Drops / Increases

→ EXPENDITURE AWARENESS
- AWS SNS ⟷ CloudWatch: Notification when Cost > Budget

→ OPTIMIZE OVER TIME
- AWS Trusted Advisor: Weekly Update Email

HOW TO PREPARE FOR THE EXAM