AWS Developer Associate Exam Master Cheat Sheet

SKILLCERTPRO

Table of contents

- AWS Fundamentals

- AWS Deep Dive

- AWS Serverless

- Docker based AWS services

  - ECS: Elastic Container Service
  - Elastic Container Registry
  - Fargate

Exam Preparation

- Exam details
  - Two question types:
    - Multiple Choice
    - Multiple response
  - Minimum passing score: 720/1000
  - Domains:
    - Deployment: CICD, Beanstalk, Serverless
    - Security: each service deep-dive + dedicated section
    - Development with AWS Services: Serverless, API, SDK, & CLI
    - Refactoring: Understand all the AWS services for the best migration
    - Monitoring and Troubleshooting: CloudWAtch, CloudTrail, X-Ray
  - Exam Guide:
    - https://aws.amazon.com//certification/certified-developer-certificate/
- EC2 + IAM Exam Checklist
  - Know how to SSH into EC2 (and change .pem file permissions)
  - Know how to properly use security groups
  - Know the fundamental differences between private vs public vs elastic IP
  - Know how to use User Data to customize your instance at boot time
  - Know that you can build custom AMI to enhance your OS
  - EC2 instances are billed by the second and can be easily created and thrown away, welcome to the cloud! Maybe on Exam:
  - Availability zones are in geographically isolated data centers
  - IAM users are NOT defined on a per-region basis
  - If you are getting a permission error exception when trying to SSH into your linux instance, then the key is missing chmod 400 permissions
  - If you are getting a network timeout when trying to SSH into your EC2 instance, then your security groups are misconfigured
  - Security groups reference IP address, CIDR block, Security group, but NOT DNS name

IAM: Identity and Access Management

When accessing AWS, the root account should **never** be used. Users must be created with the proper permissions. IAM is central to AWS.

- Users: A physical person
- Groups: Functions (admin, devops) Teams (engineering, design) which contain a group of users
- Roles: Internal usage within AWS resources
- Policies (JSON documents): Defines what each of the above can and cannot do. **Note**: IAM has predefined managed policies.

For big enterprises:

- IAM Federation: Integrate their own repository of users with IAM using SAML standard

Policies

IAM policies define permissions for an action regardless of the method that you use to perform the operation.

Policy types

- Identity-based policies
  - Attach managed and inline policies to IAM identities (users, groups to which users belong, or roles). Identity-based policies grant permissions to an identity.
- Resource-based policies
  - Attach inline policies to resources. The most common examples of resource-based policies are Amazon S3 bucket policies and IAM role trust policies. Resource-based policies grant permissions to a principal entity that is specified in the policy. Principals can be in the same account as the resource or in other accounts.
- Permissions boundaries

    - Use a managed policy as the permissions boundary for an IAM entity (user or role). That policy defines the maximum permissions that the identity-based policies can grant to an entity, but does not grant permissions. Permissions boundaries do not define the maximum permissions that a resource-based policy can grant to an entity.

- Organizations SCPs
    - Use an AWS Organizations service control policy (SCP) to define the maximum permissions for account members of an organization or organizational unit (OU). SCPs limit permissions that identity-based policies or resource-based policies grant to entities (users or roles) within the account, but do not grant permissions.
- Access control lists (ACLs)
    - Use ACLs to control which principals in other accounts can access the resource to which the ACL is attached. ACLs are similar to resource-based policies, although they are the only policy type that does not use the JSON policy document structure. ACLs are cross-account permissions policies that grant permissions to the specified principal entity. ACLs cannot grant permissions to entities within the same account.
- Session policies
    - Pass advanced session policies when you use the AWS CLI or AWS API to assume a role or a federated user. Session policies limit the permissions that the role or user's identity-based policies grant to the session. Session policies limit permissions for a created session, but do not grant permissions. For more information, see Session Policies.

AWS Policy Simulator

- When creating new custom policies you can test it here:
    - https://policysim.aws.amazon.com/home/index.jsp
    - This policy tool can you save you time in case your custom policy statement's permission is denied
- Alternatively, you can use the CLI:
    - Some AWS CLI commands (not all) contain `--dry-run` option to simulate API calls. This can be used to test permissions.
    - If the command is successful, you'll get the message: `Request would have succeeded, but DryRun flag is set`
    - Otherwise, you'll be getting the message: `An error occurred (UnauthorizedOperation) when calling the {policy_name} operation`

Best practices:

- One IAM User per person **ONLY**
- One IAM Role per Application
- IAM credentials should **NEVER** be shared
- Never write IAM credentials in your code. **EVER**
- Never use the ROOT account except for initial setup
- It's best to give users the minimal amount of permissions to perform their job

EC2: Virtual Machines

By default, your EC2 machine comes with:

- A private IP for the internal AWS Network
- A public IP for the WWW

When you SSH into your EC2 machine:

- We can't use a private IP, because we are not in the same network
- We can only use the public IP

If your machine is stopped and then restarted, the public IP will change

EC2 User Data

- It is possible to bootstrap our instances using an EC2 User data script
- Bootstrapping means launching commands when a machine starts
- That script is only run once at the instance first start
- Purpose: Ec2 data is used to automated boot tasks such as:
  - Installing updates
  - Installing software
  - Downloading common files from the internet
- The EC2 User Data Script runs with the root user

EC2 Meta Data

- Information about your EC2 instance
- It allows EC2 isntances to "learn" about themselves without having to use an IAM role for that purpose
- Powerful but one of the least known features to developers

- You can retrieve IAM roles from the metadata but **not** IAM policies
- URL: {ec2-ip-address}/latest/meta-data

EC2 Instance Launch Types

- **On Demand Instances**: short workload, predictable pricing
- **Reserved Instances**: long workloads (>= 1 year)
- **Convertible Reserved Instances**: long workloads with flexible instances
- **Scheduled Reserved Instances**: launch within time window you reserve
- **Spot Instances**: short workloads, for cheap, can lose instances
- **Dedicated Instances**: no other customers will share your hardware
- **Dedicated Hosts**: book an entire physical server, control instance placement

On Demand Instance:

- Pay for what you use
- Has the highest cost but no upfront payment
- No long term commitment
- Recommended for short-term and un-interrupted workloads, where you can't predict how the application will behave

Reserved Instances

- Up to 75% compared to On-demand
- Pay upfront for what you use with long term commitment
- Reservation period can be 1 or 3 years
- Reserve a specific instance type
- Recommended for steady state usage applications (think database)

Convertible Reserved Instances

- Can change the EC2 instance type
- Up to 54% discount

Scheduled Reserved Instances

- Launch within time window you reserve
- When you require a fraction of a day / week / month

Spot Instances

- Can get a discount of up to 90% compared to On-demand

- You bid a price and get the instance as long as its under the price
- Price varies based on offer and demand
- Spot instances are reclaimed within a 2 minute notification warning when the spot price goes above your bid
- Used for batch jobs, Big Data analysis, or workloads that are resilient to failures
- Not great for critical jobs or databases

Dedicated Instances

- Instances running on hardware that's dedicated to you
- May share hardware with other instances in same account
- No control over instance placement (can move hardware after stop / start)

Dedicated Hosts

- Physical dedicated Ec2 server for your use
- Full control of Ec2 Instance placement
- Visibility into the underlying sockets / physical cores of the hardware
- Allocated for your account for a 3 year period reservation
- More expensive
- Useful for software that have a complicated licensing model (Bring your own License)
- Or for a companies that have strong regulatory or compliance needs

Which host is right for me?

- On demand: coming and staying in resort whenever we like, we pay the full price
- Reserved: like planning ahead and if we plan to stay for a long time, we may get a good discount.
- Spot instances: the hotel allows people to bid for the empty rooms and the highest bidder keeps the rooms.You can get kicked out at any time
- Dedicated Hosts: We book an entire building of the resort

EC2 Pricing

- EC2 instances prices (per hour) varies based on these parameters:
  - Region you're in
  - Instance Type you're using
  - On-Demand vs Spot vs Reserved vs Dedicated Host
  - Linux vs Windows vs Private OS (RHEL, SLES, Windows SQL)
  - You are billed by the second, with a minimum of 60 seconds.

  - You also pay for other factors such as storage, data transfer, fixed IP public addresses, load balancing
  - You do not pay for the instance if the instance is stopped

- Example
  - t2.small in US-EAST-1 (VIRGINIA), cost $0.023 per Hour
  - If used for:
    - 6 seconds, it costs $0.023/60 = $0.000383 (minimum of 60 seconds)
    - 60 seconds, it costs $0.023/60 = $0.000383 (minimum of 60 seconds)
    - 30 minutes, it costs $0.023/2 = $0.0115
    - 1 month, it costs $0.023 * 24 * 30 = $16.56 (assuming a month is 30 days)
    - X seconds (X > 60), it costs $0.023 * X / 3600
  - The best way to know the pricing is to consult the pricing page: https://aws.amazon.com/ec2/pricing/on-demand/

AMIs

What's AMI?

- As we saw, AWS comes with base images such as:
  - Ubuntu
  - Fedora
  - RedHat
  - Windows
  - Etc...
- These images can be customized at runtime using EC2 User data
- But what if we could create our own image, ready to go?
- That's an AMI – an image to use to create our instances
- AMIs can be built for Linux or Windows machines

Why you use a custom AMI?

- Using a custom built AMI can provide the following advantages:
  - Pre-installed packages needed
  - Faster boot time (no need for long ec2 user data at boot time
  - Machine comes configured with monitoring / enterprise software
  - Security concerns – control over the machines in the network
  - Control of maintenance and updates of AMIs over time
  - Active Directory Integration out of the box

  - Installing your app ahead of time (for faster deploys when auto-scaling)
  - Using someone else's AMI that is optimized for running an app, DB, etc...
- **AMI are built for a specific AWS region (!)**

EC2 Instances Overview

- Instances have 5 distinct characteristics advertised on the website:
  - The RAM(type,amount,generation)
  - The CPU(type,make,frequency,generation,numberofcores)
  - The I/O (disk performance, EBS optimisations)
  - The Network (network bandwidth, network latency
  - The Graphical Processing Unit (GPU)
- It may be daunting to choose the right instance type (there are over 50 of them) - https://aws.amazon.com/ec2/instance-types/
- https://ec2instances.info/ can help with summarizing the types of instances
- R/C/P/G/H/X/I/F/Z/CR are specialised in RAM, CPU, I/O, Network, GPU
- M instance types are balanced
- T2/T3 instance types are "burstable" Burstable Instances (T2)
- AWS has the concept of burstable instances (T2 machines)
- Burst means that overall, the instance has OK CPU performance.
- When the machine needs to process something unexpected (a spike in load for example), it can burst, and CPU can be VERY good.
- If the machine bursts, it utilizes "burst credits"
- If all the credits are gone, the CPU becomes BAD
- If the machine stops bursting, credits are accumulated over time
- Burstable instances can be amazing to handle unexpected traffic and getting the insurance that it will be handled correctly
- If your instance consistently runs low on credit, you need to move to a different kind of non-burstable instance (all the ones described before).

T2 Unlimited

- Nov 2017: It is possible to have an "unlimited burst credit balance
- You pay extra money if you go over your credit balance, but you don't lose in performance
- Overall, it is a new offering, so be careful, costs could go high if you're not monitoring the health of your instances

Security Groups

The fundamental of network security in AWS

- Can be attached to multiple instances
- Locked down to a region / VPC combination
- Does live “outside” the EC2 - if traffic is blocked, the EC2 instance won’t see it
- It’s good to maintain one separate security group for SSH access
- If your application is not accessible (time out), then it’s usually a security group issue
- If your application gives a “connection refused” error, then it’s an application error or its not launched
- All inbound traffic is blocked by default
- All outbound traffic authorized by default

Security groups act as a firewall on EC2 Instances

They regulate:

- Access to ports
- Authorized IP ranges - IPv4 and IPv6
- Control of inbound network
- Control of outbound network

ELB: Elastic Load Balancers

Load balancers are servers that forward internet traffic to multiple servers (EC2 Instances) downstream

Why use a load balancer?

- Spread load across multiple downstream instances
- Expose a single point of access (DNS) to your application
- Seamlessly handle failures of downstream instances
- Do regular health checks to your instances
- Provide SSL termination (HTTPS) for your websites
- Enforce stickiness with cookies
- High availability across zones
- Separate public traffic from private traffic

AN ELB (EC2 Load Balancer) is a managed load balancer

- AWS guarantees that it will be working
- AWS takes care of upgrades, maintenance, high availability
- AWS provides only a few configuration knobs

It costs less to setup your own load balancer but it will be a lot more effort on your end. It is integrated with many AWS offerings / services

Types of load balancers on AWS

- Classic Load Balancer (v1 - older generation - 2009)
- Application Load Balancer (v2 - new generation - 2016)
- Network Load Balancer (v2 - new generation - 2017)
- You can setup internal or external ELBs

Health Checks

- Health checks are crucial for load balancers
- They enable the load balancer to know if instances it forwards traffic to are available to reply to requests
- The health check is done on a port and a route (/health is common)
- If the response is not 200 (OK), then the instance is unhealthy

Application Load Balancer (v2)

- Application load balancers (Layer 7) allow to do:
  - Load balancing to multiple HTTP applications across machines (target groups)
  - Load balancing to multiple applications on the same machine (ex: containers)
  - Load balancing based on route in URL
  - Load balancing based on hostname in URL
- Basically, they're awesome for micro services & container-based application (example: Docker & Amazon ECS)
- Has a port mapping feature to redirect to a dynamic port
- In comparison, we would need to create one Classic Load Balancer per application before.That was very expensive and inefficient!
- Good to Know
  - Stickiness can be enabled at the target group level
    - Same request goes to the same instance
    - Stickiness is directly generated by the ALB (NOT the application)

    - ALB supports HTTP/HTTPS & Web sockets protocols
    - The application servers don't see the IP of the client directly
        - The true IP of the client is inserted in the header X-Forwarded-For
        - We can also get Port (X-Forwarded-Port) and protocol (X-Forwarded-Proto)

Network Load Balancer (v2)

- Layer 4 allow you to do:
    - Forward TCP traffic to your instances
    - Handle millions of requests per second
    - Support for static IP or elastic IP
    - Less latency ~100ms (vs 400 ms for ALB)
- Network Load Balancers are mostly used for extreme performance and should not be the default load balancer you choose
- Overall, the creation process is the same as the Application Load Balancer

Load Balancers Good to Know

- Any Load Balancer (CLB, ALB, NLB) has a static host name. They do not resolve and use underlying IP
- LBs can scale but not instantaneously - contact AWS for a "warm up"
- NLB directly see the client IP
- 4xx errors are client induced errors
- 5xx errors are application induced errors
    - Load balancer Errors 503 means at capacity or no registered target
- If the LB can't connect to your application, check your security

ASG: Auto Scaling Group

In real-life, the load on your websites and applications can change. You can create and get rid of servers very quickly

The goal of an Auto Scaling Group (ASG) is to:

- Scale out (add EC2 Instances) to match an increased load
- Scale in (remove EC2 Instances) to match a decreased load
- Ensure we have a minimum and a maximum number of machines running
- Automatically register new instances to a load balancer

ASGs have the following attributes

- A launch configuration
  - AMI + Instance Type
  - EC2 User Data
  - EBS Volumes
  - Security Groups
  - SSH Key Pair
- Min Size / Max Size / Initial Capacity
- Network + Subnets Information
- Load Balancer Information
- Scaling Policies

Auto Scaling Alarms

- It is possible to scale an ASG based on CloudWatch alarms
- An alarm monitors a metric (such as Average CPU)
- Metrics are computed for the overall ASG instances
- Based on the alarm:
  - We can create a scale-out policies (increase the number of instances)
  - We can create a scale-in policies (decrease the number of instances)

New Auto Scaling Rules

- It is now possible to define "better" auto scaling rules that are directly managed by EC2
  - Target Average CPU Usage
  - Number of requests on the ELB per instance
  - Average Network In
  - Average Network Out
- These rules are easier to set up and can make more sense

Auto Scaling Custom Metric

- We can auto scale based on a custom metric (ex: number of connected users)
-
  i. Send custom metrics from an application on EC2 to CloudWatch (PutMetric API)
-

ii. Create a CloudWatch alarm to react to low / high values

-

iii. Use the CloudWatch Alarm as the scaling policy for ASG

ASG Summary

- Scaling policies can be on CPU, Network… and can even be on custom metrics or based on a schedule (if you know your visitors patterns)
- ASGs use Launch configurations and you update an ASG by providing a new launch configuration
- IAM roles attached to an ASG will get assigned to EC2 instances
- ASG are free. You pay for the underlying resources being launched
- Having instances under an ASG means that if they get terminated for whatever reason, the ASG will restart them. Extra safety
- ASG can terminate instances marked as unhealthy by an LB (and hence replace them)

EBS Volume

- An EC2 machine loses its root volume (main drive) when it is manually terminated.
- Unexpected terminations might happen from time to time (AWS would email you)
- Sometimes, you need a way to store your instance data somewhere
- An EBS (Elastic Block Store) Volume is a network drive you can attach to your instances while they run
- It allows your instances to persist data

EBS Volume

- It's a network drive (Not a physical drive)
  - It uses the network to communicate the instance, which means there might be a bit of latency
  - It can be detached from an EC2 instance and attached to another one quickly
- It's locked to an Availability Zone (AZ)
  - An EBS Volume in us-east-1a cannot be attached to us-east-1b
  - To move a volume across, you first need to snapshot it
- Have a provisioned capacity (size in GBs and IOPs)

  - You get billed for all the provisioned capacity
  - You can increase the capacity of the drive over time

EBS Volume Types

- EBS Volumes come in 4 types
- GP2 (SSD): General purpose SSD volume that balances price and performance for a wide variety of workloads
- IO1 (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
- When in doubt always consult the AWS documentation

EBS Volume Resizing

- Feb 2017: You can resize your EBS Volumes
- After resizing an EBS volume, you need to repartition your drive

EBS Snapshots

- EBS Volumes can be backed up using "snapshots"
- Snapshots only take the actual space of the blocks on the volume
- If you snapshot a 100GB drive that only has 5 gb of data, then your EBS snapshot will only be 5 gb
- Snapshots are used for:
  - Backups: ensuring you can save your data in case of catastrophe
  - Volume migration
    - Resizing a volume down
    - Changing the volume type
    - Encrypt a volume

EBS Encryption

- When you create an encrypted EBS volume, you get the following:
  - 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 snapshots are encrypted
- Encryption and decryption are handled transparently (you have nothing to do)
- Encryption has a minimal impact on latency
- EBS Encryption leverages keys from KMS (AES-256)
- Copying an unencrypted snapshot allows encryption

EBS vs. Instance Store

- Some instance do not come with Root EBS volumes
- Instead, they come with "instance Store"
- Instance store is physically attached to the machine
- Pros:
    - Better I/O performance
- Cons:
    - On termination, the instance store is lost
    - You can't resize the instance store
    - Backups must be operated by the user
- Overall, EBS-backed instances should fit most applications workloads

EB Deployment Modes

- Single Instance mode: Great for development environment
- High Availability with Load Balancer mode: Great for production environments

What if you want to update each deployment

- **All at once (deploy on the go)**
    - Fastest, but instances aren't available to serve traffic for awhile (longer downtime)
    - No additional cost
- **Rolling update**
    - update a few (bucket) instances at a time, and then move onto the next bucket when the current ones become healthy
    - You can set the bucket size
    - Application will run below capacity during update
    - At some point, the application will run both versions simultaneously
    - Can be a very long deployment time depending on number of instances running
    - No additional cost

- **Rolling update with additonal batches**
    - Similar to rolling updates but you spin up new instances to move the batch (so the old application is still available)
    - Application is running at capacity
    - You can set the bucket size
    - Additional batches are removed at the end of the deployment
    - Small additional cost (due to additional running instances)
    - Great for production environments
- **Immutable**
    - Spins up new instances in a new ASG, deploys versions to these instances and then swaps all the instances when everything is healthy
    - Zero downtime
    - New code is deployed on new instances in a temporary ASG
    - High cost, double capacity
    - Longest deployment
    - Quick rollback in case of failures (new ASG will be terminated)
    - Best for production environements

Blue / Green Deployment

- This is not a direct feature of Elastic Beanstalk
- Zero downtime and release facility
- Create a new staging environment and deploy your newest version there
- The new environment (green) can be validated independently and roll back if there's issues
- Route 53 can be setup using weighted policies to redirect a little bit of traffic to the staging environment
- Using the elastic beanstalk console, you can "swap URLs" when with the testing environment
- This is a manual feature, it's not directly embedded in EB

Elastic Beanstalk Extensions

- A zip file containing our code must be deployed to Elastic Beanstalk
- All the parameters set in the UI can be configured with code using files
- Requirements:
    - in the .ebextensions/ directory in the root of source code
    - YAML / JSON format
    - .config extensions (example: logging.config)

  - Able to modify some default settings using: option_settings
  - Ability to add resources such as RDS, ElastiCache, DynamoDB, etc...
- Resources managed by .ebextensions get deleted if the environment goes away
- The .ebextensions folder goes to the root of your project

Elastic Beanstalk CLI

- We can install an additional CLI called the "EB cli" which makes working with Beanstalk from the CLI easier
- Basic commands are:
  - eb create
  - eb status
  - eb health
  - eb events
  - eb logs
  - eb open
  - eb deploy
  - eb config
  - eb terminate
- It's helpful for your automated deployment pipelines!

Deployment Mechanism

- Describe dependancies
  - (requirements.txt for python, package.json for node.js)
- Package code as zip
- Zip file is uploaded to each EC2 machine
- Each EC2 machine resolves dependencies (SLOW)
- Optimization in case of long deployments:
  - Package dependencies with source code to improve deployment performance and speed

EBS Summary

- EBS can be attached to only one instance at a time
- EBS are locked at the AZ level
- Migrating an EBS volume across AZ means first backing it up (snapshot), then recreating it in the other AZ

- EBS backups use IO and you shouldn’t run them while your application is handling a lot of traffic
- Root EBS Volumes of instances get terminated by default if the EC2 instance gets terminated. (You can disable that)
- In some cases, it's better to externalize your RDS database so that it won't get deleted when you delete your elastic beanstalk enviornment
- Elastic Beanstalk relies on CloudFormation

RDS: Relational Database Service

A managed DB service for DB use SQL a query

It allows you to create databases in the cloud that are

- Postgres
- Oracle
- MySQL
- MariaDB
- Microsoft SQL Server
- Aurora (AWS proprietary database)

Advantages of RDS over deploying a database in EC2

- Managed service
- OS patching level
- Continuous backups and restore to specific timestamps (Point in Time Restore)
- Monitoring dashboards
- Read replicas for improved read performance
- Multi AZ setup for DR (Disaster Recovery)
- Maintenance windows for upgrades
- Scaling capability (vertical and horizontal)
- But you can’t SSH into your instances (amazon manages them for you)

RDS Read replicas for read scalability

- Up to 5 read replicas
- Within AZ, Cross AZ or Cross region
- Replication is Async, so reads are eventually consistent

- Replicas can be promoted to their own DB
- Applications must update the connection string to leverage read replicas

RDS Multi AZ (Disaster Recovery)

- SYNC replication
- One DNS name - automatic app failover to standby
- Increase availability
- Failover in case of loss of AZ, loss of network, instance or storage failure
- No manual intervention in apps
- Not used for scaling (only disaster recovery)

RDS Backups

- Backups are automatically enabled in RDS
- Automated backups:
  - Daily full snapshot of the database
  - Capture transaction logs in real time
  - Ability to restore to any point in time
  - 7 days retention (can be increased to 35 days)
- DB Snapshots:
  - Manually triggered by the user
  - Retention of backup for as long as you want

RDS Encryption

- Encryption at rest capability with AWS KMS - AES-256 encryption
- SSL certificates to encrypt data to RDS in flight
- To enforce SSL:
  - PostgreSQL: rds.force_ssl=1 in the AWS RDS Console (parameter groups)
- TO connect using SSL:
  - Provide the SSL Trust certificate (can be downloaded from AWS)
  - Provide SSL options when connection to the database

RDS Security

- RDS databases are usually deployed within a private subnet, not in a public one
- RDS Security works by leveraging security groups (the same concept as for EC2 instances) - it controls who can communicate with RDS

- IAM policies help control who can manage RDS
- Traditional username and password can be used to login to the database
- IAM users can now be used too (for MySQL / Aurora - New)

RDS vs. Aurora

- Aurora is a proprietary technology from AWS (not open sourced)
- Postgres and MySQL are both supported as Aurora DB (that means you r drivers will work as if Aurora was a Postgres or MySQL database)
- Aurora is "AWS cloud optimized" and claims 5x performance improvements over MySQL on RDS, over 3x the performance of Postgres on RDS
- Aurora storage automatically grows in increments of 10GB, up to 64 TB
- Aurora can have 15 replicas while MySQL has 5, and the replication process is faster (sub 10 ms replica lag)
- Failover in Aurora is instantaneous. It's HA native.
- Aurora costs more than RDS (20% more) - but is more efficient

Route 53

Route 53 is a managed DNS (Domain Name System)

DNS is a collection of rules and records which helps clients understand how to reach a server through URLs.

In AWS, the most common records are (will be on exam):

- A: URL to IPv4
- AAAA: URL to IPv6
- CNAME: URL to URL
- ALIAS: URL to AWS resource

Route 53 can use:

- Public domain names you own
- Private domain names that can be resolved by your instances in your VPCs

Route53 has advanced features such as:

- Load balancing (through DNS - also called client load balancing)
- Health checks (although limited…)
- Routing policy: simple, failover, geolocation, geoproximity, latency, weighted

Prefer Alias over CNAME for AWS resources (for performance reasons)

ElastiCache

Overview: The same way RDS is to get managed Relational Databases, ElastiCache is to get managed Redis or Memcached. Caches are in-memory databases with really high performance, low latency. They help reduce loads off of databases for read intensive workloads. They help make your application stateless.

- Write scaling using shading.
- Read scaling using Read Replicas
- Multi AZ with Failover Capability
- AWS takes care of OS maintenance / patching, optimizations, setup, configuration, monitoring, failure recovery and backups

Solution Architecture - DB Cache

- Applications queries ElastiCache, if not available, get from RDS and store in ElastiCache
- Helps relieve load in RDS
- Cache must have an invalidation strategy to make sure only the most current data is used in there User Session Store
- User logs into any of the applications
- The application writes the session data into ElastiCache
- The user hits another instance of our application
- The instance retrieves the data and the user is already logged in

Redis Overview

- Redis is an in-memory key-value store
- Super low latency (sub ms)
- Cache survive reboots by default (it’s called persistence)
- Great to host
  - User sessions
  - Leaderboards (for gaming)
  - Distributed states
  - Relieve pressure on databases (such as RDS)
  - Pub / Sub capability for messaging

- Multi AZ with Automatic failover for Disaster Recovery if you don't want to lose your cache data
- Support for Read Replicas

Memcached Overview

- Memcached is an in-memory object store
- Cache doesn't survive reboots
- Use cases:
- Quick retrieval of objects from memory
- Cache often accessed objects
- Overall, Redis has largely grown in popularity and has better feature sets than memcached
- Most likely, you'd probably only want to use Redis for caching needs

VPC: Virtual Private Cloud

Within a region, you're able to create VPCs. Each VPC contain subnets (networks). Each subnet must be mapped to an AZ. It's common to have a public ip and private ip subnet. It's common to have many subnets per AZ.

Public Subnets usually contain:

- Load Balancers
- Static Websites
- Files
- Public Authentication Layers

Private Subnets usually contain:

- Web application servers
- Databases

Public and Private subnets can communicate if they're in the same VPC

AWS VPC Summary

- VPC & Regions aren't much asked at the developer associate exam
- All new accounts come with a default VPC
- It's possible to use a VPN to connect to a VPC

- VPC flow logs allow you to monitor the traffic within, in and out of your VPC (useful for security, performance, audit)
- VPC are per Account per Region
- Subnets are per VPC per AZ
- Some AWS resources can be deployed in VPC while others can't
- You can peer VPC (within or across accounts) to make it look like they're part of the same network

S3 Buckets

- Amazon S3 allows people to store objects (files) fun "buckets" (directories)
- Buckets must have a globally unique name
    - Naming convention:
        - No uppercase
        - No underscore
        - 3-63 characters long
        - Not an IP
        - Must start with lowercase letter or number
- Objects
    - Objects (files) have a Key. The key is the FULL path:
        - <my_bucket>/my_file.txt
        - <my_bucket>/my_folder/another_folder/my_file.txt
    - There's no concept of "directories" within buckets (although the UI will trick you to think otherwise)
    - Just keys with very long names that contain slashes ("/")
    - Object Values are the content of the body:
        - Max Size is 5TB
        - If uploading more than 5GB, must use "multi-part upload"
    - Metadata (list of text key / value pairs - system or user metadata)
    - Tags (Unicode key / value pair - up to 10) - useful for security / lifecycle
    - Version ID (if versioning
    -

AWS S3 - Versioning

- It is enabled at the bucket level
- Same key overwrite will increment the "version": 1, 2, 3
- It is best practice to version your buckets

    - Protect against unintended deletes (ability to restore a version)
    - Easy roll back to previous versions
- Any file that is not version prior to enabling versioning will have the version "null"

S3 Encryption for Objects

- There are 4 methods of encrypt objects in S3
    - SSE-S3: encrypts S3 objects
        - Encryption using keys handled & managed by AWS S3
        - Object is encrypted server side
        - AES-256 encryption type
        - Must set header: "x-amz-server-side-encryption":"AES256"
    - SSE-KMS: encryption using keys handled & managed by KMS
        - KMS Advantages: user control + audit trail
        - Object is encrypted server side
        - Maintain control of the rotation policy for the encryption keys
        - Must set header: "x-amz-server-side-encryption":"aws:kms"
    - SSE-C: server-side encryption using data keys fully managed by the customer outside of AWS
        - Amazon S3 does not store the encryption key you provide
        - HTTPS must be used
        - Encryption key must provided in HTTP headers, for every HTTP request made
    - Client Side Encryption
        - Client library such as the amazon S3 Encryption Client
        - Clients must encrypt data themselves before sending to S3
        - Clients must decrypt data themselves when retrieving from S3
        - Customer fully manages the keys and encryption cycle

Encryption in transit (SSL)

- AWS S3 exposes:
    - HTTP endpoint: non encrypted
    - HTTPS endpoint: encryption in flight
- You're free to use the endpoint your ant, but HTTPS is recommended
- HTTPS is mandatory for SSE-C
- Encryption in flight is also called SSL / TLS

S3 Security

- User based
  - IAM policies - which API calls should be allowed for a specific user from IAM console
- Resource based
  - Bucket policies - bucket wide rules from the S3 console - allows cross account
  - Object Access Control List (ACL) - finer grain
  - Bucket Access Control List (ACL) - less common
- Networking
  - Support VPC endpoints (for instances in VPC without www internet)
- Logging and Audit:
  - S3 access logs can be stored in other S3 buckets
  - API calls can be logged in AWS CloudTrail
- User Security:
- MFA (multi factor authentication) can be required in versioned buckets to delete objects
- Signed URLs: URLS that are valid only for a limited time (ex: premium video services for logged in users)

S3 Bucket Policies

- JSON based policies
  - Resources: buckets and objects
  - Actions: Set of API to Allow or Deny
  - Effect: Allow / Deny
  - Principal: The account or user to apply the policy to
- Use S3 bucket for policy to:
  - Grant public access to the bucket
  - Force objects to be encrypted at upload
  - Grant access to another account (Cross Account)

S3 Websites

- S3 can host static website sand have them accessible on the world wide web
- The website URL will be:
  - .s3-website..amzonaws.com
  - OR
  - .s3-website..amazonaws.com

- If you get a 403 (forbidden) error, make sure the bucket policy allows public reads!
-

S3 Cors

- If you request data from another S3 bucket, you need to enable CORS
- Cross Origin Resource Sharing allows you to limit the number of websites that can request your files in S3 (and limit your costs)
- This is a popular exam question

AWS S3 - Consistency Model

- Read after write consistency for PUTS of new objects
    - As soon as an object is written, we can retrieve itex: (PUT 200 -> GET 200)
    - This is true, except if we did a GET before to see if the object existedex: (GET 404 -> PUT 200 -> GET 404) - eventually consistent
- Eventual Consistency for DELETES and PUTS of existing objects
    - If we read an object after updating, we might get the older versionex: (PUT 200 -> PUT 200 -> GET 200 (might be older version))
    - If we delete an object, we might still be able to retrieve it for a short timeex: (DELETE 200 -> GET 200)

AWS S3 - Other

- S3 can send notifications on changes to
    - AWS SQS: queue service
    - AWS SNS: notification service
    - AWS Lambda: serverless service
- S3 has a cross region replication feature (managed)

AWS S3 Performance

- Faster upload of large objects (>5GB), use multipart upload
    - Parallelizes PUTs for greater throughput
    - Maximize your network bandwidth
    - Decrease time to retry in case a part fails
- Use CloudFront to ache S3 objects around the world (improves reads)
- S3 Transfer Acceleration (uses edge locations) - just need to change the endpoint you write to, not the code

- If using SSE-KMS encryption, you may be limited to your AWS limits for KMS usage (~100s - 1000s downloads / uploads per second)

CLI: Command Line Interface

Add user credentials locally using this command:

- $ `aws configure`

If you are using multiple AWS accounts, you can add custom profiles with seperate credentials using this command:

- $ `aws configure --profile {my-other-aws-account}`
- if you you'd like to execute commands on a specific profile:
  - example: `aws s3 ls --profile {my-other-aws-account}`
- if you don't specify the aws profile, the commands will be executed to your **default** profile

AWS CLI on EC2

- IAM roles can be attached to EC2 instances
- IAM roles can come with a policy authorizing exactly what the EC2 instance should be able to do. This is the best practice.
- EC2 Instances can then use these profiles automatically without any additional configurations

CLI STS Decode Errors

- When you run API calls and they fail, you can get a long, encoded error message code
- This error can be decoded using STS
- run the command: `aws sts decode-authorization-message --encoded-message {encoded_message_code}`
- your IAM user must have the correct permissions to use this command by adding the STS service to your policy

SDK: Software Development Kit

If you want to perform actions on AWS directly from your application's code without using a CLI, you can use an SDK

Official SDKs:

- Java
- .NET
- Node.js
- PHP
- Python
- Ruby
- C++

SDK Takeaways

- AWS SDK are required when coding against AWS Services such as DynamoDB
- Fact: AWS CLI uses the Python SDK (boto3)
- The exam expects you to know when you should use an SDK
- If you don't specify or configure a default region, then us-east-1 will be chosen by default

SDK Credentials Security

- It's recommend to use the default credential provider chain
- The default credential provider chain works seamlessly with:
    - AWS credentials at ~/.aws/credentials (only on our computers or on premise)
    - Instance Profile Credentials using IAM Roles (for EC2 machines, etc…)
- Environment variables (AWS_ACCESS_KEY_ID, AWS_SECRET_ACCESS_KEY)
- Overall, NEVER EVER STORE AWS CREDENTIALS IN YOUR CODE.
- Use IAM Roles if working from within AWS Services to inherit credentials

Exponential Backoff

- Any API that fails because of too many calls needs to be retried with Exponential Backoff
- These apply to rate limited API
- Retry mechanism is included in SDK API calls

Elastic Beanstalk

***Elastic Beanstalk* is a developer centric view of deploying application on AWS.**

- A managed service
  - Instance configuration
  - OS is handled by Beanstalk
  - Deployment strategy is configurableut performed by Beanstalk
  - Application code configurable
- It will leverage all the AWS components that we have gone over thus far:
  - EC2
  - ASG
  - ELB
  - RDS
  - Etc..
- Elastic Beanstalk is free but you pay for the underlying instances
- Three architecture models:
  - Single instance deployment: good for developers
  - LB + ASG: great for production or staging web applications
  - ASG only: great for non-web apps in production
- Elastic Beanstalk has three components:
  - Application
  - Application Version (Each deployment gets assigned a version)
  - Environment name (dev, staging, prod): free naming
- You deploy application versions to environments and can promote application versions to the next environment
- Rollback feature to previous application versions
- Full control over the lifecycle of environments
- Support for many platforms:
  - Go
  - Java
  - Python
  - Node.js
  - Ruby
  - Single Container Docker
  - Multi Container Docker
  - Preconfigure Docker

  - Write your own custom platforms (if not supported)

AWS-Services

EC2 - Elastic Compute Cloud virtual machines

Lightsail - Provisiong service - Very hands off

Elastic Container Service - running containers such as docker at scale

Lambda - Serverless functions

Elastic Beanstalk - Easier route for developers to get up and running with their cloud

ElastiCache - Cache common searches in front of DB servers

S3 - Key pair object storage kept in buckets

EFS - NFS can be mounted on multiple instances

Glacier - Archival storage

Snowball - Hardware appliance to transfer data between on-prem and AWS

Storage gateway - Virtual appliances that live on-prem and replicate to AWS

RDS - MySQL, MSSQL, Aurora, PostGreSQL

DynamoDB - NoSQL

RedShitft - Data warehousing

AWS Migration Hub - Dashboard that lets you track your application migration

Application Discovery Service - tracks your applications dependencies

Database Migration Service - Migrate DBs to AWS

Server Migration Service

VPC Virtual Private Cloud

Cloudfront - Content Delivery Network (CDN) caches content to make it available quicker to the end user.

Route53 - Amazon's DNS service

API Gateway - Creating API's for your own services

Direct Connect - Network peering between yourself and AWS

Codestar - Project managing code. Collorbation tool

Codecommit - Source control service

Codebuild - Complies and test your code

Codedeploy - Automates your application deployment

Codepiple - CDS

X-Ray - Used to debug your serverless application

Cloud9 - Online IDE

CloudWatch - Monitoring

CloudFormation - Infrastructure as Code

CloudTrail - API logging

Config - Monitor AWS account config

OpsWorks - Config management using Chef or Puppet

Service Catalog - Managing IT services approved for use

Systems Manager - Patch maintenance Trusted Advisor - Gives advice on security, cost

Elastic Transcoder - Video transcoding. Sizing videos for various devices

Lex - Powers Alexa

Polly - Text to speech

Rekognition - Analyse images and video

Amazon translate - Language translate

Amazon Transcribe - Automatic speech regonition

Athena - run SQL queries against S3 buckets

Elastic Map Reduce - managed software framework used to process large data sets in a distributed computing environment. Used for data analysis, web indexing, data warehousing, machine learning, financial analysis, scientific simulation etc. EMR supports workloads based on Hadoop, Apache Spark, Presto and Apache HBase.

CloudSearch

ElasticSearch Service

Kinesis - Ingesting large amounts of data

Kinesis Video Streams - Ingesting lots of video streams

Data Pipeline - Moving data between AWS services

IAM - Identity and Management access

Cognito - Mobile Device authentication using federated accounts Facebook etc

Guard Duty -

Inspector - Anaylse instance security using agent

Macie - Scans S3 buskets for personal iD numbers

Certificate manager - Free SSL certs

CloudHSM - Hardware security module which store keys

Directory services - Connect AWS to onside AD

WAF Web application firewall - L7 firewall

Shield - DDOS mitigation

Artifcat - AWS compliance reports

SNS - Simple Notification Service

SQS - Simple Queue Service - is a web service that gives you access to message queues that store messages waiting to be processed. With Amazon SQS, you can quickly build message queuing applications that can run on any computer. Amazon SQS can help you build a distributed application with decoupled components, working closely with the Amazon Elastic Compute Cloud (Amazon EC2) and other AWS infrastructure web services.

SWF - Simple Workflow Service

Simple Email Service - Sending emails to customers

WorkMail - Office365

Workspaces - VDI

IAM

IAM FAQ

Users Users have the choice of being given access to the management console and/or programmatic access. Access via the management console enables a password for the account. Enabling programmatic access enables an access key ID and secret access key. This can be used to access t

Groups

Groups allow you to apply policies to multiple users. Recommended to apply policies to groups even if it is for one user.

- Users can be members of multiple groups
- Groups cannot be nested

Policies Policies are JSON documents that contain permissions to AWS services. ie

Roles

Secret

Security Token Service (STS)

- Grants users limited and temporary access to AWS resources.
- 3 sources:
    1. Federation (often Active Directory)
        - Uses SAML
        - SSO allows users to log in to AWS Console without assigning IAM credentials
    2. Federation with mobile app
        - Use Facebook/Amazon/Google or other openID provider
    3. Cross account access
        - Lets users from one AWS account access resources in another

Active Directory Federation

EC2

EC2 FAQ

Access instance meta data at http://169.254.169.254/latest/meta-data/

- Scipts can be run from the user data section when creating an instance

Load Balancers

- There are 3 types of load balancer; Application, Network and Classic. Application is used to route HTTP/HTTPS (L7) traffic. Network and Classic are used to route TCP (L4) traffic.

  i. Application - TLS termination
  ii. Network - Extermeme performance and static IP
  iii. Classic (also refered to as Elastic Load Balancer ELB) -

- Sticky sessions are a mechanism to route requests to the same target in a target group. This is useful for servers that maintain state information in order to provide a continuous experience to clients.

S3

S3 FAQ

Storage classes

- Buckets can contain objects of different storage classes

1. S3 Standard
2. S3 Standard-Infrequent Access - for data that is less frequently accessed but requires rapid access when needed. Availability drops to 99.9% and there is a data retrieval charge of $0.01 / GB.
3. S3 One Zone-Infrequent Access - Offers similar performance as other S3 classes but stores data redundantly within an Availability Zone not across Availability Zones.
4. Glacier - used for archiving data.

- No limit on number of objects in a bucket

- Largest object size is 5TB

- Smallest object size is 0 bytes
- Largest upload in a single PUT is 5GB. (Objects larger than 100MB should be uploaded with multipart uploader)
- A bucket cannot contain a bucket
- Need to delete large amounts of
- S3 provides read-after-write consistency for PUTS of new objects.
- S3 offers eventual consistency for overwrite PUTS and DELETES.
- If you expect more than 300 PUT/LIST/DELETE requests per second or more than 800 GET request per second raise a support request with AWS to prepare for the workload.
- Event notifications can sent in response to actions such as PUTs, POSTs, COPYs or DELETEs, Messages can be sent through SNS, SQS or Lambda.
- CORS (Cross-Origin resource sharing) enables a way for client web applications loaded in one domain to interact with resources in a different domain.

Encryption

- Server-Side

1. KMS-Managed Encryption keys
2. Amazon S3-Managed Encryption keys
3. Customer-Provided Encryption keys

- Client-Side

1. AWS KMS-managed customer master key
2. Client-side master key

DynamoDB

DynamoDB FAQ

NoSQL database Stored on SSD

Spread across 3 geographically distinct data centres

Consistency models

1. Eventual consistent reads (default). Offers best read performance. Consistency across all copies of data is usually reached within a second.
2. Strongly consistent reads. Returns a result that reflects all writes that received a successful response prior to the read.

Indexes

Two types of primary keys available; Single Attribute - partition key (Customer no, driver license etc) Composite - partition key & sort key (Customer no & date range)

Local Secondary Index same partition key but different sort key Can only be created when creating a table

Global Secondary Index Different partition key and different sort key Can be created at table creation or added later

Streams

Four options for streams only 1 can be selected

1. Keys Only - Only the key attributes of the modified item
2. New image the entire item, as it appears after it was modified
3. Old image - the entire item, as it appeared before it was modified
4. New and old images - both the new and the old images of the item

- Max 24 hour storage
- Can have Lambda triggered from streams

If a new item is added to the table, the stream captures an image of the entire item, including all attributes if item is updated, stream captures the before and after image of any attributes that were modified if item is deleted the stream captures an image of the item before deletion

Query vs Scan

- Query - a query find items in a table using only the primary key.
- Scan - a scan operation examines every item in the table. By default a scan returns all of the data attributes for every item. You can use the ProjectionExpression parameter so that Scan only returns some of the attributes.

Query is more efficient than scan

Batch get item for more efficient queries of large items

Provisioned Throughput

DynamoDB is priced on the storage size and its 'Provisioned Throughput'. Provisioned throughput is made up of read capacity units and write capacity units. All reads rounded up to 4KB. Eventually consistent reads (default) consists of 2 reads per second. Strongly consistent reads consist of 1 read per second. All writes are 1KB. All writes consist of 1 write per second.

Formula is (size of read rounded to 4KB chunk / 4KB) * no of items = read throughput Divide by 2 if eventually consistent

If you exceed your provisioned throughput you will get a HTTP status code 400, ProvisionedThroughputExceededException.

Simple Queue Service SQS

SQS FAQ

SQS tutorial

SQS is a pull based messaging service.

Allows the 'decoupling' of components of an application.

FIFO queues are not supported in all regions. Currently only: US East (Ohio), US East (N. Virginia), US West (Oregon), and EU (Ireland) regions.

The maximum amount of time that a message can live in a SQS queue is 14 days. The retention period can be configred to be anywhere betweeen 1 minute and 14 days. The default is 4 days. Once the message retention limit is reached, your messages are automatically deleted.

SQS messages must be between 1 and 256 KB in size. Billed in 64KB chunks.

SQS supports two types of pull based polling:

Short polling - SQS returns a response immediately, even if there is no message in the queue **Long polling** - doesn’t return a response until a message arrives in the message queue, or the long poll times out. Can be cheaper then short polling as it can reduce the number of empty receives. In almost all cases, long polling is preferable to short polling. One case you might want to use short polling is if you application uses a single thread to poll multiple queues.

When a consumer receives a message from the SQS queue, it stays in the SQS queue. The message must be deleted by the consumer once the message has been fully processed. To prevent other conumers from receiving the message, SQS sets a Visibility Timeout, which is the period of time where SQS prevents other consuming components from receiving and processing the message.

First 1 million requests are free, then $0.50 for every million after.

Simple Notification Service (SNS)

SNS FAQ

SNS tutorial After a message has been published to a topic it cant be deleted (recalled)

SNS is a messaging service that 'pushes' messages to clients.

Messages protocols:

- Application
- SMS text message
- Email
- Email-JSON
- AWS SQS
- HTTP
- HTTPS

SNS can be used with SQS to fan messages out to multiple queues.

SNS uses Topics to send messages. To receive messages published to a topic you have to subscribe. Once a message is published, SNS attempts to deliver to every endpoint that is subscribed.

Messages can be customised by protocol type.

Messages are stored reduntly across mulitple AZ's.

Simple Workflow Service (SWF)

SWF FAQ

- Workers are programs that interact with SWF to get tasks, process received tasks and return the results.

- Decider is a program that controls the coordination of tasks.

Tasks assigned only once and never duplicated.

Domains - workflow and activity types and the workflow execution itself are all scoped to a domain. Domains isolate a set of types, executions, and task lists from other within the same account. You can register a domain by using the console or SWF API. Using JSON.

SWF vs SQS

- SWF presents task oriented API whereas SQS offers message oriented API.
- SWF ensures that a task is assigned only once. With SQS you need to handle duplicated messages and may also need to ensure that a message is processed only once.
- SWF keeps track of all tasks and events in an application. With SQS you need to implement your own application-level tracking.

Elastic Beanstalk

Elastic Beanstalk FAQ

- Can have multiple versions of your applications (Dev/Test)
- Your applications can be split into tiers. Frontend/backend etc
- able to update application
- can update your configuration ie change instance type behind the app
- Updates can be 1 instance at a time or % of instances or immutable

Languages

- Apache Tomcat for Java
- Apache HTTP server for PHP
- Apache HTTP Server for Python
- Nginx or Apache HTTP for Node.js
- Passenger or Puma for Ruby
- Microsoft IIS 7.5, 8.0 and 8.5 for .NET
- Java SE
- Docker
- Go

CloudFormation

CloudFormation FAQ

A cloudFormation is made up of the following sections:

Resources (required) - specify the stack resources and their properties such as an EC2 instance or a S3 bucket. You can refer to resources in the Resources and Outputs sections of the template.

Metadata (optional) - objects that provide additional information about the template.

Parameters (optional) - specifies values that you can pass in to your template at runtime (when you create or update a stack). You can refer to parameters in the Resources and Outputs sections of the template.

Mappings (optional) - a mapping of keys and associated values that you can use to specify conditional parameter values, similar to a lookup table. You can match a key to a corresponding value by using the Fn::FindInMap intrinsic function in the Resources and Outputs section.

Conditions (optional) - defines conditions that control whether certain resources are created or whether certain resource properties are assigned a value during stack creation or update. For example, you could conditionally create a resource that depends on whether the stack is for a production or test environment.

Transform (optional) - for serverless applications (also referred to as Lambda-based applications), specifies the version of the AWS Serverless Application Model (AWS SAM) to use.

Outputs describes the values that are returned whenever you view your stack’s properties. For example, you can declare an output for an S3 bucket name and then call the aws cloudformation describe-stacks AWS CLI command to view the name. The only required section in a Cloudformation template is the Resources section

- Automatic rollback on error is enabled by default.
- Use function Fn:GetAtt to output data
- Stacks can wait for applications to be provisioned using the 'waitCondition'

AWS Shared Responsibility

IaaS - Customer manages OS and above including security and patches. AWS manages hypervisor and below including physical infrastructure.

SaaS – AWS manages everything except user credentials.

Route-53

Route53 FAQ

- Limit of 50 domain names - speak to AWS support to adjust.
- CNAME (Canonical Name) can be used to resolve one domain another
- A Record (Address Record) for resolving a domain name to an IP address
- Alias records are an AWS / Route 53 specific term, similar to CNAME with the key distinction that CNAMEs can't be used on the zone apex (root domain i.e. cnames could be used against sofa.furniture.com, but not against furniture.com - for this you'd need to use either an A Record or Alias record)
- In the exam always choose Alias over CNAME. Amazon dont charge to resolve Alias records and they can be used to map naked domain apex to an ELB.

Routing Policies

- Simple - default policy, used when there is only one resource.
- Weighted - send a specified amount of traffic to certain resources. ie for every 10 requests send 70% to us-east-1 and 30% to eu-west-1.
- Latency - used to send traffic to lowest latency region. This requires you to create an A record for each region you want the latency to be evaluated against.
- Failover - failover allows you to have an active/passive design. Using health checks to assess whether to send traffic to the primary or secondary resource. A health check can use Cloud watch alarms, other health checks or simply use a TCP connection to an IP or domain name.
- Geolocation - used to send traffic to a particular region based on source location. ie Customers in a Eurozone country always get routed to a server with prices in Euros.

VPC

VPC FAQ

- By default all traffic between subnets is allowed

- /16 is the largest CIDR block available
- Subnets have a 1 to 1 mapping to an Availability Zone
- 1 Internet Gateway per VPC
- You cannot change the ip range of a VPC
- Elastic IP addresses (EIPs) are public IP addresses that
- Elastic Network interface

CloudFront

- Content Delivery Network (CDN). Provides content quicker to customers by caching it in edge locations. ie customer 1 watches a video from s3. s3 bucket is in Ireland but user is in Sydney. The content flows Ireland -----> Sydney. CloudFront caches it locally near Sydney so the second time its accessed the content flows, CloudFront Sydney -> Sydney.
- Edge locations can be used for write as well as read.
- Objects are cached for ther life of their TTL. TTL can be 0 seconds to 365 days. Default is 24 hours.
- Origin can be S3, EC2, ELB, Route53 and non AWS server ie on-prem
- Restrict viewer access by signed URL or Signed Cookies
- Restrict content based on geo location (whitelist and blacklist)

Lambda

- Compute service allows you to run code without provisioning and managing servers. Under the hood are EC2 Instances managed by AWS.
- Lambda is stateless and event driven.
- If we increase memory, cpu usage will get increase. Max memory limit is 3008 MBs. Max execution timeouts is 300
- Temporary objects downloaded by lambda are stored in /tmp directory.

- Alias can be use to manage different versions for lambda. You can change version behind lambda.

- Use AWS Lambda Environment Variables to pass operational parameters to your function.

- Lambda Optimization Tips: Avoid using recursion, keep deployment size minimum, install only dependecies that is required, keep your function logic outside handler.
  (source: https://docs.aws.amazon.com/lambda/latest/dg/best-practices.html)