

**aws\_developer\_associate\_notes**

**Tina Bu**

## **AWS Developer Associate Notes**

- AWS Developer Associate Notes
  - Exam Details
  - 1 IAM
  - 2 EC2
    - 2.1 EC2 101
    - 2.2 Security Group
    - 2.3 EBS Volume
    - 2.4 Command Line CLI
    - 2.5 CloudWatch
    - 2.6 EFS - Elastic File System
    - 2.7 Spread placement groups
    - 2.8 Load Balancer
  - 3 Databases
    - 3.1 Relational Database
    - 3.2 RDS (OLTP)
    - 3.3 Non-relational databases
    - 3.4 Dynamo DB
    - 3.5 Data Warehousing - Redshift (OLAP)
    - 3.6 ElastiCache
    - 3.7 Aurora
  - 4 Route53
    - 4.1 DNS 101
    - 4.2 Route53
  - 5 S3 (Simple Storage Service)
  - 4 Serverless Computing
  - 5 DynamoDB
  - 6 KMS & Encryption
  - 7 Other AWS Services
  - 8 Developer Theory
  - 9 Monitoring

## **Exam Details**

- Exam Domains
  - 22% Deployment (CI/CD, Beanstalk, prepare deployment)

- package, serverless)
- 26% Security (make authenticated calls, encryption, application authentication and authorization)
- 30% Developing with AWS Services (code serverless, functional requirements -> application design, application design -> code, interact with AWS via APIs, SDKs, AWS CLI)
- 10% Refactoring (optimize application to best use AWS)
- 12% Monitoring and Troubleshooting (write code that can be monitored, root cause analysis on faults)
- Exam Format
  - Multiple choice or multiple response questions
  - 130 min in length, 65 questions
  - scenario based questions
  - Score 100 - 1000, passmark is 720
  - \$150
  - certification valid for 2 years
- serverless focused, less on EC2

## 1 IAM

- Always use roles, never use secret or access keys
- Manage users and their level of access to AWS
- capabilities
  - centralized control: universal, doesn't apply to region
  - shared access to your AWS account
  - granular permissions
  - Identity Federation
    - Active Directory: log in with Windows login password
    - Facebook / LinkedIn / ...
    - Using SAML (Security Assertion Markup Language 2.0), you can give your federated users single sign-on (SSO) access to the AWS Management Console.
  - Multifactor authentication
    - multiple independent authentication mechanisms
  - provide temporary access for users/devices and services
  - set up password rotation policy
  - integrates with many different AWS services
  - supports PCI DSS compliance
- Key terminology
  - *users*: end users
    - root account
      - account created when first setup AWS account
      - full admin access
      - always setup Multifactor authentication on root account
      - shouldn't be using day-to-day
    - new users have NO permissions when created (least privilege)
    - new users are assigned access key ID and secret access keys when created
  - programmatic access (CLI)
    - access key ID
    - secret access key (only visible once after

- creation)
    - never share access key among developers
    - don't upload to GitHub
  - console access (browser)
    - password
- *groups*
  - a collection of users
  - under one set of permissions
  - Users inherit the permissions of their group
  - easier to collectively manage users who need the same permissions
- *policies*
  - a JSON document that defines permissions
 

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "*",
      "Resource": "*"
    }
  ]
}
```
  - give permissions as to what a user/role/group can do
- *roles*
  - a set of defined access
  - create roles and assign them to AWS resources
  - a secure way to grant permissions to entities
    - IAM user
    - application code
    - AWS service
    - users with identity federation
  - example: EC2 talk to S3
    - instead of assigning user access keys, give them roles
    - delete saved credentials `rm ~/.aws/credentials` and `rm ~/.aws/config`

- create role -> EC2 -> AmazonS3FullAccess
  - go to EC2 -> actions -> instance settings -> attach/replace IAM role
- preferred from security perspective
  - allow you to not use access key id and secret access key
  - controlled by policies
  - if you change a policy it takes effect immediately
  - can attach/detach roles to running EC2 without having to stop or terminate the instances
- Advanced IAM

## 2 EC2

### 2.1 EC2 101

- Elastic Compute Cloud
- resizable compute capacity (virtual machines) in the cloud
- Instance purchasing options
  - *on-demand*
    - fixed rate, low cost, flexibility
      - no up-front payment
      - no long-term commitment
    - applications with short term, spiky, unpredictable workloads
    - Linux: by the second
    - Windows by the hour
  - *reserved instances*
    - contract 1 year / 3 years
    - upfront charge with discounted hourly price
      - standard RI: up to 75% off on-demand
      - convertible RI: up to 54% off on-demand, have capacity to change your instance types
      - scheduled RI: launch within the time window you reserved
    - applications with steady state or predictable usage
    - can't change region
  - *spot*
    - bid price
    - if your application have flexible start and end times
    - applications only feasible at very low compute prices
      - genomics, pharmaceutical, chemical companies
    - if your spot instance is terminated by AWS, you won't be charged for a partial hour of usage
    - if you terminate the instance yourself you will be charged for the complete hour
    - pay for what you bid on
  - *dedicated hosts*



- physical EC2 servers dedicated for your use
- useful for regulatory requirements or licenses that may not support multi-tenant virtualization or cloud deployments
- can be purchased on-demand or as a reservation
- Instance Types - F1 G1 I3 G3 H1 T2 D2 R4 M5

| Family | Speciality                    | Use Case  | Memorize    |
|--------|-------------------------------|---|-------------|
| F1     | field programmable gate array | Genomics research, financial analytics, realtime video processing, big data |             |
| I3     | high speed storage            | NoSQL DBs, data warehousing   | IOPS        |
| G3     | graphics intensive            | video encoding/3D application streaming                                     | graphics    |
| H1     | high disk throughput          | MapReduce, distributed file systems (HDFS, MapR-FS)                         | high        |
| T2     | lowest cost, general purpose  | web servers, small DBs  |             |
| D2     | dense storage                 | file servers, data warehousing, Hadoop                                      | density     |
| R4     | memory optimized              | memory intensive apps/DBs   | RAM         |
| M5     | general purpose               | applications servers  | main choice |



|    |                                 |                           |                |
|----|---------------------------------|---------------------------|----------------|
| C5 | compute<br>optimized            | CPU intensive<br>apps/DBs | compute        |
| P3 | graphics/general<br>purpose GPU | ML, bitcoin<br>mining     | pics           |
| X1 | memory<br>optimized             | SAP HANA/Apache<br>Spark  | extreme memory |

- How I remember them now;
  - F for FPGA
  - I for IOPS
  - G - Graphics
  - H - High Disk Throughput
  - T cheap general purpose (think T2 Micro)
  - D for Density
  - R for RAM
  - M - main choice for general purpose apps
  - C for Compute
  - P - Graphics (think Pics)
  - X - Extreme Memory

- created from Amazon Machine Image (AMI)
  - marketplace & community has pre-built AMI for purchase
- key pair
  - to login to your EC2
  - `chmod 400 key.pem`
  - save in `~/.ssh`
  - `ssh ec2-user@public_ip -i key.pem`
- to using the instance as a web server
  - `sudo su` to elevate access to root
  - `yum update -y` update the OS and relevant packages
  - `yum install httpd -y` to install Apache
  - `service httpd start` to start the server
  - `chkconfig httpd on` to turn Apache on automatically if instance restarts
  - `service httpd status` to check if Apache is running
  - `cd /var/www/html` go to root directory for web server

- `vim index.html` to add content in webpage
- `<html><body><h1>Hello</h1></body></html>`
- go to web browser with the public ip, you should see the content you put
- 1 subnet == 1 availability zone
- Bootstrap Script
  - Advanced details
  - runs at root level, executes commands in chronological order
  - automating software installs and updates
 

```
#!/bin/bash
yum install httpd php php-mysql -y
yum update -y
chkconfig httpd on
service httpd start
echo "<?php phpinfo();?>" > /var/www/html/index.php
cd /var/www/html
wget https://s3.amazonaws.com/acloudguru-production/connect.php
connect.php `` `<?php $username = "acloudguru";
$password = "acloudguru"; $hostname =
"yourhostnameaddress"; $dbname = "acloudguru";
//connection to the database $dbhandle =
mysql_connect($hostname, $username, $password) or
die("Unable to connect to MySQL"); echo "Connected to
MySQL using username - $username, password - $password,
host - $hostname
"; $selected = mysql_select_db("$dbname",$dbhandle) or
die("Unable to connect to MySQL DB - check the database
name and try again."); ?> `` `
```
- to get information about an instance
  - `curl https://169.254.169.254/latest/meta-data/`
  - `curl https://169.254.169.254/latest/user-data/`  
bootstrap scripts
- Termination protection
  - prevent your instance from being accidentally terminated from the console, CLI, or API.
  - disabled by default

- The `DisableApiTermination` attribute does not prevent you from terminating an instance by initiating shutdown from the instance (using an operating system command for system shutdown) when the `InstanceInitiatedShutdownBehavior` attribute is set.

## **2.2 Security Group**

- Updated rule take effect immediately
- Inbound
  - everything blocked by default
  - white list
    - allow rules only, no deny rules
    - use Network Access Control Lists to block specific IP
- Outbound
  - all outbound traffic is allowed
- Stateful
  - If you create an inbound rule allowing traffic in, that traffic is automatically allowed back out again
- EC2: Security Groups = m:m

## **2.3 EBS Volume**

- block storage
  - With block storage, files are split into evenly sized blocks of data, each with its own address but with no additional information (metadata) to provide more context for what that block of data is.
  - Another key difference is that block storage can be directly accessed by the operating system as a mounted drive volume, while object storage cannot do so without significant degradation to performance.
- EC3 block storage
  - Elastic Block Storage (EBS) volume
  - instance store volume
- Elastic Block Storage
  - root device volume
    - or boot volume
    - where Linux/Windows is installed (like C://)

- additional drive
  - attached to an EC2
  - create a file system on it
  - run a database on it
- need to be in same availability zone as the EC2 instance
- automatically replicated to avoid failure
- Types

| EBS Types                | AWS Name | Disk | Feature   | Use Case  |
|--------------------------|----------|------|---|---|
| general purpose SSD      | GP2      | SSD  | balances price & performance<br>(3 ~ 10,000 IOPS per GB)                | most workloads                                      |
| provisioned IOPS SSD     | I01      | SSD  | high performance<br>(> 10,000 IOPS)                                     | I/O intensive application; DB (relational or NoSQL) |
| throughput optimized HDD | ST1      | HDD  | frequent access, throughput intensive; can't be a boot volume           | Big data, data warehouse, log processing            |
| cold HDD                 | SC1      | HDD  | less frequently accessed workloads, lowest cost; can't be a boot volume | file server   |
| EBS magnetic (standard)  |          | HDD  | legacy generation; only magnetic  | cheap, infrequent data access                       |

volume that  
is bootable

---

\*HDD: magnetic storage volumes

- EBS lab
  - add an additional volume to an EC2
  - `sudo su` then `lsblk` to show all volumes
    - `xvda` has `MOUNTPOINT: /` means that's your root volume
    - `xvdf` is the EBS you attached, with no `MOUNTPOINT`
  - `file -s /dev/xvdf` if returns `/dev/xvdf: data` then it means there is no data on the volume
  - create a file system on the additional volume: `mkfs -t ext4 /dev/xvdf`
  - `file -s /dev/xvdf` should return `Linux rev 1.0 filesystem data` now
  - mount the volume to a directory: `cd` then `mkdir filesystem` then `mount /dev/xvdf /filesystem`
  - `lsblk` should now show `MOUNTPOINT: /filesystem` for the additional volume
  - use the additional volume: `cd filesystem` then `echo "hello" > hello.txt`
  - unmount the volume: `cd` then `umount -d /dev/xvdf`
  - detach the volume in AWS console
  - `lsblk` should only show `xvda` now
  - `cd filesystem` then `ls` should return nothing, `hello.txt` was deleted
  - EBS volume -> actions -> create snapshot (encrypted)
  - snapshot -> create volume
  - EBS volume -> actions -> attach volume to EC2
  - `lsblk` should show `xvdf` now, `file -s /dev/xvdf` should say there is a filesystem on it
  - `mount /dev/xvdf /filesystem` then `cd /filesystem` then `ls` you should see `hello.txt`
- volume size can be changed on the fly
  - size
  - storage type

- When instance terminated
  - root EBS volume
    - deleted by default
    - can turn termination protection on
  - additional ECS volumes persists
- Encryption
  - can't encrypt EBS root volume of default AMI
  - to encrypt root volume
    - use OS (bitlocker if Windows)
    - take a snapshot -> create a copy and select encrypt
    - > create image from the encrypted copy -> launch a new instance from the AMI (only larger instances available)
  - can encrypt additional volume
- Snapshot
  - exist on S3
  - time copies of the volumes
  - Incremental
  - can be taken when instance is running, but stop the instance first if the EBS serves as root device to ensure consistency
  - Encryption
    - snapshots of encrypted volumes are encrypted automatically
    - volumes created from encrypted snapshots are encrypted automatically; volumes created from unencrypted snapshots are unencrypted automatically
    - you can share snapshots, but only if they are unencrypted
      - with other AWS accounts
      - public, sell in marketplace
  - can be used to move EC2 volume to another availability zone
- to move EC2 to another AZ
  - take a snapshot
  - create an AMI from the snapshot
  - use the AMI to launch a EC2 in a new AZ
- to move EC2 to another region

- take a snapshot
- create an AMI from the snapshot
- copy AMI to another region
- use the AMI to launch a EC2 in the new region
- Can create AMI (Amazon Machine Image) from EBS-backed instances and snapshots
- EBS vs Instance Store
  - instance store (ephemeral storage)
    - The data in an instance store persists only during the lifetime of its associated instance
    - can't be stopped, can reboot or terminate
      - will lose data if stopped, or underlying disk drive fails or instance terminates
      - won't lose data if reboot (intentionally or unintentionally)
      - ROOT volumes will be deleted on termination
  - You can attach additional instance store volumes to your instance. You can also attach additional EBS volumes after launching an instance, but not instance store volumes.
  - the root device for an instance launched from the AMI is a instance store created from a template stored in S3 (can take a little longer than EBS backed volumes)
- EBS backed volumes
  - the root device for an instance launched from the AMI is a EBS volume created from a EBS snapshot
  - can be stopped and snapshotted
  - won't lose data if stopped
  - won't lose data if reboot
  - by default, ROOT volumes will be deleted on termination, but can change setting to keep it

## **2.4 Command Line CLI**

- prerequisite
  - setup access in IAM:
  - add user -> programmatic access (access key ID and secret access key)

- configure credentials
  - `aws configure` to put in access key ID and secret access key
- example commands
  - `aws s3 ls` list all buckets
  - `aws s3 mb s3://test-bucket` create a new bucket
  - `echo "test" > hello.txt` create a new file
  - `aws s3 cp hello.txt s3://test-bucket` copy file to bucket
  - `aws s3 ls s3://test-bucket` list files in bucket
- install on your laptop

## 2.5 CloudWatch

- monitoring performance
- CloudTrail is about auditing API calls
- AWS services and user applications
- Interval
  - every 5 minutes by default
  - can have 1 minute interval if turn on detailed monitoring
- Features
  - Alarms
  - Dashboards
  - Events
  - Logs
- Cloud Watch vs Access Logs
  - access logs: requests or connections, the time it was received, the client's IP address, latencies, request paths, and server responses

## 2.6 EFS - Elastic File System

- supports the Network File System version 4 (NSFv4) protocol
- only pay for the storage you use, no pre-provisioning
- can scale to TB
- can support thousands of concurrent NFS connections
- can't share EBS with multiple EC2
- can create EFS mount
- stored multi AZ within a region



- read after write consistency

## **2.7 Spread placement groups**

- EC2 attempts to place instances so that they spread out across underlying hardware to minimize correlated failures by default
- Spread placement groups configures the placement of a group of instances
- 2 types
  - Cluster
    - packs instances close together inside 1 AZ
    - low-latency network performance
    - high throughput
    - tightly-coupled node-to-node communication that is typical of HPC applications
  - spread placement group
    - critical instances
    - different hardware
- unique namespace within AWS account
- can't merge placement groups
- can't move existing instance to a placement group
- maximum of 7 running instances per Availability Zone

## **2.8 Load Balancer**

- Balance load across web servers
- Types of Load Balancers
  - *Application load balancer*
    - layer 7 (application layer): application aware
    - intelligent, advanced request routing
    - best suited for balancing HTTP and HTTPS traffic
  - *Network load balancer*
    - layer 4 - connection level
    - TCP traffic
    - extreme performance: millions of requests per second
    - most expensive, usually used in production
  - *classic load balancer / elastic load balancer*
    - balancing HTTP/HTTPS applications use Layer 7-

- specific features, e.g. x-forwarded and sticky sessions
  - Public IP -> load balancer -> private IP to server, the x-forwarded-for header contains the public IP (IPv4 address)
  - also Layer 4 load balancing for applications that rely purely on the TCP protocol
  - low cost
  - legacy
- 504 Error
  - gateway timed out
  - application not responding
  - -> trouble shoot the application
    - web server / db
    - scale it up or out if necessary

## **3 Databases**

### **3.1 Relational Database**

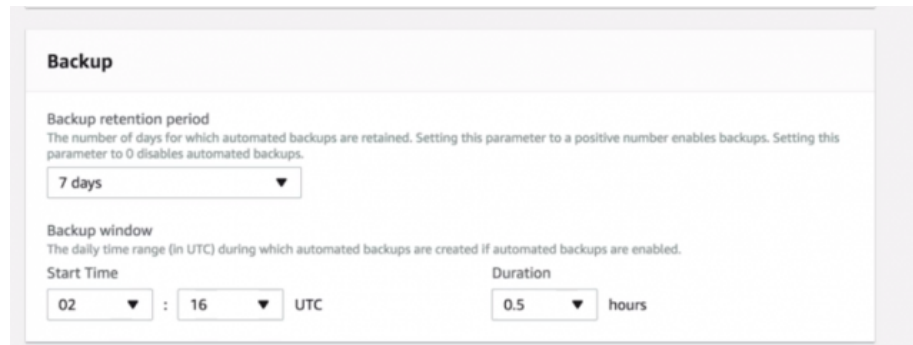
- since the 70s
- database
- tables
- row
- fields (columns)

### **3.2 RDS (OLTP)**

- relational database service
- runs on VM
  - but you can't log into these VMs
  - no OS level access
  - AWS patches the OS and DB
  - NOT serverless
  - Aurora Serverless IS serverless
- 6 Engines
  - MySQL
  - PostgreSQL
  - Oracle
  - SQL Server
  - MariaDB (fully compatible with MySQL)
  - Aurora (not available for free tier yet)
- Instance Types
  - optimized for memory
  - optimized for performance
  - optimized for I/O
- Storage
  - general purpose SSD
  - provisioned IOPS SSD: for I/O-intensive workloads, particularly database workloads
  - magnetic: backward compatibility
  - You can create MySQL, MariaDB, and PostgreSQL RDS DB instances with up to 32 TiB of storage. You can create Oracle RDS DB instances with up to 64 TiB of storage.

You can create SQL Server RDS DB instances with up to 16 TiB of storage. For this amount of storage, use the Provisioned IOPS SSD and General Purpose SSD storage types.

- 2 types of Backups on AWS
  - *automated backups*



The screenshot shows the 'Backup' configuration page for an AWS RDS instance. It includes two main sections: 'Backup retention period' and 'Backup window'. The 'Backup retention period' section has a dropdown menu set to '7 days'. The 'Backup window' section includes a 'Start Time' dropdown set to '02', a colon separator, a 'Duration' dropdown set to '0.5', and a 'UTC' label. Below the 'Start Time' dropdown is another dropdown set to '16'. The 'Duration' dropdown is followed by the word 'hours'.

- Recover to any point in time within a retention period
  - 1 ~ 35 days
- daily snapshot + transaction logs throughout the day
  - choose the most recent daily snapshot, then apply the logs
  - allows point in time recovery down to a second within the retention period
- enabled by default
- backup data stored in S3
- free storage space equal to the size of your DB
- taken during a defined window, you may experience latency
  - change to backup window takes effect immediately
- deleted after deletion of original RDS instance
- *snapshots*
  - done manually, user initiated
  - stored even after deletion of original RDS instance
- when restore a backup/snapshot, the restored version of the DB will be a new RDS instance with a new DNS endpoint
- Encryption

- supporting all 6 engines
- done using the KMS service (Key Management Service)
- If the RDS instance is encrypted, the data stored, the automated backups, read replicas and snapshots are also going to be encrypted
- Multi-AZ
  - When you provision a Multi-AZ DB Instance, Amazon RDS automatically creates a primary DB Instance and synchronously replicates the data to a standby instance in a different Availability Zone (AZ)
  - for disaster recovery only
    - synchronous replications
    - exact copy of main instance in another AZ
    - not for improving performance (use read replicas for performance improvement)
    - can't use secondary DB instances for writing purposes
  - auto failover
    - DB maintenance/DB instance failure/availability zone failure
    - AWS will point the endpoint to the secondary instance
    - Reboot from failover: force change AZ
    - Loss of availability in primary Availability Zone
    - Loss of network connectivity to primary Compute unit
    - failure on primary Storage
    - failure on primary
  - same DNS routed to secondary DB
    - CNAME changed to standby DB
  - supporting all 5 other engines besides Aurora
    - Aurora is fault tolerant itself so no need for multi-AZ
- Read replica
  - performance, scaling
  - spread load across multiple read replicas
  - for read-heavy DB workload
  - must have automatic backups turned on

- read-only copy of your production DB
- asynchronous replications
  - up to 5 of any database
  - can have read replicas of read replicas (latency)
  - each read replica has its own DNS end point
  - can have multi-AZ
  - can be in another region
  - can be created for multi-AZ DB
  - can be Aurora or MySQL
- can be promoted to master as their own database, breaks the replication though
- read from different DB but only write to 1 primary
  - read directed first to primary DB and then sent to secondary DB
- no auto failover
- supporting
  - MySQL
  - PostgreSQL
  - MariaDB
  - Aurora
- no charge for replicating data from your primary RDS instance to your secondary RDS instance
- Security Group
  - Technically a destination port number is needed, however with a DB security group the RDS instance port number is automatically applied to the RDS DB Security Group.
- RDS Reserved instances
  - reserve a DB instance for a one or three year term
  - discount compared to the On-Demand Instance pricing
  - available for multi-AZ deployments
- database on EC2
  - EBS
- endpoint
  - DNS address
  - never given a public IPv4 address
  - AWS manages the mapping from DNS to IPv4
- security group

- edit inbound rule to allow MYSQL/Aurora traffic from web server
- add in the web server security group instead of an IP address

### **3.3 Non-relational databases**

- collection
- document
- key-value pairs
- JSON

### **3.4 Dynamo DB**

#### **3.5 Data Warehousing - Redshift (OLAP)**

- business intelligence, reporting
- tools: Cognos, Jaspersoft, SQL Server Reporting Services, Oracle Hyperion, SAP NetWeaver
- PB scale, large and complex datasets
- use a separate data store to not affect production DB performance
- different architecture of DB and infrastructure layer
  - single Node (160 GB)
  - multi-node
    - leader node
      - manages clients connections and receives queries
    - compute node
      - stores data and perform queries
      - up to 128
- Advanced Compression
  - compression based on columns instead of rows
    - data of same type
    - stored sequentially on disk
  - doesn't require indexes or materialized views
    - use less space
  - auto samples data and selects most appropriate compression scheme when loading data
  - massively parallel processing (MPP)
- Backups

- enabled by default with 1 day retention period
  - Max retention period 35 days
- always attempts to maintain at least 3 copies of your data
  - original on compute node
  - replica on compute node
  - backup in S3
- asynchronously replicate your snapshots to S3 in another region for disaster recovery
- Billing
  - billed for compute node hours
    - 3 node cluster run for 1 day will incur 72 billable hours
    - only compute node, not leader node
  - billed for backup
  - billed for data transfer (only within a VPC, not outside it)
- Security
  - encrypted in transit using SSL
  - encrypted at rest using AES-256
  - by default redshift takes care of key management
    - can manage your own key through
      - HSM
      - KMS key management service
- Availability
  - currently only available in 1 AZ
  - can restore snapshots to new AZ in the event of an outage

### **3.6 ElastiCache**

- in-memory caching
- improves web applications performance
  - caching frequent queries
  - pull from in-memory cache instead of disk-based DB
  - takes off load from production DB
- for read-heavy DB workload
  - social networking
  - gaming



- media sharing
- Q&A portals
- or compute-intensive workload
  - recommendation engine
- 2 engines
  - Memcached
    - object caching
    - simple cache to offload DB
    - scale horizontally (scale out)
    - multi-threaded performance
    - no persistence
    - managed as a pool that can grow and shrink, similar to EC2 auto scaling group
    - individual nodes are expendable
    - automatic node replacement
    - auto discovery
  - Redis
    - in-memory key-value store that supports data structures like sets and lists (advanced data types)
    - support master/slave replication
    - support multi-AZ with failover
    - ranking & sorting: leaderboards
    - pub/sub capabilities
    - persistence
    - Backup & restore capabilities
    - managed more as a relational database
    - Redis clusters are managed as stateful entities that include failover, similar to RDS
- if a DB is having a lot of load, you should use ElastiCache if DB is read-heavy and not prone to frequent changing; if it's because management keep running OLAP transactions, use Redshift

### **3.7 Aurora**

- MySQL & PostgreSQL compatible
- open source
- cost effective

- Scalility
- auto scaling storage
- start with 10 GB, scales in 10GB increments to 64 TB
- compute resources: scale up to 32vCPUs and 244GB of memory
- Availability
- 2 copies of your data is contained in each AZ
- minimum of 3 AZ
- -> 6 copies of your data
- designed to handle the loss of
  - up to 2 copies of data without affecting DB write availability
  - up to 3 copies without affecting read availability
- Self-healing
- automated backups always enabled
  - doesn't impact performance
- snapshots
  - doesn't impact performance
  - can be shared with other AWS accounts
- Performance
- read replicas
  - Aurora replicas: up to 15
  - MySQL replicas: up to 5
- Automated failover from Aurora to Aurora read replica
- No Automated failover from Aurora to MySQL read replica
- Endpoints
  - cluster: primary instance for write operations
  - reader: read replicas
  - instance: individual instance

## 4 Route53

### 4.1 DNS 101

- Naming
  - The "route" part comes from the historic "Route 66" of the USA
  - 53 is port 53 for TCP and UDP
- DNS
  - convert human friendly domain names -> Internet Protocol (IP) address
  - IP addresses are used by computers to identify each other on the network
  - IPv4 or IPv6
    - IPv4
      - 32 bit field
      - 4 billion addresses
      - running out
    - IPv6
      - 128 bits
      - 340 undecillion addresses
- top level domains
  - string of characters separated by dots
  - last word is top level domain
    - .com
    - .edu
    - .gov
  - second level domain name
    - .co.uk: .uk is top level, .co is second level domain
  - Controlled by the Internet Assigned Numbers Authority (IANA)
  - domain registrars
    - assign domain names directly under 1 or more top-level domains
    - Amazon
    - GoDaddy.com
    - 123-reg.co.uk etc.

- SOA record
  - start of authority record
- NS record
  - name server record
  - used by top level domain servers to direct traffic to the content DNS server
  - "acloudguru.com" -> .com server -> NS records -> SOA (DNS records)
- different types of DNS record
  - A record (A = address)
    - points to a IPv4 address
  - AAAA record
    - IPv6 address
  - CName
    - Canonical Name
    - resolve 1 domain name to another
    - "m.acloudguru.com" -> "mobile.acloudguru.com" -> "IPv4"
    - BATMAN -> "Bruce Wayne" -> 412-412-4121
  - Alias Records
    - map resource records to ELB, CloudFront, or S3 buckets websites
    - DNS -> target name
    - can't be used for naked domain name (without a www.)
    - has to be an A record or an alias
    - always choose an alias record over a CName in an exam scenario
    - Provides a router53-specific extension to DNS functionality
  - MX record
    - Mail exchange
  - PTR record
    - reverse of an A Record
    - address -> domain name
- TTL
  - time to live
  - length a DNS record is cached in the resolving server

- or user local PC
  - lower TTL, faster changes to DNS records take to propagate throughout the internet
- naked domain name
  - domain.com instead of www.domain.com
  - use A record with an Alias
  - Alias target: S3, ELB, CloudFront, Elastic Beanstalk, etc

## 4.2 Route53

- Domain Registration
  - can buy domain names directly in AWS
  - and map your domain name to
    - EC2
    - Load Balancer
    - S3 bucket
  - takes up to 3 days to register
- Routing policies
  - simple routing
    - 1 record with multiple IP address
    - Route53 returns a random value
    - hosted zones -> create record set -> naked name -> A record -> put 3 IPs for EC2 in
    - access tinabu.com and the server will change
    - can modify TTL
  - weighted routing
    - split traffic based on weights
    - e.g. 20% Paris 80% N. Virginia
  - latency-based routing
    - route your traffic based on the lowest network latency for your end user
    - fastest response time
  - failover routing
    - create an active/passive set up
    - e.g. primary site in EU-WEST-2 and secondary disaster recovery site in AP-SOUTHEAST-2
    - failover to passive if active is down
  - geolocation routing

- choose where your traffic will be sent based on the user's geographic locations
  - continent / country based
- geoproximity routing
  - let Route53 route traffic based on user geographic location and your resources
  - must use Route53 traffic flow
  - you can influence it with "bias"
  - traffic policy tab
- multivalue answer routing
  - like simple routing: configure Route53 to return multiple values
  - but plus health checks and return only healthy resources
- Health check
  - create on individual record sets
  - Add health check to A record
  - if a record fails a health check, it will be removed from Route53 until it passes the health check again
  - can set SNS notificatin for health check failures
- Lab
  - Register a domain
    - choose a domain name: .com, .co, etc
    - fill registrant contact information
  - Hosted zones -> Create Record Set
  - Create 3 EC2 in 3 regions
    - N. Virginia 3.87.24.95
    - Seoul 54.180.113.112
    - Paris 35.181.48.173
  - Create a load balancer
    - Application load balancer: internet-facing, IPv4
    - configure target group and health check
    - add EC2s to target group
    - ELBs don't have pre-defined IPv4 addresses, you resolve to them using a DNS name
  - Go to Route53, hosted zones
    - create record set, A record
    - add ELB as the target

- go to `domain.com` instead of public-ip in browser

## 5 S3 (Simple Storage Service)

- object-based storage
  - key: name
  - value: data (files)
    - 0 byte to 5 TB
    - unlimited storage
    - stored in buckets (folder)
    - all newly created buckets are private by default
    - if uploaded successful, HTTP 200 code
    - can setup logs
  - version ID
  - metadata
  - subresources
    - Access Control List
    - Torrent
- universal name space
  - bucket name needs to be unique globally
  - `https://s3-eu-west-1.amazonaws.com/bucket_name`
- data consistency
  - read after write consistency: PUTS of new object
  - eventual consistency: overwrite PUTS and DELETES
- guarantee
  - 11 x 9s durability (won't be lost)
- Tiered Storage

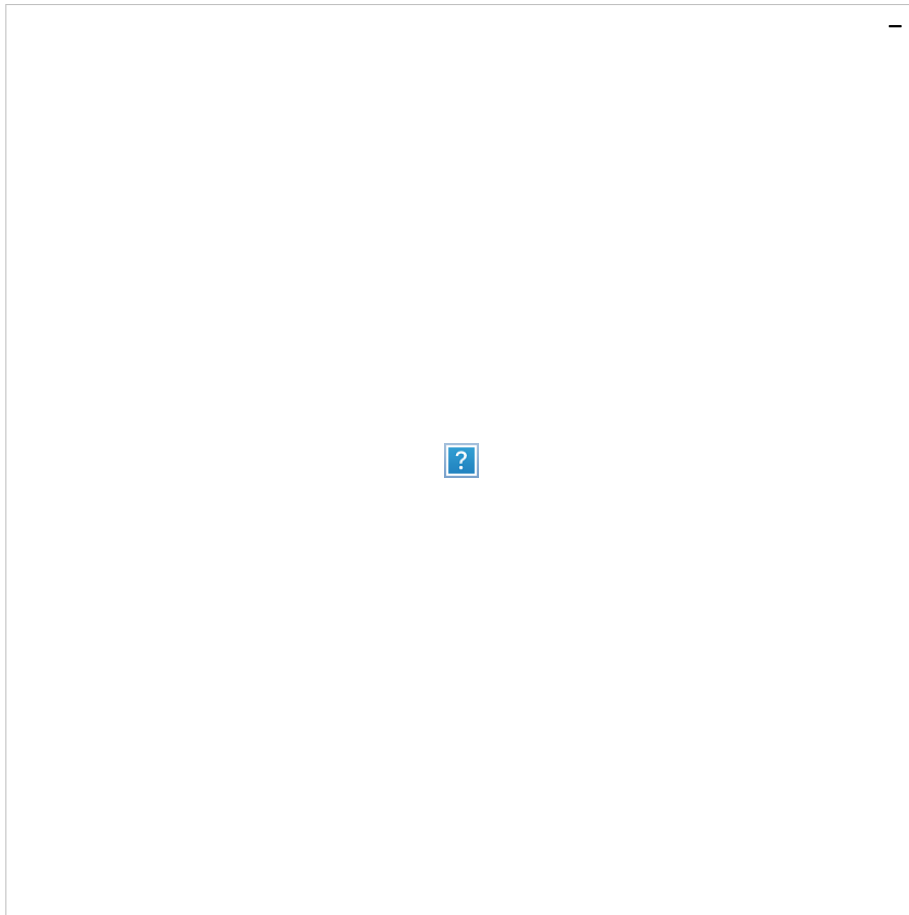
| S3 Type     | Availability | Durability | Features   | Retrieval Time |
|-------------|--------------|------------|--|----------------|
| S3 standard | 99.99%       | 11 * 9     | stored redundantly across multiple device in multiple facilities; sustain loss | ms             |

|                                |       |        |  |          |
|--------------------------------|-------|--------|--|----------|
|                                |       |        | of 2<br>facilities<br>concurrently                                     |          |
| S3 - IA                        | 99.9% | 11 * 9 | rapid but<br>infrequent<br>access;<br>charged with<br>retrieval<br>fee | ms       |
| S3 One Zone<br>- IA            | 99.5% | 11 * 9 | low cost,<br>infrequently<br>access; no<br>multi-AZ                    | ms       |
| S3 -<br>Intelligent<br>Tiering |       |        |  | ms       |
| S3 Glacier                     |       | 11 * 9 | data<br>archive; low<br>cost   | min ~ hr |
| S3 Glacier<br>Deep Archive     |       |        | lowest cost  | 12 hours |

---

- MFA delete
- secure data with
- access control list
- bucket policies
- pre-signed URLs
- Pricing
- storage
- requests
- storage management (tiers)
- data transfer
- transfer acceleration
  - data transfer between user and S3 bucket
  - use CloudFront's edge locations
  - amazon backbone network
- cross region replication





sync a bucket to another region

- versioning must be enabled on both source and destination
- regions must be unique
- files in an existing bucket are not replicated automatically
- all subsequent updated files will be replicated automatically
- Delete markers are not replicated
- deleting individual versions or markers will not be replicated
- encryption
  - default: in transit
  - SSL/TLS: HTTPS
- at rest
  - server side encryption

- S3 managed keys - SSE-S3
- AWS key management service, SSE-KMS
- with customer provided keys - SSE-C
- client side encryption
- encrypt data -> upload to S3
- client library such as Amazon S3 Encryption Client
- version control
- billed for total size
- Lifecycle management
- automates moving your objects between the storage tiers
- can be used in conjunction with versioning
- can be applied to current or previous versions

#### **4 Serverless Computing**

#### **5 DynamoDB**

#### **6 KMS & Encryption**

#### **7 Other AWS Services**

#### **8 Developer Theory**

#### **9 Monitoring**