AWS Reviewer for Solutions Architect

|============================================|

|General purpose summary of Core AWS Services|

|============================================|

AWS EC2

AWS S3

AWS CloudFront

AWS CloudWatch

// AWS Accounts Management & Accounts Security

AWS IAM

AWS Organizations - Enables consolidation of multiple AWS Accounts as well as AWS consolidate billings into an organizational tree with each branch representing a team or department.

IAM Credentials Report (account-level) - a report that lists all your account's users and the status of their various credentials

IAM Access Advisor (user-level) - Enables you to see the service permissions granted to a user and when those services were last accessed.

AWS Budgeting/Expenses/Costs Tools

AWS Pricing Calculator - Enables you to model your cloud solutions before building them, explore the price points and calculations behind estimates, and find the available instance types and contract terms that meet your needs.

AWS Budgets

AWS Cost and Usage Report

AWS Cost Explorer - Enables you to view and analyze your costs and usage.

|=================|

|AWS Terminologies|

|=================|

AWS Reserved Instances:

NURI - No Upfront Reserved Instance => No paying in start of term, billed at a discoutned hourly rate for every hour within the term, regardless if the Reserved Instance is used or not.

PURI - Partial Upfront Reserved Instance => Pay a portion of the Reserved Instance's cost at the start of the term, with the remaining hours billed at a discounted hourly rate, regardless of whether the Reserved Instance is being used.

AURI - All Upfront Reserved Instance => Pay full at the start of the term, with no other costs or additional hourly charges incurred for the remainder of the term, regardless of hours used.

AWS Region - A physical geographical location with one or more Availability Zone.

Availability Zone - Contains Two or more Data Centers.

ENI (Elastic Network Interfaces) - Gives EC2 instances access to the network as a Virtual Network Card.

Elasticity - Ability to acquire resources as you need them and release resources when you no longer need them.

IOPS (Input/Output Operations Per Second) - is the standard unit of measurement for the maximum number of reads and writes to non-contiguous storage locations.

Golden AMI - is an AMI that contains the latest security patches, software, configurations, that you need to install for logging, security maintenance and performance monitoring.

Horizontal Scaling - is adding additional nodes or adding more machines of similar capacity to your workload.

Vertical Scaling - is adding more power to your current machines to your workload.

Scale out - Adding more resources

Scale in - Reducing/removing resources

ACID - Atomicity, Consistency, Isolation, and Durability. A set of principles that ensure database transactions are processed reliably.

Atomicity - refers to how a given set of database operations must all either complete successfully or fail altogether. Either transaction must finish and update or revert the database to old state if transaction fails.

Consistency - ensuring the information in the database is always correct state. Data integrity constraints must always be followed. Also ensures that if any violation if data integrity checks occurs, the entire containing transaction will be aborted, and any changes already made will be rolled back.

Isolation - refers to the need to separate the details of multiple transactions in process at the same time. Database should perform operations sequentially and not at the same time.

Durability - requires that every successful transaction's data operations are not removed and cannot be lost-even in the event of a complete system shutdown.

Read Capacity Units (RCUs) - Read capacity of a database

Write Capacity Units - Write capacity of a database

Cache hit - If the cache contains the read data requested

Cache miss - If the cache does not contain the read data requested

Stale data - Data that is expired or is very old

Time to Live (TTL) - Lifespan of a data in a computer or network

CI/CD- is the combined practices of continuous integration and continuous delivery or continuous deployement.

Public IP

* Machines can be identified on the Internet
* Must be unique across the whole web (not two machines can have the same public IP)
* Can be geo-located easily

Private IP

* Machine can only be identified on a private network
* IP must be unique across the private network
* Two different private networks can have the same Ips
* Machines connect to the Internet using a NAT (Network Address Translation) + Internet Gateway
* Only specified range of Ips can be used as Private Ips

Elastic Ips

* Stopping an EC2 instance can change its public IP
* Use Elastic IP if fixed public IP is needed
* A public IPv4 IP you own as long as you don’t delete it
* Can be attached to one instance at a time
* Can be used to mask the failure of an instance or software by rapidly remapping the address to another instance in account
* Can only have 5 Elastic IP in your account
* Overall, try to avoid using Elastic IP
  + Can often reflect poor architectural decisions
  + Use random public IP and register a DNS name to it instead

|===============================================|

|6 Pillars of the AWS Well-Architected Framework|

|===============================================|

Operational Excellence - Ability to support development and run workloads effectively, gain insight to their operation, and continuously improve supporting processes and procedures to delivery business value.

Security - Ability to protect data, systems, and assets to take advantage of cloud technologies to improve your security.

Reliability - Ability of a workload to perform its intended function correctly and consistently when it’s expected to. This includes the ability to operate and test the workload through its total lifecycle.

Performance Efficiency - Ability to use computing resources efficiently to meet system requirements, and to maintain that efficiency as demand changes and technologies evolve.

Cost Optimization - Ability to run systems to deliver business value at the lowest price point.

Sustainability - Addresses the long-term environmental, economic, and societal impact of your business activities.

|================================|

|EC2 Instances Purchasing Options|

|================================|

On-Demand Instances - Short workload, predictable pricing, pay by the second

Reserved (1 & 3 years) - Long workloads, and flexible instances(converting instances to other types)

Savings Plans(1 & 3 years) - commitment to an amount of usage instead of instance types, for long workloads.

Spot Instances - Short workloads, cheap, but can lose instances(less reliable) because of set price threshold that will remove the instance if the budget goes over the set price. If all instances are already used by someone else, you won't be able to use an instance

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

-You have access to the physical server

Dedicated Instances - No other customers will share your hardware

-You have your own instance in your own hardware

Capacity Reservations - Reserve capacity in a specific AZ for any duration, but will be billed the same on-demand instance pricing regardless if you use the instance or not.

|==================|

|EC2 Instance Types|

|==================|

General Purpose Instance Type - Good for general purpose applications that uses all resources equally.

Compute optimized Instance Type - Good for high-performance processing, high-performance computing (HPC).

Memory Optimized Instance Type - Good for High-performance databases, big data analysis, and delivers high performance workloads that process large data sets and memory.

Accelerated Computing Instance Type - Good for Machine Learning, HPC, Graphics.

Storage optimized Instance type - Good for high sequential read/write access to very large data sets. Optimized to deliver IOPS. Suited for real-time analytics, transactional workloads, log processing, etc.

|=====================|

|AWS Compute Optimizer|

|=====================|

-Analyzes the configuration and utilization metrics of your resources to report whether resources are optimal, and generates optimization recommendations to reduce the cost and improve the performance of workloads.

-Can be used to get recommendations on how to optimize EC2 instance types, to reduce costs and improve performance of workloads.

-Uses Amazon Machine Learning to analyze resources.

|===========================|

|Spot Instances & Spot Fleet|

|===========================|

Spot Fleets = set of Spot Instances + (optional) On-Demand Instances

-Spot fleet will try to meet the target capcity with price constraints

-Strategies to allocate Spot Instances:

lowestPrice: from the pool with the lowest price (cost optimization, short workload)

diversified: distributed across all pools (great for availability, long workloads)

capacityOptimized: pool with the optimal capcity for the number of instances

-Spot fleets will allow us to automatically request Spot Instances with the lowest price.

|====================|

|EC2 Placement Groups|

|====================|

Cluster - Instances are grouped together into a low-latency group hardware within a single AZ. (High performance but high risk)

-Great network(10 Gbps bandwidth between instances)

-If the hardware fails, then all instances will fail at the same time.

-Used for apps with big data job, or need fast network throughput

Spread - Instances are spread across different hardware. But restricted to 7 instances per group per AZ. Used for critical applications.

-Used for apps that needs high availability

-Used for critical applications where each instance must be isolated in cases of failure.

Partition - Spreads instances across many different partitions which relies on different sets of hardware within an AZ. (They are still spread but are not isolated from another failure, but a partition should be isolated to another partition of failure. Allows scaling of 100s of Ec2 instances per group. Hadoop, Cassandra, Kafka.)

-Up to 7 partitions per AZ

-Can span across Multiple AZs in the same region

-Up to 100s of EC2 instances

-Instances in a partition do not share rack with the instances in the other partitions

-EC2 instances get access to the partition information as metadata service.

-Used in Big Data service: HDFS, HBase, Cassandra, Kafka

|================================|

|Elastic Network Interfaces (ENI)|

|================================|

-Logical Component in a VPC that represents a virtual network card

-You can create ENI independently and attach them on the fly (move them) on EC2 instances for a failover. The Private IP will be moved from one EC2 to another in cases of a failover.

-Bound to a specific AZ

|=============|

|EC2 Hibernate|

|=============|

-In-memory (RAM) state is preserved

-Instance boot is much faster(the OS is not stopped/restart)

-Under the hood: the RAM state is written to a file in the root EBS volume

-The root EBS volume must be encrypted.

-Used for long-running processing

-Used for saving the RAM state

-Used for services that take time to initialize

-Supports a lot of Instance Families

-Works for a lot of AMI

-Works for RootVolume(EBS)

-Available for On-Demand, Reserved, and Spot Instnaces

-An instance can NOT be hibernated more than 60 days.

|================|

|AWS Auto Scaling|

|================|

-Launches or terminates instances based on specified conditions

-Automates schedules and health checks as demand increases or decreases

-Automatically registers new instances with load balancers when specified

-Can launch across AZs if an AZ is unhealthy or unavailable. It will then find an unaffected AZ and launches it there.

Scaling options

Scheduled - Scale based on date and time, useful for predictable workloads

Dynamic - Supports target tracking or conditional changes that are unpredictable. Provides extra capacity when traffic spikes and prevents excessive idle resources.

Predictive - Good for predicted demand, scale based on machine learning. Can use AWS EC2 Auto Scaling with AWS Auto Scaling to implement predictve scaling.

Dynamic Scaling Policy Types

Simple scaling - Simple scaling adjustment.

Use cases: New workloads, spiky workloads

Step Scaling - Adjustment depends on size of alarm breach when capacity is exceeded

Use cases: Predictable workloads

Target tracking scaling - Target value for specific metric, Auto Scaling adds or removes resources in order to get close as much as possible to target value

Use cases: Horizontally scalable applications, such as load-balanced applications.

Auto Scaling groups

Specify the minimum, and maximum number of instances for that group

-Minimum capacity

-Maximum capacity

-Desired capacity: Reflects the number of instances that are running and can fluctuate in response to events, can never be lower or greater than the minimum and maximum capacities

AWS EC2 Auto Scaling: Purchasing options

On-demand Instances or Reserved Instances

Spot Instances

-Specify what percentage of the desired capacity should be fulfilled with EC2 instances

-Auto Scaling then provisions the lowest priced combination of instances to meet the desired capacity based on preferences

-Auto Scaling requests on other instance type pools if requests fails from an instance type pool

Best practice:

Using of few instance types to avoid launching instances from instance pools that don't have sufficient capacity

|=================|

|EBS VS EFS VS S3 |

|=================|

EBS - High performance storage, that doesn't go away on reboot. Your data will not disappear when you reboot EC2 instances.

-Mounted used directly by the host, just like any virtual hard drives

Block storage - A type of storage that places data in blocks, and then stores those blocks in separate pieces. Each block has an identifier where you're given the key to put everything back together.

When to use EBS - A high performance low-latency drive.

-Really good for booting up hosts and EC2 instances with AMI stored inside it.

S3 - is an object storage that places data into objects each object having unique identities. Given that S3 is an object storage, OS cannot be stored inside it.

EFS - A network file system. Grows on demand, will shrink and expand when needed. Used when you have a lot of computers that needs to access data on a network.

-Used when you need to store a lot of data on a network at a high performance that a lot of computers are going to access.

|==========|

|EBS Volume|

|==========|

-Is a network drive you can attach to your instances while they run

-Allows your instances to persist data, even after their termination

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

-Bound to a specific availability zone

-"Think of them as a network USB stick"

-It's a network drive (i.e. not a physical drive)

-Uses the network to communicate the instance, which means a bit of latency

-Can be detached from an EC2 instance and be attached to another one quickly in cases of failover.

-An EBS can stay unattached to any EC2 Instances

-An EBS Volume in another AZ cannot be attached to another AZ

-Moving a volume across needs you to do a snapshot of it

-Must have a provisioned capacity (size in GBs, and IOPS)

-You get billed for all the provisioned capacity

-You can increase the capacity overtime.

EBS Delete on Termination Attribute

-Controls the EBS behavior when an instance is terminated

-By default, the root EBS volume is deleted (attribute enabled)

-By default, any other attached EBS volume is not deleted (attribute disabled)

-Can be controlled by the AWS console / AWS CLI

-Used to preserve root volume when instance is terminated

|=============|

|EBS Snapshots|

|=============|

-Make a backup (snapshot) of your EBS volume at a point in time

-Not necessary to detach volume to do snapshot, but recommended

-Can copy snapshots across AZ or region

EBS Snapshots Features

EBS Snapshot Archive

-Move a snapshot to an "archive tier" that is 75% cheaper

-Takes within 24 to 72 hours for restoring the archive

Recycle Bin for EBS Snapshots

-Setup rules to retain deleted snapshots s oyou can recover them after an accidental deletion

-Specify retention (from 1 day to 1 year)

Fast Snapshot Restore (FSR)

-Force full initialization of snapshot to have no latency on the first use (costs a lot of money)

|==================|

|EC2 Instance Store|

|==================|

-EBS Volumes are network drives with good but "limited" performance

-EC2 Instance store are more high-performance hardware disk.

-Better I/O performance

-EC2 Instance Store lose their storage if they're stopped (ephemeral)

-Good for buffer/cache/scratch data/temporary content

-Risk of data loss if hardware fails

-Backups and replication are your responsibility

|=================|

|EC2 Image Builder|

|=================|

-Enables automation of creation, management, and deployement of up-to-date and compliant golden VM images.

-Enforces version control

-Validates AMI images with tests provided by AWS or your own tests.

|================|

|EBS Volume Types|

|================|

EBS Volumes come in 6 types

-gp2/gp3 (SSD): General Purpose SSD Volume that balances price and performance for a wide variety of workloads

-io1/io2 (SSD): Highest-performance SSD volume for mission-critical low-latency or high-throughput workloads

-st1 (HDD): Low cost HDD volume designed for frequently accessed, throughput-intensive workloads

-sc1 (HDD): Lowest cost HDD volume designed for less frequently accessed workloads

-EBS Volumes are characterized in Size | Throughput | IOPS (I/O Ops Per Sec)

-Only gp2/gp3 and io1/io2 can be used as boot volumes

EBS Volume Types Use Cases

General Purpose SSD

-Cost effective storage, low-latency

-gp3: Newer generation of volumes

-gp2: An older generation of volumes

Provisioned IOPS (PIOPS) SSD

-Critical business applications with sustained IOPS

-Great for database workloads (sensitive to storage performance and consistency)

-io2: Newer generation (4 GiB - 16TiB)

-io2 Block Express (4 GiB - 64TiB)

-Supports EBS Multi-attach

Hard Disk Drives (HDD)

-Cannot be a boot volume

-st1: Throughput Optimized HDD - Big Data, Data Warehouses

-sc1: Cold HDD - For data that is infrequently accessed, archive, scenarios where lowest cost is important.

EBS Multi Attach - io1/io2 family

-Attach the same EBS volume to multiple EC2 instances in the same AZ

-Each instance has full read & write permissions to the high-performance volume

-Used for achieving higher application availability in clustered Linux applications

-Applications must manage concurrent write operations

-Up to 16 EC2 Instances at a time

-Must use a file system that's cluster-aware (not XFS, EX4, etc..)

EBS Encryption

-Creating an encrypted EBS volume gives you:

-Data at rest is encrypted inside the volume

-All the data in flight moving between the instance and the volume is encrypted

-All snapshots are encrypted

-All volumes created from the snapshot are encrypted

-Encryption and decryption are handled transparently (you have nothing to do)

-Encryption has minimal impact on latency

-EBS Encryption leverages keys from KMS (AES-256)

-Copying an unencrypted snapshot allows encryption

|=======|

|AWS EFS|

|=======|

-Managed NFS (Network File System) that can be mounted on many EC2 instances

-EFS works with EC2 Instances in Multi-AZ

-Highly available, scalable, expensive (3x gp2), pay per use

-All EC2 Instances from different AZs can connect to the same NFS through EFS

-Used for content management, web serving, data sharing, Wordpress

-Uses NFSv4.1 protocol

-Uses security group to control access to EFS

-Only Compatible with Linux based AMI (not Windows)

-Encryption at rest using KMS

-POSIX file system(Linux) that has a standard file API

-File system scales automatically, pay-per-use, no capacity planning needed.

EFS - Performance & Storage Classes

EFS Scale:

-1000s of concurrent NFS clients, 10GB+/s throughput

-Grow to Petabyte-scale NFS, automatically

Performance Mode (set at EFS creation time)

-General Purpose(Default): latency-sensitive use cases (web server, CMS, etc.)

-Max I/O - Higher latency, throughput, high parallel (big data, media processing)

Throughput mode:

-Bursting (1 TB = 50MiB/s + burst of up to 100MiB/s)

Provisioned: Set your throughput regardless of storage size, ex: 1GiB/s for 1 TB storage

EFS Storage Classes

-Storage Tiers (life cycle management feature - move file after N days)(

-Standard: for frequently accessed files

-Infrequent access (EFS-IA): cost to retrieve files, lower price to store. Enable EFS-IA with a Lifecycle Policy.

-Availability and Durability:

Standard: Multi-AZ, great for production

One Zone: One AZ, great for dev, backup enabled by default, compatible with IA (EFS One Zone-IA)

|=========================|

|Network File System (NFS)|

|=========================|

-Is a distributed file system protocol that lets users access files over a network similar to the way they access a local storage.

-NFS users can "mount" a portion of the file system on a server, and those files are designated as an additional drive in the user's operating system and become accessible locally.

|=========|

|Amazon S3|

|=========|

Types of S3 Encryption:

Server-side encryption - Amazon S3 encrypts objects before it saves the objects to disk, and decrypts the objects when you download them

Client-side encryption - Amazon S3 encrypt data on the client side and upload the encrypted data to Amazon S3. In this case, you manage the encryption process

Versioning

-Versioning is disabled by default

-Similar to github versioning

-New version of data uploaded every upload

-Versioning cannot be disabled once enabled, but can be suspended

-Three possible states for S3 bucket versioning:

Default: Versioning not enabled

Versioning-enabled

Versioning-suspended

AWS S3 Support for Cross-Origin Resource Sharing (CORS)

-Enables access of external websites to AWS S3 files

-Enables interaction from a client web application with resources in a different domain.

-Enables building rich client-side applications with AWS S3

Cross Region Replication - Objects that are uploaded to a bucket in one region will be automatically copied to other S3 buckets in other regions.

Multipart Upload - Is a feature that enables uploading of large files over 5MB Size.

-Recommended to be used when your object size reaches 100MB.

-Enables you to upload files over 5TB in size over the network.

-Enables quick recovery of uploading large files since files are split into multiple smaller files.

-Enables pausing and resume uploading of objects without expiry. You can decide when to upload a part of an object.

|===============================|

|Amazon S3 Transfer Acceleration|

|===============================|

-Enables acceleration of Amazon S3 Data transfers

-50-500% speed improvement for cross-country transfer of larger objects

-Useful if you have customers all around the world that upload to a centralized bucket.

-Useful if you upload TBs of data across continents

-Takes advantage of Amazon CloudFront and AWS Edge Locations

-Can go even higher under certain conditions

Amazon S3 Transfer Acceleration Speed Comparison Tool

-Shows speed advantage gained (by region)

|============|

|AWS Snowball|

|============|

-Enables transport of multiple terabytes of data into or out of Amazon S3

-Enables usage of multiple devices to transfer petabytes

-AWS Snowball provides a physical device to do the data transfer

-No need to code or purchase hardware

|==============|

|AWS Snowmobile|

|==============|

-Even larger operation than AWS Snowball

-Enables transport of Exabyte (1 million TBs)

-An actual physical data transport shipping container pulled by a semi-trailer truck

-Offers multiple layers of security, such as dedicated security pesonnel, GPS tracking, data encrypted with 256-bit encryption keys.

-100 TB data transfer per snowmobile

|=======|

|AWS RDS|

|=======|

-Works well for applications that:

Have more compelx data

Need ot combine and join datasets

Need enforced syntax rules

-Supported by:

Microsoft SQL Server

PostgreSQL

Oracle

Aurora

MySQL

MariaDB

|======================|

|AWS RDS Instance Types|

|======================|

-T Family: Good for unpredictable or spiky CPU demands aka(Bursting).

-M Family: General purpose with additional options for CPU intensive workloads.

-R Family: Memory-optimized workloads.

|================|

|AWS Read Replica|

|================|

-Updates that are made to the source or primary database are asynchronously copied to the read replica.

-Enhances performance by reducing the load from the source by routing load to the read replica.

-Cannot enable encryption once a database is created, thus read replica cannot also change encryption settings.

-Increases availability

-Can become a standalone Database when needed.

-Each database engines allow replication of up to five read replicas per primary database.

|=============|

|Amazon Aurora|

|=============|

-Performance of traditional database and simplicity of open-source databases.

-Up to five times the throughput of MySQL

-Up to three times the throughput of PostgreSQL

-Replicates data six ways across three Available Zones

-Requires little change to application code.

|===============|

|Amazon Redshift|

|===============|

-Does not run on Amazon RDS

-Powers petabyte scale data warehouse and data lake analytics workloads.

-Provides consistent fast performance

-Use by companies to store frequently-accessed highly structured-data.

-Can also access semi-structured, unstructured data in Amazon S3.

|===============|

|Amazon DynamoDB|

|===============|

-Serverless

-Encrypts all data at rest by default

-Supports multi-region replication

-Handles full backups of data with no performance impact

-Works well for applications that:

Have simple high-volume data

Mst scale quickly

Don't need complex joins

Require ultra-high throughput and low latency

Key features:

NoSQL tables

Items can have differing attributes

In-memory caching

Support for peaks of more than 20 million requests per second

|=============================|

|AWS DynamoDB Read Consistency|

|=============================|

-How and when the result of a read operation is the same data that you wrote

-DynamoDB stores multiple copies of a data

-Reading from DynamoDB means reading from one of the multiple copies.

Eventually Consistent Read

-If the data you have written is not yet updated, and you have read from the same copy, then there will be no updates to the data yet. However, after a second it will be updated and you just need to re-read from the DB again.

Strongly Consistent Read

-An optional feature that makes DynamoDB returns the most up-to-date data.

-Not available during network delay or outage. You must wait for all write operations to complete, it takes longer than a eventually consistent read.

-Consumes more resources.

|===================================|

|AWS Database Migration Service(DMS)|

|===================================|

-Enables migration and replication of existing databases to Amazon RDS.

-Supports migration between widely used databases.

-Enables one-time migrations

-Accomplish continuous data replication

-AWS Schema Conversion Tool (AWS SCT)

Supports changing the database engine between source and target

Homogenous Migrations

-Database migrations between databases that use the same engine.

Heteregenous Migrations

-Database migrations between databases that uses different engines.

Ex. Mysql database that runs on EC2 Instance is migrated to Amazon Aurora

|=================|

|AWS Snowball Edge|

|=================|

-Provides a piece of hardware known as an Edge device to transfer data to the cloud faster than you could through the network.

-Edge Device is owned by AWS. AWS sends you the device and you store data to it, and send it back.

-Edge device provides large amounts of onboard storage, encrypts your data, and ensures both security.

-AWS automatically loads the data into an S3 Bucket.

-AWS DMS then migrates the data from the S3 bucket to the target database.

|=======|

|AWS VPC|

|=======|

-Enables you to provision a logically isolation section of the AWS Cloud.

-You can launch AWS resources in your own VPC.

-Ability to select own IP address range, create subnets, etc.

-Can use both ipv4 and ipv6

-Can customize the network configuration for VPC.

VPC Deployement

-A VPC only belongs to a single AWS Region.

-VPC spans all Availability Zones in a region.

Classless Inter-Domain Routing (CIDR)

-Set of private IP address that you want instances in your VPC to use.

-Default/Primary CIDR Block 10.0.0.0/16 of VPC

-Can assign block sizes between /28 or 16 IP address and /16 or 65,536 IP addresses

-By default, all VPCs and subnets must have IPv4 CIDR blocks.

-Can optionally associate an IPv6 CIDR block with VPC.

-Your VPC can enable dual stack mode where your resources can communicate between IPv4 and IPv6

Subnets: Dividing your VPC

-Is a segment or partition of a VPC's IP address range where you can allocate a group of resources.

-Subnets are not isolation boundaries, instead they are containers for routing policies.

-Subnets are a subset of the VPC CIDR block

-Subnet CIDR blocks cannot overlap

-Each subnet resides entirely within one Availability Zone

-You can add one or more subnets in each Availability Zone or in a Local Zone, which enables distribution of EC2 instances across multiple locations

-AWS reserves five IP addresses in each subnet

VPC design best practices

-Create one subnet per available Availability Zone for each group of hosts

-Divide your VPC network range evenly across all availability Zones in a Region.

-Do not allocate all network addresses at once. Instead, ensrue that you reserve some address

Single VPC deployement

-One VPC for small applications

Multiple VPC deployement

-Works best for a single team or organization that maintains full control over the resources being used in each application environment.

Multiple accounts for VPC

-Works best for large organizations with multiple teams working on different parts of an application.

-For example, if an organization have two teams or more teams. This pattern can be used to support developers who have full access to development environment resources, but limited or no access to the production environment.

Amazon VPC Quotas

-Up to 5 VPCs per AWS Region per account.

-Can request a quota increase.

Internet Gateways

-Allows communication between resources in your VPC and the internet

-Horizontally scaled, redunant, and highly available by default

-Provides a target in your subnet route tables for internet-routable traffic.

Directing traffic between VPC resources

-Route tables are required to direct traffic between VPC resources

-A main route is created automatically when a VPC is created

-Every route table contains only a single local route.

-Local route enables communication for all the resources in the VPC.

-Local route cannot be modified

-Launching an instance in the VPC is automatically covered by the Local route

-Each VPC has a main (default) route table

-All subnets must be associated with a route table

-You can create a custom route tables

Best practice: Use custom route tables for each subnet.

Elastic IP Address

-Is a static, public IPv4 addresses associated with your AWS account

-Can be associated with an instance or elastic network interface

-Resilient, you can mask the failure of an instance by rapidly remapping the address to another instance in your VPC

Connecting private Subnets to the Internet

NAT Gateways

-Enable instances in a private subnet to initiate outbound traffic to the internet or other AWS services(Unless a private instance initiates a connection to the Internet, the Internet cannot initiate a connection to the private instance)

-Prevent private instances from receiving inbound connection requests from the internet.

-An Elastic IP address is required to associate with the NAT gateway

Bastion Hosts

-A server whose purpose is to provide access to a private network from an external network

-Minimizes the chances of penetration

Security Groups

-Stateful firewalls that control inbound and outbound traffic to AWS resources.

Stateful means that state information is kept even after a request is processed. Sending a request from instance, the response traffic for that request is allowed to flow in regardless of inbound security group rules.

Responses to allowed inbound traffic are allowed to flow out, regardless of outbound rules.

-Act at the level of the instance or network interface

Network Access Control Lists (Network ACL)

-Act at the subnet level

It controls traffic going into and out of one or more subnets.

-VPC automatically comes with a modifiable default network ACL.

-Allows all inbound and outbound traffic by default

-Allows setting up of rules similar to security groups.

-Has separate inbound and outbund rules.

-Each subnet in VPC must be associated with a network ACL, and can be associated with only one network ACL at a time.

-Can associate a network ACL with multiple subnets.

-Are stateless firewalls that require explicit rules for both inbound and outbound traffic

Good for: Specific network security requirements

Best practice: Secure infrastructure with multiple defense by running structure into a VPC, and then further protection with security groups and define network ACLs to further protect. Also can secure by adding a firewall at the operating system level.

|====================|

|AWS Site-to-Site VPN|

|====================|

-A highly available solution that enables you to securely connect your on-premises network or branch office site to your VPC.

-Uses internet protocol security (IPSec) communications to create encrypted Virtual Private Network(VPN) tunnels.

-A VPN Tunnel is an encrypted link where data passes between the customer network and the VPC.

-Provides two encrypted tunnels per VPN connection across multiple AZs

-You can use these encrypted tunnels at the same time for high availability.

-Charged per VPN connection-hour where your VPN is provisioned and available.

The AWS side of the connection is the Virtual Private Gateway(VPG)

The on-premises side of the connection is the on-premises the customer gateway

Static and dynamic routing

Static routing

-Requires you to specify all routes (IP prefixes)

-Specify static routing if your customer gateway device does not support Border Gateway Protocol(BGP)

-Supports 50 non-propagated routes per route table by default, up to a maximum of 1,000 non-propagated routes

Dynamic routing

-Uses the BGP to advertise its routes to the VPG

-Specify dynamic routing if your customer gateway device supports BGP

-Supports a maximum of 100 propagated routes per route table

Recommended to use BGP-cable devices since the BGP protocol offers robust liveness dection checks that can assist failover to the second VPN tunnel if the first tunnel fails.

Devices that don't support BGP may also perform health checks to assist failover to the second tunnel when needed.

|================|

|AWS VPN CloudHub|

|================|

-Allows establishing of multiple VPN connections from multiple customer gateway devices to a single VPG. This implements redundancy and failovers.

|=======================|

|AWS Direct Connect (DX)|

|=======================|

-Can be deployed in multiple AWS regions, but will require at at least multiple DX locations in at least two regions.

-Another solution to connect on-premise servers to VPCs

-Provides you with a dedicated, private network connection capacity of either 1 Gbps or 10 Gbps from your premises to AWS.

-Uses open standard 802.1q Virtual local area Network (VLANs)

-Reduces network costs, increased bandwidth throughput, and provide a more consistent network experience than internet-based connections.

Used for:

Hybrid environments - On-premises resources, while taking advantage of elasticity and other benefits of AWS

Transferring large datasets

Network performance predictabiliy

Security and compliance

-Since you're reducing the internet bandwidth used by applications, you can reduce the network fees you pay to your internet provider.

-All data that's transferred over AWS DX is charged at the reduced Direct Connect data transfer rates instead of internet data transfer rates which can lower network costs.

Conecting VPCs

-Isolating some of your workloads is generally a good practice

VPC peering

-Allows routing of traffic between VPCs privately

-Allows communication of instances from separate VPCs as if they are in the same network

-Ability to connect a VPC with another VPC from another AWS account, or a VPC in a different AWS region.

-One-to-One networking connection between two VPCs

-Cloud resources can communicate with each other using private IP address without gateways, VPN connections, and separate network appliances needed

-No single point of failure or bandwidth bottleneck.

Inter-Region VPC peering

-Provides a simple and cost-effective way to share resources between Regions replicate data for geographic redundancy.

-All traffic is encrypted, with no single point of failure or bandwidth bottleneck.

-Traffic never goes public

VPC Peering Connection Restrictions

-Use private IP addresses

-Can be established between different AWS accounts

-Cannot have overlapping CIDR blocks

-Can have only one peering resource between any two VPCs

-Do not support transitive peering relationships

|===================|

|AWS Transit Gateway|

|===================|

-Enables connection to VPCs and on-premises networks to a single gateway

-Fully managed, highly available, flexible routing service

-Acts as a hub for all traffic to flow through between your networks

-Connects up to 5,000 VPCs and on-premises environment with a single gateway

-Uses a hub-and-spoke model

-Can also be used to achieve isolation in your VPC environment.

|==========================|

|Two types of VPC endpoints|

|==========================|

Interface endpoint - An elastic network interface with a private IP address that serves as an entry point for traffic destined to a supported service.

Powered by AWS Private Link

Examples:

-Amazon CloudWatch

-Amazon EC2 API

-Elastic Load Blaancing

Gateway endpoint - Gateway specified as a target for route in your route table for traffic destined to a supported AWS service

Supported AWS Services:

-Amazon S3

-Amazon DynamoDB

|=======|

|AWS IAM|

|=======|

IAM User - Permissions given to users to interact with AWS resources

-Must be unique name without spaces

-Given security credentials

IAM Group - Collection of IAM users to simplify permission managing for multiple users

IAM Policy - Permissions which resources can be accessed for groups and users

-Written in JSON

Two types of policies:

Identity-based - Attach to an IAM principal, IAM users, IAM roles, groups or roles

Types of Identity-based policies:

AWS managed

Customer Managed - Created by the user

Inline - Created by the user but policy is attached to a user, role and groups directly

Resource-based: Attach to an AWS resource

-Policy always denies a permission by default unless if explicitly allowed

-Always inline policy

IAM role - Granting temporary access to specific AWS services

-Can be given to a user, application, or a service.

IAM Policy document structure

Effect: Can be either Allow or Deny

Action: Type of access that is allowed or denied

Resource: Resources that the action will act on

Condition: Conditions that must be met for the policy to apply

ARNs and wildcards

Resources are identified by using Amazon Resource Name(ARN) format

Syntax-arn:partition:service:region:account:resource

Example: "Resource": "arn:aws:iam:12345678912:user/mmajor"

Wildcard gives access to all actions for a specific AWS service

Examples:

"Action": "s3:\*"

"Action": "iam:\*AccessKey\*"

|==============|

|AWS CloudTrail|

|==============|

-Logs and monitors user

-Provides event history of AWS account

Actions taken through the AWS Management Console, SDK, AWS CLi

90-day event history provided by default, at no cost

-Identify who accessed your account, when and from where, what action they took

-Helpful to perform security analysis, and discover which calls were blocked

IAM Role

-Provides temporary security credentials

-Can be assumable by a person, application or service

-Not uniquely associated with one person

Use cases:

-Provide AWS resources with access to AWS services

-Provide access to externally authneticated users

-Provide access to third parties

|====================================|

|AWS Security Token Service (AWS STS)|

|====================================|

-IAM users, applications and services can use AWS STS to make it request that allows them to assume a role.

-If IAM document is approved, then AWS STS will process the granting of temporary and limited privileged credentials the user requests.

Access Control Approaches

Role-based access control (RBAC)

-Traditional approach

-Create an IAM role for each permission combination needed

-Create roles for different users in different groups to assume roles based on their group

Ex:

Create developer roles for developers to assume, network admin roles for network admin to assume. If user is the same, grant the user the access to assume both roles.

Disadvantage: Need to update roles if new resources are added

Best practice: Tagging

-Makes it easier to find, manage and filter resources by adding tag to these resources

Attribute-based access control (ABAC)

-Using of tags as attributes and define permissions based on the existence or nonexistence of the tag attributes on a resource.

-Tags are then cross-matched to the tags of the resource and the tags of the user if they match properly. Afterwards, the user is then granted access to the resource.

-Policy documents are more straight forward

-Permissions automatically apply, based on attributes

-No need to list specific resources in the policy file.

-No need to modfiy permission settings when new resources are added unlike RBAC

AWS Cognito

-Fully managed service

-Provides authentication, authorization, and user management for web and mobile apps

-Provides web identity federation

They can be used as the identity broker that supports IdPs that are compatible with OpenID Connect(OIDC)

-Federated identities

Users sign in with social identity providers (Amazon, Facebook, Google or with SAML)

-Can maintain a directory with user profiles authentication tokens from Federating IdP

==========|

Risks from Manual Proccesses (Manually creating systems and infrastructures from scratch)

==========|

Does not support repeatability at scale

-How will you replicate deployements to multiple regions?

No version control

-How will you rollback the production environment to a prior version?

Lack of audit trails

-How will you ensrue compliance? How will you track changes to configuration details at the resource level?

Inconsistent data management

-For example, how will you ensure matching configurations across multiple AWS EC2 instances

|==================|

|AWS CloudFormation|

|==================|

-Provides a simplified way to model, create, and manage a collection of AWS resources

No extra charge, pay only for resources you create

Collection of resources is called an AWS CloudFormation stack

-Can create, update and delete stacks

-Enables version control of AWS resource deployements

|============================|

|Infrastructure as Code (IaC)|

|============================|

-Process of provisioning and managing your cloud resources by writing a template file that is human readable and machine consumable

-It is infrastructure you can replicate, de-deploy, re-purpose

-You can roll back to the last good state on failures

Benefits:

Consistency from test environment to deployement environment because of same template used

Easily clean-up resources not used and/or outdated resources

-Can be written in:

YAML - Much easier to debug because of no commas, and other syntax-dependent characters

JSON - Much more used with other computer systems

|===================|

|AWS Systems Manager|

|===================|

-Automates operational tasks

Apply OS patches and software upgrades across a fleet of EC2 instances

-Manages servers on-premises and in the cloud

-Allows creation of system images

|============|

|AWS OpsWorks|

|============|

-A configuration management service

-Automate how servers are configured, deployed, and managed

-Provides managed instances of Chef and Puppet

Chef and Puppet are popular automation platforms

|=====================|

|AWS Elastic Beanstalk|

|=====================|

-Easy way to get web applications up and running

-Managed service that automatically handles:

Infrastructure provisioning and configuration

Deployement

Load balancing

Automatic scaling

Health monitoring

Logging

-No additional charge for using it

-Upload code and Elastic Beanstalk automatically handles deployement

|========================|

|Content Delivery Network|

|========================|

-Globally distributed system of caching servers

-Caches copies of commonly requested files (static content)

-Delivers a local copy of the requested content from a nearby cache edge or Point of Presence

|==============|

|AWS CloudFront|

|==============|

-Amazon Global CDN

-Optimized for all delivery use cases with a multi-tier cache by default

-Provides extra layer of security for architectures

-Supports WebSockets and HTTP or HTTPS methods

|==================================|

|AWS Web Application Firewall (WAF)|

|==================================|

-Enables monitoring of HTTP and HTTPS requests that are forwarded to Amazon API Gateway API, CloudFront, or Application Load Balancer(ALB)

-Enables control to access content

|=================================|

|AWS DynamoDB Accelerator (AWS DAX|

|=================================|

-Fully managed, highly available, in-memory cache for DynamoDB

-Performance improvement up to 10 times - from miliseconds to microseconds-even at millons of requests per second.

|==========|

|Side Cache|

|==========|

-Typically used for read-heavy workloads

-Isn't directly connected to the database-instead, it's used adjacently to the database

-Typically built on key-value NoSQL stores such as Redis of Memcached

|===============|

|AWS ElastiCache|

|===============|

-Is a side cache that works as an in-memory data store to support the most demanding applications

-Does not require management tasks such as hardware provisioning, software patching, etc.

-Provides web applications with in-memory data store in the cloud

-Offers high performance

-Supports Redis and Memcached

-Can scale out and scale in to meet fluctating application demands

-Supports sharding

-Write and memory scaling with sharding

-Supports AWS VPC which enables isolation of cluster to the IP ranges chosen for nodes

ElastiCache for Memchad

-Scale up to 20 nodes per cluster

ElastiCache for Redis

-Scales up to 250 nodes per cluster

|==========================|

|AWS ElastiCache components|

|==========================|

Cache node - is the smallest building block of an ElastiCache deployement

-Fixed-size chunk of secure, network-attached RAM

-Each node has its own DNS name and port

-A group of nodes is called a cluster

Caching Strategies

Lazy loading - Loads data into the cache only when necessary

-Use lazy loading when data will be read often, but written infrequently

-For example, a user's profile that rarely changes but is accessed throughout the app

Write-through - Adds or updates data in the cache when data is written to the database

-Use when data must be updated in real time

-This approach is proactive: avoid unnecessary cache misses when you have data that you know is going to be accessed such as top-100-game leaderboard, recommendations, etc.

-Increases likelihood that application will find that value in the cache when it looks for it

-Disadvantage is potentially caching data that is unneeded, which can cause added costs

Time to live(TTL) - An integer value or key that specifies the number of seconds or miliseconds, depending on the in-memory engine, until the key expires.

-When an application attempts to read an expired key, it's treated as though the data isn't found in the cache.

|=======|

|AWS SQS|

|=======|

-Fully managed message queueing service

-Uses a pull mechanism

-Messages are encrypted and stored until they are processed and deleted

-Acts as a buffer between producers and consumers

|===============|

|System Coupling|

|===============|

Application-level coupling - relates to managing incoming and outgoing dependencies.

Platform coupling relates - to the interoperability of heterogeneous systems. components

Spatial coupling - relates to managing components at a network-topology level or protocol level.

Temporal(or runtime) - coupling refers to the ability of a system component to do meaningful work while it performs a synchronous, blocking operation.

Producer - Application component that produces messages and adds them to queue

Consumer - Application component that polls queue for messages and processes them

Message - for communication between software components - not between people

-No single computer, network, or AZ failure can make messages inaccessible.

-Application can easily recover from failed steps because messages will remain in the queue

Message Queue - A temporary repository for messages that are waiting to be processed

-Usually small, can be things like requests, replies, error messages, or plain information

Process:

Producer sends a message, message queue takes in message and wait for consumer to take and process the message, consumer processes the message and deletes the message afterwards

AWS SQS general use cases

Work queues - Decouple components of a distributed application that might not all process the same amount of work simultaneously

Buffering batch operations - Add scalability and reliability to your architecture, and smooth out temporary vo

Queue Types

Standard Queues

At-least-once delivery - A message is delivered at least once, but occasionally more than one copy of a message is delivered

Best effort ordering - Occasionally, messages might be delivered in an order that is different from the order they were sent in.

Nearly unlimited throughput - Standard queues support a nearly unlimited number of Transactions per Section (TPS) per API action.

First in, first out (FIFO) queues

Are designated to guarantee that messages are processed exactly once, in the exact order that they are sent and received.

Provide high throughput - FIFO queues support up to 300 messages per second (300 send, receive, or delete operations per second). When you batch 10 messages per operation (maximum), FIFO queues can support up to 3,000 messages per second.

AWS SQS Features

Dead-letter queue (DQL) - A queue of messages that could not be processed

-Receives messages after the maximum number of processing attempts has been reached.

Visibility timeout - The period of time when AWS SQS prevents other consumers from receiving and processing the same message.

-Ensures that a job does not get processed multiple times and cause duplication.

-During visibility timeout, the component that received the message processes it and then deletes it from teh queue.

-If the consumer fails to process and delete the message before the visibility timeout expires, the message becomes visible to other consumers and it might be processed again.

Short Polling - Queries only a subset of the servers to find messages that can be included in the response.

Long Polling - Queries all the servers for messages.

-AWS SQS sends the response after it collects the maximum number of messages for the response, or the polling wait time expires.

-Inexpensive to retrieve messages from SQS queue as soon as the messages are available.

-Might reduce the cost of using AWS SQS because of reduced number of empty receives.

|===================================|

|AWS Simple Notification Service SNS|

|===================================|

-Is publisher/subscriber messaging paradigm

-Highly available, durable, secure, and fully managed pub/sub messaging service

-Enables applications to publish an unlimited number of messages/topics at any time

-Create a topic and set policies that restrict who can pulish or subscribe to the topic

-Publisher sends messages to topics that they have either created or that they have permission to publish to.

AWS SNS Supports transport protocols for delivering messages:

Email

HTTP or HTTPS

Short Message Service (SMS)

AWS SQS queues - Users specify an SQS standard queue as the endpoint.

-SNS will enqueue a notification message to the specified queue.

AWS Lambda functions - Messages are delivered to AWS Lambda functions, which handle message customizations, enable message persistence or communication with other AWS services

General use cases for AWS SNS

Application and system alerts - Use AWS SNS to receive immediate notification when an event occurs, such as a change to an AWS Auto Scaling group.

Push email and text messaging - Use AWS SNS to push a targeted news headliens to subscribers by email or SMS

Mobile push notifications - Use AWS SNS to send notifications to an application, indicating that an update is available.

|======|

|AWS MQ|

|======|

-A managed message broker service for Apache ActiveMQ

-Manages provisioning, setup, and maintenance of ActiveMQ

-Simplifies message migration to the cloud

-Compatible with open-standard APIs and protocols

Message Brokers - software that facilitates the exchange of messages between applications, systems, and services. It allows linked systems to communicate directly, even if they're built using different technologies, deployed on separate platforms, or use distinct messaging protocols.

|=============|

|Microservices|

|=============|

-Applications that are composed of independent services that communicate over well-defined APIs

-Improves maintainability and scalability of software applications

Common characteristics of Microservices:

Decentralized - Microservice architectures are distributed systems with decentralized management. They dont' rely on a unifying schema in a central database. Each microservice has its own view about data models.

Independent - Each component service can be changed, upgraded or replaced independently without affect the function of other services. Services do not need to share any of their code or implementation with other services.

Specialized - Each component service is designed for set capabilities and focuses on a specific domain.

Polyglot - Microservices don't follow a single approach. Teams have the freedom to choose the best tool for their specific problem.

Black boxes - Individual component services are designed as black boxes and their details of complexity are hidden from other components.

|=========|

|Container|

|=========|

-A method of operating system virtualization that enables you to run an application and its dependencies in resource-isolated processes.

-Are lightweight, standalone software package.

-Contains everything that a software appliaction needs to run such as the application code, runtime engine, system tools, system libraries, and configurations

A problem that containers solve

-Getting software to run reliably in different work environments

-Ensures that applications deploy quickly, reliably, and consistently, regardless of deployment environments.

Container terminology

Image - Used to create a container using a read-only template.

-Typically built from a Dockerfile, which is a plaintext file that specifies all the components that are included in the container.

-Can create images from scratch, or use images that others created

Container image - Is the snapshot of the file system that is available to the container.

-For example, having a Debian OS as a container image means the container will have a Debian OS when run.

Registry - Where container images are stored.

-Can download the images from the registry and run on cluster.

-Registries can exist in or outside AWS infrastructure.

|===================================|

|AWS Elastic Container Service (ECS)|

|===================================|

-Orchestrates when containers run

-Maintains and scales the fleet of instances that run your containers

-Removes complexity of standing up the infrastructure

-Highly scalable, high-performance, container-management service

-Supports Docker containers and enables you to easily run applications on a managed cluster of AWS EC2 instances

-Can be launched with either AWS Fargate or AWS EC2

|=============|

|AWS Cloud Map|

|=============|

-A fully managed discovery service for cloud resources

-Can be used to define custom names for application resources

-Maintains updated location of dynamically changing resources, which increases application availability

|============|

|AWS App Mesh|

|============|

-A service mesh that provides application-level networking to make it easy for your services to communicate with each other across multiple types of compute infrastructure

-Captures metrics, logs, and traces from all your microservices

-Enables exporting of this data to Amazon CloudWatch, AWS X-Ray, and compatible AWS Partner Network

|===========|

|AWS Fargate|

|===========|

-Fully managed container service

-Works with AWS and Amazon Elastic Kubernetes Service (AWS EKS)

-Provisions, manages, and scales your container clusters

-Provides automatic scaling

-Manages runtime environment

-Enables developer to focus on designing and building applications instead of managing the infrastructure that runs them

|==========|

|Serverless|

|==========|

-A way to build and run applications and services without thinking about servers

-Native architecture of the cloud that shifts operational responsibilities to AWS, which can increase agility and innovation

-Application still runs on servers, however, AWS does all the server management tasks such as server or cluster provisioning, patching, operating system maintenance, and capacity provisioning.

Tenets of serverless architectures

-No infrastructure provisioning, no management

-Automatic scaling (Scales by uinit of work or consumption rather than by sertver unit)

-Pay for value (Pay only for the duration that a resource runs, rather than by server unit)

-Highly available and secure (Built-in availability and fault tolerance)

|==========|

|AWS Lambda|

|==========|

-Fully managed compute service

-Runs code on a schedule or in response to events(for example, changes to an AWS S3 bucket or an AWS DynamoDB table)

-Supports Java, Go, C#, etc.

-Can run at edge locations closer to your users

-Handles all administration of the compute resources, including server and operating system maintenance, capacity provisioning, automatic scaling, etc.

Lambda Function - custom code that you write in one of the languages that Lambda supports.

-Lambda functions are stateless, which means that they have no affinity to the underlying infrastructure.

-Can rapidly as many copies of the function needed to scale to the rate of incoming events

|===============|

|AWS API Gateway|

|===============|

-Enables creation, publish, maintain, monitor and secure APIs that act as entry points to backend resources

-Can be used to create RESTful and WebSocket APIs that act as entry points for apps to access backend resources

-Handles up to hundreds of thousands of concurrent API calls

-Can host and use multiple version and stages of API

|=============================|

|Recovery Point Objective(RPO)|

|=============================|

-Is the maximum acceptable amount of data loss, measured in time.

-For example, the business can recover from losing (at most) the last 8 hours of data.

-Organizations of all sizes often have a Business Continuity Plan (BCP) which revolves around providing IT Service Continuity, including IT disaster recovery planning

-To calculate RPO, first determine how much data loss is acceptable, according to BCP. Then, figure out how quickly that data loss

|============================|

|Recovery Time Objective(RTO)|

|============================|

-Maximum acceptable amount of time after disaster strikes that a business process can remain out of commission

-The time it takes after a disruption to restore your applications and recover data.

|============|

|AWS DataSync|

|============|

-Provides movement of large amounts of data online between on-premises storage and AWS S3, AWS EFS, or AWS FSx for Windows File Server.

-Supports scripted copy jobs and schedule data transfers from on-premises NFS and Server Message Block(SMB) storage.

-Can optionally use AWS Direct Connect links

Best Practice: AWS S3 Cross-Region Replication

-Most S3 storage classes replicate data across Availability Zones within a single Region

-Configuire S3 cross-Region replication for higher-level data security

Automatically, asynchronously replicates objects created after you add the replication configuration

Can also help meet compliance requirements and reduce latency for users who are accessing objects

Best Practice: EBS Volume Snapshots

-Can backup the data that is on EBS volumes to AWS S3 by taking point-in-time snapshots

-Snapshots are incremental backups, which means that it saves only the blocks on the device that have changed since the most recent snapshot.

-Minimizes the time that is required to create the snapshot, and saves on storage by not duplicating data

-Each snapshot contains all the information that is needed to restore data (from teh moment when the snapshot was taken) to a new EBS volume.

-New snapshot acts as an exact replica of the original volume

-Replicated volume loads data in the background for immediate use.

|==========================|

|AWS Data Lifecycle Manager|

|==========================|

-Automates the creation, retention, and deletion of snapshots that backup your EBS volumes.

-Helps protect valuable data by enforcing a regular backup schedule

-Retain backups as required by auditors or internal compliance

-Reduce storage costs by deleting outdated backups

Networking: Design for resilience, recovery

AWS Route 53 - provides load balancing and networking routing capabilities that enable you to distribute network traffic.

Also provides the ability to fail over between multiple endpoints and even to a static website that is hosted in AWS S3.

AWS ELB - automatically distributes incoming application traffic across multiple EC2 instances.

-Enables to achieve fault tolerance in applications by providing the load-balancing capacity that is needed in response to incoming application traffic

AWS VPC - to extend an existing on-premises network topology to the cloud

-Can be appropriate when recovering enterprise applications that might be hosted on an internal network

AWS Direct Connect - simplifies the setup of a dedicated network from an on-premises data center to AWS

-Reduces network costs, increase bandwidth throughput and provide a more consistent network experience than internet-based connections

Databases: Features that support recovery

AWS RDS - STakes snapshot data and save it in a separate Region

-Combine read replicas with Multi-AZ deployements to build a resilient disaster recovery strategy

-Retain automated backups

AWS DynamoDB

-Backup entire tables in seconds

-Use point-in-time-recovery to continuously back up tables for up to 35 days

-Initiate backups with a single click in the console or a single API call

Automation Services; Quickly replicate or redeploy environments

AWS CloudFormation - Use templates to quickly deploy collections of resources as needed

-Duplicate production environments in new Region or VPC in minutes

-Provisions resources in a repeatable manner, which enables building and rebuilding of infrastructure and applications

-Not required to perform manual actions or write custom scripts

AWS Elastic Beanstalk - Quickly redeploy your entire stack in only a few clicks

AWS OpsWorks - Automatic host replacement

-Combine it with AWS CloudFormation in the recovery phase

-Provision a new stack that supports the defined RTO

|==============================|

|Four disater recovery patterns|

|==============================|

Backup and restore

Pilot light

Warm standby

Multi-site

Each pattern is suited to a different combination of:

RPO

RTO

Cost-effectiveness

Backup and restore pattern

-Data is backed up to tape and sent offsite regularly.

-Takes a logn time to restore system when a disaster occurs

-Data from on-premise server sent to AWS storages and then retrieved when recovery is needed

|===================|

|AWS Storage Gateway|

|===================|

-Is a hybrid storage service that enables on-premises applications to use AWS Cloud storage.

-Can use the service for backup and archiving, disaster recovery, cloud processing, storage tiering and migration

-Applications connect to the service through a VM or hardware gateway appliance by using standard storage protocols.

Pilot light pattern

-A minimal backup version of environment is always running.

-Analogy comes from a gas heater: a small flame is always on, even when the heater is off.

-Pilot light can quickly ignite the entire furnace to heat a house.

-Pilot light is the secondary database that is always running

-Similar to backup-and-restore scenario, however recovery time is typcailly faster because the core pieces of the system are already running and are continually kept-up-to-date

-When time comes for recovery, can rapidly provision a full production environment around the critical core

-Pattern is relatively inexpensive to implement

Warm standby pattern

-Similar to the pilot light, but more resources are already running

-A scaled-down version of a fully functional environment is always running in the cloud

-Necessary components running 24/7

-Further decreases the recovery time because some services are always running.

-Can then quickly begin to scale up to handle the production load when AWS Route53 switches over from main system to secondary system.

Multi-Site pattern

-Have a fully functional system that runs in a second Region of AWS.

-Runs at the same time as the on-premises systems or the systems that run in a different AWS region

-Runs in an active-active configuration

-Data replication method that is employed is determined from the recovery point that is chosen

|===================|

|Summary of Patterns|

|===================|

-The backup and restore pattern typically can be accomplished at the lowest cost, but it has a longer RTO. As a result, your systems are likely to be restored more slowly than with the other options.

-The warm standby and multi-site patterns support a much faster RTO, but it is costly to have extra servers that are always running.