API Gateway

**IAM (Identity Access Management) – Summary**

Identity Access Management is used to define user access permission within AWS.

There are three types of Policies

-Managed Policies – AWS managed default policies – recommended by AWS

-Customer Managed Policies – Customer can create policies of their own.

-Inline Policies – it is embedded within group, user or role. It has strict 1:1 relationship between entity and policy.

**Customer Master Key**

| **Type of CMK** | **Can view CMK metadata** | **Can manage CMK** | **Used only for my AWS account** | **Automatic rotation** |
| --- | --- | --- | --- | --- |
| [Customer managed CMK](https://docs.aws.amazon.com/kms/latest/developerguide/concepts.html#customer-cmk) | Yes | Yes | Yes | Optional. Every 365 days (1 year). |
| [AWS managed CMK](https://docs.aws.amazon.com/kms/latest/developerguide/concepts.html#aws-managed-cmk) | Yes | No | Yes | Required. Every 1095 days (3 years). |
| [AWS owned CMK](https://docs.aws.amazon.com/kms/latest/developerguide/concepts.html#aws-owned-cmk) | No | No | No | Varies |

Envelope Key

**Amazon EC2 (Elastic Compute Cloud) – Summary**

EC2 provides new server instances within minutes.

It helps to pay for the services that you actually use.

**EC2 Pricing Options:**

**On Demand:** Allows user to pay by the hour or by the second without any commitment.

**Reserved**: Provides a capacity reservation with commitment of 1 to 3 years.

**Spot:** Enables user to bid for an instance capacity. It’s beneficial for application with flexible executions.

**Dedicated Servers:** These are Physical dedicated servers, which allow existing server-bound software licenses.

**EC2 Instance Types**

\*\* Detailed knowledge not required for Associate level exam \*\*

**FIGHT DR MC PX**

F1 – Field Programmable Gate Array

I3 – High Speed Storage

G3—Graphics Intensive

H1 – High Disk Throughput

T2 – Lowest Cost General Purpose

D2 – Dense Storage

R4 -- Memory Optimized (RAM) -- Memory Intensive Apps/DBs

M5- General Purpose

C5 – Compute Optimized

P3 – Graphics/General Purpose GPU

X1—Memory Optimized – SAP HANA/Apache Spark etc (Extreme Memory)

Command to install Apache and make EC2 Server a web server

**Service httpd start**

**EBS—Elastic Block Storage**

It allows you to create block storage and attach them to EC2 Instances.

**EBS Volume Type:**

**General Purpose SSD (GP2):** Balance in price and performance, suitable for less than 10,000 IOps

**Provisioned IOps SSD (IO1):** Designed for I/O intensive applications, use if you need more than 10,000 IOps

**Throughput Optimized HDD (ST1):** Big data and cannot be root volume

**Cold HDD ():** File Server, lowest cost for infrequently accessed workloads, cannot be boot volume

**Magnetic Standard:** Lowest storage cost, can be boot volume

**Elastic Load Balancers**

**Application Load Balancers –** Works on **OSI layer 7,** can make clever routing decisions.

**Network Load Balancers –** Fast speed

**Classic Load Balancers –** Legacy ELB

**X-Forwarded** Provides private address from DNS to EC2 instance

**X-Forwarded For** providespublic IP address

**RDS – Backups, Multi AZ (Availability Zone) and Read Replicas.**

Automated backup can be configured for 1 to 35 days.

Multi Availability Zones are for Disaster Recovery.

Read Replicas are for performance improvement.

**Amazon S3 (Simple Storage Service) – Summary**

S3 is ideal for files storage, image storage but not for OS or Database storage.

Object Based storage.

Data is spread across multiple facilities.

File size can be from 0B to 5TB, storage size is unlimited.

S3 has universal namespace, it must be unique globally.

**Data consistency Model**

Read after write consistency is provided for PUTS of new objects.

Eventual consistency for overwrite PUTS and DELETES (propagates after some time)

S3 is object based.

Key is name of the file

Value is data of file

Version Id

Metadata

**S3 Storage Tiers/Classes**

S3: 99.99% availability, 99.(11 -9s) durability

S3 IA (Infrequently Accessed) – Lower fees than s3 but retrieval in charged.

S3 One Zone IA: 20% lower cost but 99.5% availability

Reduced Redundancy Storage: 99.99% durability

Glacier: Archival, take 3-5 hours to retrieve data (no real time access)

**S3 Charges:**

Storage per GB

Requests (Get, Put, Copy etc)

Storage Management Pricing

Data Management Pricing

Transfer Acceleration

**S3 Security:**

By default all newly created buckets are PRIVATE.

**Bucket Policies:** Applied at bucket level.

**Access Control List:** Applied at object level.

S3 buckets can be configured to create access logs.

Encryption:

In Transit: SSL/TLS (HTTPS)

At Rest (Server Side Encryption):

S3 Managed Keys: **SSE-S3**

AWS Key Management Service, Managed Keys: **SSE-KMS**

Server Side Encryption with Customer provided Keys: **SSE- C**

Client Side Encryption:

Client encrypt file before uploading to S3.

You can enforce encryption on all S3 files using bucket policy to deny all PUT requests without x-amz-server-side-encryption parameter.

CORS Configuration: Used for inter bucket access; need to provide Origin of request.

<https://aws.amazon.com/s3/faqs/> good read before exam.

CloudFront – Content Delivery Network

Edge Location:

Origin:

Distribution

Web Distribution

RTMP – Real Time Media Protocol

**Serverless Computing**

AWS Lambda is a compute service where you can upload your code and create lambda function.

AWS Lambda takes care of provisioning and managing servers that you need to run your code.

It is Function-as-a-service.

AWS Lambda can be used in following ways

Even Driven Compute Service – Runs a function on change even in s3 bucket or DB.

Request Driven Compute Service – Runs a function as response to HTTP request.

White Paper link: <https://d1.awsstatic.com/whitepapers/serverless-architectures-with-aws-lambda.pdf>

**AWS X-Ray Service:**

It is the service that collects data about requests that your application serves and provide tools that you can use to view, filter and gain insights into that data to identify issues and opportunity for optimization.

**XRay SDK Provides:**

* Interceptors to add to your code to track incoming HTTP request.
* Client Handlers to instrument AWS SDK client that your application uses to call other AWS services
* An HTTP client to use to instrument call to other internal and external HTTP web services.

**Dynamo DB**

For applications that need single digit millisecond latency at any scale dynamo DB can be used.

It supports both document and key-value data models.

Data is stored on SSD storage

**Indexes:**

**Local Secondary Index:** Can only be created while creating table

Cannot added, modified or removed later, has same partition key as table

It has different Sort Key

**Global Secondary Index:** It can created any time either at table creation time or later

Different Partition key and Sort Key

**Consistency Models**

Eventual consistent reads

Strongly consistent reads

Amazon Dynamo DB is low latency NOSQL database

Consists of Tables items and attributes

Supports two models document and Key-Value pair

JSON, HTML and XML document formats are supported

**Two types of PK**

Partition Key

Composite Key (Partition Key + Sort Key)

Access is controlled using IAM policies

**Calculation of Write /Read throughput**:

1KB Per second write throughput Write capacity Units can be allocated accordingly

4KB Strongly consistent read per second

2 \* 4KB Eventually consistent read

To Calculate how much read capcity units to be allocated first divide item size by 4KB (3/4 = 0.75)

Round it up to nearest whole number, multiply by reads needed.

**Elasticache:**

**Memecached**

**Redis**

**Dynamo DB Accelerator (DAX):** Provides in-memory caching for Dynamo DB tables

Improves response time for eventually consistent reads only

You point your API calls to DAX cluster instead of tables

If queries item gets cache hit then it is returned otherwise it will perform eventually consistent getItem operation

Not suitable for write intensive applications or applications that needs strongly consistent read model.

**Dynamo DB Transactions:**

ACID Transactions (Automic, Consistent, Isolated and Durable)

Read or Write multiple items across multiple tables as all or nothing expression

Dynamo DB TTL (Time to Live):

Time to Live attribute define expiry time for your data.

Great for removing irrelevant or old data

TTL is expressed as EPOCH time (Unix time) – counts time since 1-Jan-1970

Scan Page size

**Provisioned Throughput Exceeded Exception and Exponential Backoff:**

This exception comes when your read/write request rate is too high for capacity provisioned

SDK will automatically retries requests until successful

If SDK is not used then you can reduce request rate or use Exponential Backoff

Exponential Backoff provides progressively longer waits between consecutive retries

**Dynamodb: LeadingKeys** – Fine grained access control can be provided using IAM conditional parameter it allows access only to items where partition key matches their user id.

**KMS – Key Management Service**

**Other AWS Services**

**SQS – Simple Queue Service**

SQS is distributed message queuing system.

It allows decoupling of components to keep them independent

Pull based not push based

Standard Queue: Best effort ordering, Message delivered at least once.

FIFO Queue: Ordering is strictly preserved; messages are delivered only once, no duplicates.

**SNS – Simple Notification Service**

It is scalable and highly available service which allows application to push notifications from cloud.

Variety of message types are supported

Push Mechanism

**Elastic Beanstalk**

It is fastest and easiest way to deploy application on AWS.

Full Control over resources used by application can be retained by user or elastic beanstalk can do it.

**Elastic Beanstalk Deployment Policies:**

* **All at once** : deployment of everything at once and has downtime

Not ideal for mission critical system, if failed rollback needed to revert back to original version

* **Rolling** : Add new version in batches, capacity is reduced, instances are updated in batches

Not ideal for performance sensitive systems.

* **Rolling with additional Batch**: Launches additional batch of instances, deploy new versions in batches.

It maintains full capacity during deployment.

* **Immutable Deployment Updates**: Deploy new version in fresh group of instances in their own auto-scaling groups. Maintains full capacity, rollback is very easy. Preferable for mission critical applications.

**Kinesis**

There are three core Kinesis services.

Kinesis Video Streams: Securely stream video from connected devices to AWS for analytics and ML

Kinesis Data Streams: Build custom applications to process data in near real time.

Kinesis Firehose: capture, transform, load data streams into AWS data store for near real time analytics with BI tools.

**Continuous Integration and Continuous Deployment:**

Continuous Integration: Integrating or merging code changes frequently at least once per day. **CodeCommit**

Continuous Delivery: Automating build, test and Deployment. **CodeBuild** and **CodeDeploy**

Continuous Deployment: Fully automated release process, Code is deployed into staging or production as soon as it has successfully passed through release pipeline. **CodePipeline**

**CodeCommit:**

* Centralized Code Repository
* Enables Collaboration – Manages updates from multiple users
* Version Control – Tracks and Manages code changes

**CodeDeploy:**

* In-Place deployment : suitable for first time, capacity is reduced
* Blue Green Deployment: Full capacity is maintained, easy to switch or rollback
* Appspec file – OS, files, hooks
* Appspec.yml should be in root folder
* Run Order :

Before Block Traffic

Block Traffic

After Block Traffic

Application Stop

Download Bundle

Before Install

Install

After Install

Application Start

Validate Service

Before Allow Traffic

Allow Traffic

After Allow Traffic

**CodePipeline:**

Continuous Integration/Continuous Delivery Service – It orchestrates e2e software release.

It is fully automated

Integrates with AWS (Code commit, build and Deploy) and other third party tools like Jenkins

**Cloud Formation**

Cloud Formation is the service that allows user to manage, configure and provision AWS infrastructure as code.

Resources are defined using Cloud Formation Template.

Cloud Formation interprets template and make API calls to create resources which are defined in template.

It supports YAML and JSON.

**Benefits:**

Infrastructure provisioned consistently with fewer mistakes

Less time and efforts than manual configuration

Version control and peer review of template is possible

Free to use (Charged for what resources are created)

Can be used to manage and update dependencies

Can be used to rollback and delete entire stack as well

**Monitoring in AWS**

Cloud Watch: Performance monitoring

Cloud Trail: Monitors API Calls in AWS

AWS Config: Records state of AWS environment and can notify about changes