SECTION 1

Let's Get Started!

© Digital Cloud Training | https://digitalcloud.training

SECTION 2

AWS Accounts and IAM

© Digital Cloud Training | https://digitalcloud.training

AWS Account Overview

© Digital Cloud Training | https://digitalcloud.training

AWS Account Overview

It's an IAM best practice to create **individual users** and to avoid using the **Root** account

Unique **email address** required

The Root user has **full control** over the account

© Digital Cloud Training | https://digitalcloud.training

AWS Account Overview

Authentication: IAM principals authenticate to IAM using the console, API, or CLI

Authorization: IAM principals can then create resources across AWS Regions

All AWS **identities** and **resources** are created within the AWS account

© Digital Cloud Training | https://digitalcloud.training

Create your AWS Free Tier Account

© Digital Cloud Training | https://digitalcloud.training

What you need...

Credit card for setting up the account and paying any bills

Unique email address for this account

Check if you can use a **dynamic alias** with an existing email address

john@gmail.com

john+ACCOUNT-ALIAS-1@gmail.com

john+ACCOUNT-ALIAS-2@gmail.com

AWS account name / alias

Phone to receive an **SMS** verification code

© Digital Cloud Training | https://digitalcloud.training

Configure Account and Create a Budget and Alarm

© Digital Cloud Training | https://digitalcloud.training

Account Configuration

- Configure **Account Alias**
- Enable access to billing for **IAM users**
- Update **billing preferences**
- Create a **budget and alarm**

© Digital Cloud Training | https://digitalcloud.training

AWS IAM Overview

© Digital Cloud Training | https://digitalcloud.training

AWS Identity and Access Management (IAM)

© Digital Cloud Training | https://digitalcloud.training

IAM Users, Groups, Roles, and Policies

© Digital Cloud Training | https://digitalcloud.training

Users, Groups, Roles and Policies

© Digital Cloud Training | https://digitalcloud.training

IAM Users

© Digital Cloud Training | https://digitalcloud.training

IAM Groups

Admin Group

Eric

Sunil

Development Group

Ethan

Lee

Operations Group

Andrea

The user gains the **permissions** applied to the **group** through the **policy**

Groups are collections of users. Users can be members of up to 10 groups

The main reason to use groups is to apply **permissions** to users using **policies**

DigitalCloud TRAINING

© Digital Cloud Training | https://digitalcloud.training

IAM Roles

An **IAM role** is an IAM **identity** that that has specific **permissions**

© Digital Cloud Training | https://digitalcloud.training

IAM Policies

Policies are **documents** that define **permissions** and are written in JSON

AdministratorAccess

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "*",
            "Resource": "*"
        }
    ]
}
```

All permissions are **implicitly denied** by default

User

Group

Role

Identity-based policies can be applied to users, groups, and roles

Bucket Policy

```
{
    "Version": "2012-10-17",
    "Id": "Policy1561964929358",
    "Statement":[
    {
        "Sid": "Stmt1561964454052",
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::515148227241:user/Paul"
        },
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::dctcompany",
        "Condition": {
            "StringLike": {
                "s3:prefix": "Confidential/*"
            }
        }
    }
    ]
}
```

Resource-based policies apply to **resources** such as S3 buckets or DynamoDB tables

S3 Bucket

© Digital Cloud Training | https://digitalcloud.training

DigitalCloud TRAINING

Create an IAM User and Group

© Digital Cloud Training | https://digitalcloud.training

IAM Authentication and MFA

© Digital Cloud Training | https://digitalcloud.training

IAM Authentication Methods

© Digital Cloud Training | https://digitalcloud.training

Multi-Factor Authentication

Something you **know**:

EJPx!*21p9%

Password

Something you **have**:

Something you **are**:

© Digital Cloud Training | https://digitalcloud.training

Multi-Factor Authentication

Something you **know**:

IAM User

EJPx!*21p9%

Password

Something you **have**:

MFA Virtual MFA

e.g. Google Authenticator on your smart phone

MFA Physical MFA

Physical tokens can be purchased from **third parties**

© Digital Cloud Training | https://digitalcloud.training

DigitalCloud TRAINING

Enable Multi-Factor Authentication (MFA)

© Digital Cloud Training | https://digitalcloud.training

AWS Security Token Service (STS)

© Digital Cloud Training | https://digitalcloud.training

AWS Security Token Service (STS)

Credentials include:
- AccessKeyId
- Expiration
- SecretAccessKey
- SessionToken

EC2 Instance

Application

S3 Bucket

Temporary security credentials are returned

AWS STS

Instance Profile

IAM Role

EC2 attempts to assume role (sts:AssumeRole API call)

Trust policies control who can assume the role

```
{
  "Effect": "Allow",
  "Principal": {
    "Service": "ec2.amazonaws.com"
  },
  "Action": "sts:AssumeRole"
}
```

Trust Policy

Permissions Policy

Temporary credentials are used with identity federation, delegation, cross-account access, and IAM roles

© Digital Cloud Training | https://digitalcloud.training

Access Control Methods - RBAC & ABAC

© Digital Cloud Training | https://digitalcloud.training

Role-Based Access Control (RBAC)

Admin Group

Eric

Sunil

Development Group

Ethan

Lee

Operations Group

Andrea

Users are assigned permissions through policies attached to groups

Groups are organized by job function

Best practice is to grant the minimum permissions required to perform the job

© Digital Cloud Training | https://digitalcloud.training

Role-Based Access Control (RBAC)

Job function policies:

- Administrator
- Billing
- Database administrator
- Data scientist
- Developer power user
- Network administrator
- Security auditor
- Support user
- System administrator
- View-only user

The Billing managed policy is attached to the group

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "aws-portal:*Billing",
                "aws-portal:*Usage",
                "aws-portal:*PaymentMethods",
                "budgets:ViewBudget",
                "budgets:ModifyBudget",
                "ce:UpdatePreferences",
                "ce:CreateReport",
                "ce:UpdateReport",
                "ce:DeleteReport",
                "ce:CreateNotificationSubscription",
                "ce:UpdateNotificationSubscription",
                "ce:DeleteNotificationSubscription",
                "cur:DescribeReportDefinitions",
                "cur:PutReportDefinition",
                "cur:ModifyReportDefinition",
                "cur:DeleteReportDefinition",
                "purchase-orders:*PurchaseOrders"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS managed policies for job functions are designed to closely align to common job functions in the IT industry

© Digital Cloud Training | https://digitalcloud.training

Attribute-Based Access Control (ABAC)

© Digital Cloud Training | https://digitalcloud.training

DigitalCloud TRAINING

Assuming IAM Roles (Console)

© Digital Cloud Training | https://digitalcloud.training

SECTION 3

AWS Command Line Interface (CLI)

© Digital Cloud Training | https://digitalcloud.training

Install the AWS Command Line Interface (CLI)

© Digital Cloud Training | https://digitalcloud.training

Configure Credentials for the AWS CLI

© Digital Cloud Training | https://digitalcloud.training

Overview of Using the AWS CLI

© Digital Cloud Training | https://digitalcloud.training

Assuming IAM Roles (CLI)

© Digital Cloud Training | https://digitalcloud.training

SECTION 4

Amazon VPC, EC2, and ELB

© Digital Cloud Training | https://digitalcloud.training

Amazon VPC, Security Groups, and NACLs

DigitalCloud TRAINING

© Digital Cloud Training | https://digitalcloud.training

Amazon Virtual Private Cloud (VPC)

A **VPC** is a logically isolated portion of the AWS cloud within a region

Subnets are created within **AZs**

The **VPC router** takes care of routing within the VPC and outside of the VPC

You can launch **EC2 instances** into your VPC subnets

An **Internet Gateway** is attached to a VPC and used to connect to the Internet

Main Route Table

Destination	Target
10.0.0.0/16	Local
0.0.0.0/0	igw-id

The **route table** is used to configure the VPC router

© Digital Cloud Training | https://digitalcloud.training

Amazon VPC

Each **VPC** has a different block of IP addresses

CIDR stands for Classless Interdomain Routing

Region

VPC

CIDR 10.0.0.0/16

Availability Zone

Public subnet

10.0.0.0/24

Private subnet

10.0.2.0/24

Availability Zone

Public subnet

10.0.1.0/24

Private subnet

10.0.3.0/24

VPC

CIDR 10.1.0.0/16

Availability Zone

Public subnet

10.1.0.0/24

Private subnet

10.1.3.0/24

Availability Zone

Public subnet

10.1.2.0/24

Private subnet

10.1.4.0/24

Each subnet has a block of **IP addresses** from the CIDR block

You can create **multiple VPCs** within each region

© Digital Cloud Training | https://digitalcloud.training

Amazon VPC Components

VPC Component	What it is
Virtual Private Cloud (VPC)	A logically isolated virtual network in the AWS cloud
Subnet	A segment of a VPC's IP address range where you can place groups of isolated resources
Internet Gateway/Egress-only Internet Gateway	The Amazon VPC side of a connection to the public Internet for IPv4/IPv6
Router	Routers interconnect subnets and direct traffic between Internet gateways, virtual private gateways, NAT gateways, and subnets
Peering Connection	Direct connection between two VPCs
VPC Endpoints	Private connection to public AWS services
NAT Instance	Enables Internet access for EC2 instances in private subnets managed by you)
NAT Gateway	Enables Internet access for EC2 instances in private subnets (managed by AWS)
Virtual Private Gateway	The Amazon VPC side of a Virtual Private Network (VPN) connection
Customer Gateway	Customer side of a VPN connection
AWS Direct Connect	High speed, high bandwidth, private network connection from customer to aws
Security Group	Instance-level firewall
Network ACL	Subnet-level firewall

DigitalCloud TRAINING

Amazon VPC Core Knowledge

- A virtual private cloud (VPC) is a virtual network dedicated to your AWS account
- Analogous to having your own data center inside AWS
- It is logically isolated from other virtual networks in the AWS Cloud
- Provides complete control over the virtual networking environment including selection of IP ranges, creation of subnets, and configuration of route tables and gateways
- You can launch your AWS resources, such as Amazon EC2 instances, into your VPC

© Digital Cloud Training | https://digitalcloud.training

Amazon VPC Core Knowledge

- When you create a VPC, you must specify a range of IPv4 addresses for the VPC in the form of a Classless Inter-Domain Routing (CIDR) block; for example, 10.0.0.0/16
- A VPC spans all the Availability Zones in the region
- You have full control over who has access to the AWS resources inside your VPC
- By default you can create up to 5 VPCs per region
- A default VPC is created in each region with a subnet in each AZ

© Digital Cloud Training | https://digitalcloud.training

Security Groups and Network ACLs

© Digital Cloud Training | https://digitalcloud.training

Security Group Rules

Security groups support **allow** rules only

Inbound rules

Type	Protocol	Port range	Source
SSH	TCP	22	0.0.0.0/0
RDP	TCP	3389	0.0.0.0/0
RDP	TCP	3389	::/0
HTTPS	TCP	443	0.0.0.0/0
HTTPS	TCP	443	::/0
All ICMP - IPv4	ICMP	All	0.0.0.0/0

Separate rules are defined for outbound traffic

A source can be an **IP address** or **security group ID**

© Digital Cloud Training | https://digitalcloud.training

Network ACLs

Inbound Rules

Rule #	Type	Protocol	Port Range	Source	Allow / Deny
100	ALL Traffic	ALL	ALL	0.0.0.0/0	ALLOW
101	ALL Traffic	ALL	ALL	::/0	ALLOW
*	ALL Traffic	ALL	ALL	0.0.0.0/0	DENY
*	ALL Traffic	ALL	ALL	::/0	DENY

Outbound Rules

Rule #	Type	Protocol	Port Range	Destination	
100	ALL Traffic	ALL	ALL	0.0.0.0/0	ALLOW
101	ALL Traffic	ALL	ALL	::/0	ALLOW
*	ALL Traffic	ALL	ALL	0.0.0.0/0	DENY
*	ALL Traffic	ALL	ALL	::/0	DENY

NACLs have an explicit deny

Rules are processed in order

© Digital Cloud Training | https://digitalcloud.training

DigitalCloud TRAINING

Amazon EC2 Overview

© Digital Cloud Training | https://digitalcloud.training

Amazon Elastic Compute Cloud (EC2)

EC2 instances run Windows, Linux, or MacOS

An **EC2 instance** is a virtual server

EC2 hosts are **managed by AWS**

EC2 Host Server

A selection of **instance types** come with varying combinations of CPU, memory, storage and networking

© Digital Cloud Training | https://digitalcloud.training

Public, Private, and Elastic IP addresses

Type	Description
Public IP address	Lost when the instance is stopped Used in Public Subnets No charge Associated with a private IP address on the instance Cannot be moved between instances
Private IP address	Retained when the instance is stopped Used in Public and Private Subnets
Elastic IP address	Static Public IP address You are charged if not used Associated with a private IP address on the instance Can be moved between instances and Elastic Network Adapters

© Digital Cloud Training | https://digitalcloud.training

Public Subnets

Public Subnet Route Table

Destination	Target
172.31.0.0/16	Local
0.0.0.0/0	igw-id

© Digital Cloud Training | https://digitalcloud.training

Private Subnets

NAT = Network Address Translation

Public Subnet Route Table

Destination	Target
172.31.0.0/16	Local
0.0.0.0/0	igw-id

Private Subnet Route Table

Destination	Target
172.31.0.0/16	Local
0.0.0.0/0	nat-gateway-id

© Digital Cloud Training | https://digitalcloud.training

Create a Custom VPC with Subnets

© Digital Cloud Training | https://digitalcloud.training

VPC CIDR Block and Subnets

VPC CIDR Block	10	0	0	0
/16 Subnet Mask	255	255	0	0

Subnet Name	IPv4 CIDR block	Availability Zone	Route Table	Auto-assign Public IPv4
private-1a	10.0.3.0/24	us-east-1a	Private-RT	No
private-1b	10.0.4.0/24	us-east-1b	Private-RT	No
public-1a	10.0.1.0/24	us-east-1a	MAIN	Yes
public-1b	10.0.2.0/24	us-east-1b	MAIN	Yes

Has a route to an **Internet Gateway**

Automatically assign **IPv4 Public addresses**

© Digital Cloud Training | https://digitalcloud.training

Launch Instances and Test VPC

© Digital Cloud Training | https://digitalcloud.training

Amazon EBS and Instance Stores

© Digital Cloud Training | https://digitalcloud.training

Amazon Elastic Block Store (EBS)

The volume is automatically replicated **within the AZ**

© Digital Cloud Training | https://digitalcloud.training

Amazon EBS Volume Types

	General Purpose SSD		Provisioned IOPS SSD		
Volume type	gp3	gp2	io2 Block Express ‡	io2	io1
Durability	99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)	99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)	99.999% durability (0.001% annual failure rate)		99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)
Use cases	• Low-latency interactive apps • Development and test environments		Workloads that require sub-millisecond latency, and sustained IOPS performance or more than 64,000 IOPS or 1,000 MiB/s of throughput	• Workloads that require sustained IOPS performance or more than 16,000 IOPS • I/O-intensive database workloads	
Volume size	1 GiB - 16 TiB		4 GiB - 64 TiB	4 GiB - 16 TiB	
Max IOPS per volume (16 KiB I/O)	16,000		256,000	64,000 †	
Max throughput per volume	1,000 MiB/s	250 MiB/s *	4,000 MiB/s	1,000 MiB/s †	
Amazon EBS Multi-attach	Not supported		Not supported	Supported	
Boot volume	Supported				

© Digital Cloud Training | https://digitalcloud.training

Amazon EBS vs Instance Store

EBS volumes are **attached** over the **network**

EBS volumes offer persistent storage

Instance Store volumes are physically attached to the host and offer high - performance

Instance Store volumes are **ephemeral** (non-persistent)

© Digital Cloud Training | https://digitalcloud.training

Amazon EBS Snapshots

© Digital Cloud Training | https://digitalcloud.training

Create and Attach an EBS Volume

© Digital Cloud Training | https://digitalcloud.training

Amazon Elastic File System (EFS)

© Digital Cloud Training | https://digitalcloud.training

Amazon Elastic File System (EFS)

© Digital Cloud Training | https://digitalcloud.training

Create an Amazon EFS Filesystem

© Digital Cloud Training | https://digitalcloud.training

Mounting EFS File Systems

```
mkdir ~/efs-mount-point

# Install NFS client

sudo yum -y install nfs-utils

# Mount using the NFS client

sudo mount -t nfs4 -o nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2,noresvport fs-0187058e25d360ce2.efs.us-east-1.amazonaws.com:/ ~/efs-mount-point

# Install EFS utils

sudo yum install -y amazon-efs-utils

# Mount using the EFS mount helper:

sudo mount -t efs -o tls fs-0187058e25d360ce2.efs.us-east-1.amazonaws.com:/ ~/efs-mount-point
```

© Digital Cloud Training | https://digitalcloud.training

Amazon EC2 User Data and Metadata

© Digital Cloud Training | https://digitalcloud.training

Amazon EC2 User Data

AWS Management Console

The code is run when the instance starts for the **first time**

Batch and **PowerShell** scripts can be run on Windows

User data (i) ◉ As text ○ As file ☐ Input is already base64 encoded

```
#!/bin/bash
yum update -y
yum install -y httpd
systemctl start httpd
systemctl enable httpd
```

Limited to **16 KB**

EC2 Instance

EC2 Instance with a **web service** is launched

© Digital Cloud Training | https://digitalcloud.training

DigitalCloud TRAINING

Amazon EC2 Metadata

- Instance metadata is data about your EC2 instance
- Instance metadata is available at http://169.254.169.254/latest/meta-data
- Examples:

```
[ec2-user@ip-172-31-42-248 ~]$ curl http://169.254.169.254/latest/meta-data
ami-id
ami-launch-index
ami-manifest-path
block-device-mapping/
events/
hibernation/
hostname
identity-credentials/
instance-action
instance-id
instance-life-cycle
instance-type
local-hostname
local-ipv4
```

Amazon EC2 Metadata

- Examples ctd.:

```
[ec2-user@ip-172-31-42-248 ~]$ curl http://169.254.169.254/latest/meta-data/local-ipv4
172.31.42.248[ec2-user@ip-172-31-42-248 ~]$
```

```
[ec2-user@ip-172-31-42-248 ~]$ curl http://169.254.169.254/latest/meta-data/public-ipv4
3.26.54.18[ec2-user@ip-172-31-42-248 ~]$
```

© Digital Cloud Training | https://digitalcloud.training

Accessing Services – Access Keys and IAM Roles

© Digital Cloud Training | https://digitalcloud.training

Access Keys

© Digital Cloud Training | https://digitalcloud.training

Amazon EC2 Instance Profiles (IAM Roles for EC2)

© Digital Cloud Training | https://digitalcloud.training

Launch Auto Scaling Web Application on EC2

© Digital Cloud Training | https://digitalcloud.training

Types of Elastic Load Balancer (ELB)

© Digital Cloud Training | https://digitalcloud.training

Types of Elastic Load Balancer (ELB)

Application Load Balancer

- Operates at the request level
- Routes based on the content of the request (layer 7)
- Supports path-based routing, host-based routing, query string parameter-based routing, and source IP address-based routing
- Supports instances, IP addresses, Lambda functions and containers as targets

Network Load Balancer

- Operates at the connection level
- Routes connections based on IP protocol data (layer 4)
- Offers ultra high performance, low latency and TLS offloading at scale
- Can have a static IP / Elastic IP
- Supports UDP and static IP addresses as targets

© Digital Cloud Training | https://digitalcloud.training

Types of Elastic Load Balancer (ELB)

Old and **shouldn't** be in the exam anymore

Classic Load Balancer

- Old generation; not recommended for new applications
- Performs routing at Layer 4 and Layer 7
- Use for existing applications running in EC2-Classic

Gateway Load Balancer

- Used in front of virtual appliances such as firewalls, IDS/IPS, and deep packet inspection systems.
- Operates at Layer 3 – listens for all packets on all ports
- Forwards traffic to the TG specified in the listener rules
- Exchanges traffic with appliances using the GENEVE protocol on port 6081

New and **starting** to appear in the exam

© Digital Cloud Training | https://digitalcloud.training

ELB Features

Feature	ALB	NLB
OSI Layer	7	4
Target Type	IP, Instance, Lambda, ECS	IP, Instance
Protocol Listeners	HTTP, HTTPS, gRPC	TCP, UDP, TLS
PrivateLink support	No	(TCP, TLS)
Static IP address	No	Yes
HTTP header based routing	Yes	No
Source IP preservation	x-forwarded-for	Native
SSL termination	Load Balancer	Load Balancer or target

© Digital Cloud Training | https://digitalcloud.training

ELB Use Cases

Application Load Balancer

- Web applications with L7 routing (HTTP/HTTPS)
- Microservices architectures (e.g. Docker containers)
- Lambda targets

Network Load Balancer

- TCP and UDP based applications
- Ultra-low latency
- Static IP addresses
- VPC endpoint services

© Digital Cloud Training | https://digitalcloud.training

ELB Use Cases

Gateway Load Balancer

- Load balance virtual appliances such as:
  - Intrusion detection systems (IDS)
  - Intrusion prevention systems (IPS)
  - Next generation firewalls (NGFW)
  - Web application firewalls (WAF)
  - Distributed denial of service protection systems (DDoS)
- Integrate with Auto Scaling groups for elasticity
- Apply network monitoring and logging for analytics

© Digital Cloud Training | https://digitalcloud.training

Routing and Session Management

© Digital Cloud Training | https://digitalcloud.training

Application Load Balancer (ALB)

© Digital Cloud Training | https://digitalcloud.training

Network Load Balancer (NLB)

Network Load Balancer (NLB)

Requests are routed based on **IP protocol** data

NLBs listen on **TCP**, **TLS**, **UDP** or **TCP_UDP**

NLB **nodes** can have **elastic IPs** in each subnet

TG1

TG2

54.22.182.2

Subnet

Targets can be EC2 instances or IP addresses

54.239.28.85

Subnet

54.12.10.212

Subnet

Targets can be outside a VPC (e.g. on-premises)

New Tab

https://example.com

New Tab

https://example.com:8080

A **separate** listener on a **unique port** is required for routing

© Digital Cloud Training | https://digitalcloud.training

DigitalCloud TRAINING

Storing Session State

© Digital Cloud Training | https://digitalcloud.training

Sticky Sessions

Session data such as authentication details stored **locally**

Cookie is generated and client bound to instance for **cookie lifetime**

If an instance fails, session state is **lost** – use with **session state store** for more resiliency

Client is directed to another **instance**

© Digital Cloud Training | https://digitalcloud.training

Deploy an Application Load Balancer

© Digital Cloud Training | https://digitalcloud.training

Create ASG and ALB using the AWS CLI

© Digital Cloud Training | https://digitalcloud.training

SECTION 5

Amazon S3 and CloudFront

© Digital Cloud Training | https://digitalcloud.training

Amazon S3 Overview

© Digital Cloud Training | https://digitalcloud.training

Amazon Simple Storage Service (S3)

S3 Bucket

A **bucket** is a container for objects

Object

Object

Object

An **object** is a file you upload

You can store millions of **objects** in a **bucket**

Accessing objects in a bucket:

```
https://bucket.s3.aws-region.amazonaws.com/key
https://s3.aws-region.amazonaws.com/bucket/key
```

The **HTTP protocol** is used with a **REST API** (e.g. GET, PUT, POST, SELECT, DELETE)

© Digital Cloud Training | https://digitalcloud.training

Amazon Simple Storage Service (S3)

- You can store any type of file in S3
- Files can be anywhere from 0 bytes to 5 TB
- There is unlimited storage available
- S3 is a universal namespace so **bucket names** must be **unique globally**
- However, you create your buckets within a **REGION**
- It is a best practice to create buckets in regions that are physically closest to your users to reduce latency
- There is no hierarchy for objects within a bucket
- Delivers strong read-after-write consistency

© Digital Cloud Training | https://digitalcloud.training

Buckets, Folders, and Objects

© Digital Cloud Training | https://digitalcloud.training

Buckets, Folders, and Objects

- Folders **can** be created within folders
- Buckets **cannot** be created within other buckets
- An objects consists of:
  - Key (the name of the object)
  - Version ID
  - Value (actual data)
  - Metadata
  - Subresources
  - Access control information

© Digital Cloud Training | https://digitalcloud.training

Accessing Amazon S3

© Digital Cloud Training | https://digitalcloud.training

Amazon S3 Storage Classes

© Digital Cloud Training | https://digitalcloud.training

S3 Storage Classes

	S3 Standard	S3 Intelligent Tiering	S3 Standard-IA	S3 One Zone-IA	S3 Glacier	S3 Glacier Deep Archive
Designed for durability	99.999999999%	99.999999999%	99.999999999%	99.999999999%	99.999999999%	99.999999999%
Designed for availability	99.99%	99.9%	99.9%	99.5%	99.99%	99.99%
Availability SLA	99.9%	99%	99%	99%	99.9%	99.9%
Availability Zones	≥3	≥3	≥3	1	≥3	≥3
Minimum capacity charge per object	N/A	N/A	128KB	128KB	40KB	40KB
Minimum storage duration charge	N/A	30 days	30 days	30 days	90 days	180 days
Retrieval fee	N/A	N/A	Per GB retrieved	Per GB retrieved	Per GB retrieved	Per GB retrieved
First byte latency	milliseconds	milliseconds	milliseconds	milliseconds	select minutes or hours	select hours
Storage type	Object	Object	Object	Object	Object	Object
Lifecycle transitions	Yes	Yes	Yes	Yes	Yes	Yes

© Digital Cloud Training | https://digitalcloud.training

S3 Storage Class Use Cases

- S3 Standard
- S3 Intelligent Tiering
- S3 Standard-IA
- S3 One Zone-IA
- S3 Glacier
- S3 Glacier Deep Archive

Require long-term storage for production data with low cost and immediate access for occasional requests

Copy of backup data required in a separate Region with minimal access latency

Frequently accessed data; ad-hoc needs; short-term requirements (<30 days)

Lowest cost required for long-term archival of data for compliance purposes

© Digital Cloud Training | https://digitalcloud.training

Create Amazon S3 Bucket

© Digital Cloud Training | https://digitalcloud.training

IAM Policies, Bucket Policies and ACLs

© Digital Cloud Training | https://digitalcloud.training

IAM Policies

- IAM Policies are identity-based policies
- Specify what actions are allowed on what AWS resources

© Digital Cloud Training | https://digitalcloud.training

Bucket Policies

- Bucket Policies are resource-based policies
- Can only be attached to Amazon S3 buckets
- Also use the AWS access policy language

```
{
    "Version": "2012-10-17",
    "Id": "Policy1561964929358",
    "Statement":[
    {
        "Sid": "Stmt1561964454052",
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::515148227241:user/Paul"
        },
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::dctcompany"
    }
    ]
}
```

© Digital Cloud Training | https://digitalcloud.training

S3 Access Control Lists (ACLs)

- Legacy access control mechanism that predates IAM
- AWS generally recommends using **S3 bucket policies** or **IAM policies** rather than ACLs
- Can be attached to a **bucket** or directly to an **object**
- Limited options for grantees and permissions

© Digital Cloud Training | https://digitalcloud.training

When to use each access control mechanism

- **Use IAM policies if:**
  - You need to control access to AWS services other than S3
  - You have numerous S3 buckets each with different permissions requirements (IAM policies will be easier to manage)
  - You prefer to keep access control policies in the IAM environment
- **Use S3 bucket policies if:**
  - You want a simple way to grant cross-account access to your S3 environment, without using IAM roles
  - Your IAM policies are reaching the size limits
  - You prefer to keep access control policies in the S3 environment

© Digital Cloud Training | https://digitalcloud.training

Authorization Process

© Digital Cloud Training | https://digitalcloud.training

Access Control Lists (ACLs)

© Digital Cloud Training | https://digitalcloud.training

Bucket and User Policy Practice

© Digital Cloud Training | https://digitalcloud.training

MFA with Amazon S3

© Digital Cloud Training | https://digitalcloud.training

S3 Multi-Factor Authentication Delete (MFA Delete)

- Adds MFA requirement for bucket owners to the following operations:
  - Changing the versioning state of a bucket
  - Permanently deleting an object version
- The x-amz-mfa request header must be included in the above requests
- The second factor is a token generated by a hardware device or software program
- Requires **versioning** to be enabled on the bucket

© Digital Cloud Training | https://digitalcloud.training

S3 Multi-Factor Authentication Delete (MFA Delete)

- **Versioning** can be enabled by:
  - Bucket owners (root account)
  - AWS account that created the bucket
  - Authorized IAM users
- **MFA delete** can be enabled by:
  - Bucket owner (root account)

© Digital Cloud Training | https://digitalcloud.training

MFA-Protected API Access

- Used to enforce another authentication factor (MFA code) when accessing AWS resources (not just S3)
- Enforced using the aws:MultiFactorAuthAge key in a bucket policy:

```
{
    "Version": "2012-10-17",
    "Id": "123",
    "Statement": [
      {
        "Sid": "",
        "Effect": "Deny",
        "Principal": "*",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::examplebucket/securedocuments/*",
        "Condition": { "Null": { "aws:MultiFactorAuthAge": true }}
      }
    ]
}
```

Denies any API operation that is not authenticated using MFA

© Digital Cloud Training | https://digitalcloud.training

Amazon S3 Encryption

© Digital Cloud Training | https://digitalcloud.training

Amazon S3 Encryption

Server-side encryption with S3 managed keys (SSE-S3)

- S3 managed keys
- Unique object keys
- Master key
- AES 256

Server-side encryption with AWS KMS managed keys (SSE-KMS)

- KMS managed keys
- Can be AWS managed keys
- Or customer managed KMS keys

© Digital Cloud Training | https://digitalcloud.training

DigitalCloud TRAINING

Amazon S3 Encryption

Server-side encryption with client provided keys (SSE-C)

Encryption / decryption

User

- Client managed keys
- Not stored on AWS

Client-side encryption

Encryption / decryption

User

- Client managed keys
- Not stored on AWS
- Or you can use a KMS Key

© Digital Cloud Training | https://digitalcloud.training

Amazon S3 Default Encryption

- All Amazon S3 buckets have encryption configured by default
- All new object uploads to Amazon S3 are automatically encrypted
- There is no additional cost and no impact on performance
- Objects are automatically encrypted by using server-side encryption with Amazon S3 managed keys (SSE-S3)
- To encrypt existing unencrypted Amazon S3 objects, you can use Amazon S3 Batch Operations
- You can also encrypt existing objects by using the CopyObject API operation or the copy-object AWS CLI command

© Digital Cloud Training | https://digitalcloud.training

Enforce Encryption with Bucket Policy

```
{
    "Version": "2012-10-17",
    "Id": "PutObjPolicy",
    "Statement": [
        {
            "Sid": "DenyIncorrectEncryptionHeader",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::<bucket_name>/*",
            "Condition": {
                "StringNotEquals": {
                    "s3:x-amz-server-side-encryption": "aws:kms"
                    }
                }
        },
        {
            "Sid": "DenyUnEncryptedObjectUploads",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::<bucket_name>/*",
            "Condition": {
                "Null": {
                    "s3:x-amz-server-side-encryption": true
                    }
                }
        }
    ]
}
```

Enforces encryption using **SSE-KMS**

Example PUT request

```
PUT /example-object HTTP/1.1
Host: myBucket.s3.amazonaws.com
Date: Wed, 8 Jun 2016 17:50:00 GMT
Authorization: authorization string
Content-Type: text/plain
Content-Length: 11434
x-amz-meta-author: Janet
Expect: 100-continue
x-amz-server-side-encryption: aws:kms
[11434 bytes of object data]
```

© Digital Cloud Training | https://digitalcloud.training

Enforce Encryption

© Digital Cloud Training | https://digitalcloud.training

S3 Event Notifications

© Digital Cloud Training | https://digitalcloud.training

S3 Presigned URLs

© Digital Cloud Training | https://digitalcloud.training

S3 Presigned URLs

AWS S3 CLI command to generate a presigned URL

```
aws s3 presign s3://dct-data-bucket/cool_image.jpeg
```

```
https://dct-data-bucket.s3.ap-southeast-2.amazonaws.com/cool_image.jpeg?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIA3KSVPHP6MAHNW5YH%2F20200909%2Fap-southeast-2%2Fs3%2Faws4_request&X-Amz-Date=20200909T053538Z&X-Amz-Expires=3600&X-Amz-SignedHeaders=host&X-Amz-Signature=8b74653beee371da07a73dfdb4ff6883742383afa528aecd5c95c326c97764db
```

This is the response; the URL expires after 1 hour

© Digital Cloud Training | https://digitalcloud.training

Server Access Logging

© Digital Cloud Training | https://digitalcloud.training

Server Access Logging

- Provides detailed records for the requests that are made to a bucket
- Details include the requester, bucket name, request time, request action, response status, and error code (if applicable)
- Disabled by default
- Only pay for the storage space used
- Must configure a separate bucket as the destination (can specify a prefix)
- Must grant write permissions to the Amazon S3 Log Delivery group on destination bucket

Cross-Origin Resource Sharing (CORS)

© Digital Cloud Training | https://digitalcloud.training

CORS with Amazon S3

- Allows requests from an origin to another origin
- Origin is defined by DNS name, protocol, and port

© Digital Cloud Training | https://digitalcloud.training

CORS with Amazon S3

- Enabled through setting:
  - Access-Control-Allow-Origin
  - Access-Control-Allow-Methods
  - Access-Control-Allow-Headers
- These settings are defined using rules
- Rules are added using JSON files in S3

© Digital Cloud Training | https://digitalcloud.training

Example CORS Rule

```
[
    {
        "AllowedHeaders": [
            "*"
        ],
        "AllowedMethods": [
            "PUT",
            "POST",
            "DELETE"
        ],
        "AllowedOrigins": [
            "http://www.mycompany.com"
        ],
        "ExposeHeaders": []
    }
]
```

© Digital Cloud Training | https://digitalcloud.training

Amazon S3
Performance Optimization

© Digital Cloud Training | https://digitalcloud.training

Design Patterns for Optimizing S3 Performance

- S3 Automatically scales to high request rates with at least (per prefix):
  - 3,500 PUT/COPY/POST/DELETE requests/second
  - 5,500 GET/HEAD requests/second
- Can increase read and write performance by using parallelization across **multiple prefixes**
- To increase uploads over long distances, use Amazon S3 Transfer Acceleration
- Byte-Range fetches use the Range HTTP header to transfer only specified byte-range from an object

© Digital Cloud Training | https://digitalcloud.training

Design Patterns for Optimizing S3 Performance

- Combine S3 and EC2 in the same AWS Region
- Use the latest version of the AWS SDKs
- Use caching services to cache the latest content:
  - Amazon CloudFront (CDN)
  - Amazon ElastiCache (in-memory cache)
- Horizontally scale requests across S3 endpoints
- More info here:
- https://docs.aws.amazon.com/AmazonS3/latest/userguide/optimizing-performance.html

© Digital Cloud Training | https://digitalcloud.training

Amazon CloudFront Origins and Distributions

© Digital Cloud Training | https://digitalcloud.training

Amazon CloudFront

© Digital Cloud Training | https://digitalcloud.training

CloudFront Origins and Distributions

CloudFront Distribution

Name: **d1schtd9zdwrm1.cloudfront.net**

S3 Origin

S3 static websites can also be origins

Behaviors

Path Pattern
Viewer Protocol Policy
Cache Policy
Origin Request Policy

Custom Origin

RTMP distributions were discontinued so only **web distributions** are currently available

CloudFront Web Distribution:

- Speed up distribution of static and dynamic content, for example, .html, .css, .php, and graphics files
- Distribute media files using HTTP or HTTPS
- Add, update, or delete objects, and submit data from web forms
- Use live streaming to stream an event in real time

© Digital Cloud Training | https://digitalcloud.training

CloudFront Signed URLs and OAI/OAC

© Digital Cloud Training | https://digitalcloud.training

CloudFront Signed URLs

- Signed URLs provide more control over access to content
- Can specify beginning and expiration date and time, IP addresses

Signed URLs should be used for **individual files** and clients that don't support **cookies**

© Digital Cloud Training | https://digitalcloud.training

CloudFront Signed Cookies

- Similar to Signed URLs
- Use signed cookies when you don’t want to change URLs
- Can also be used when you want to provide access to **multiple restricted files** (Signed URLs are for individual files)

© Digital Cloud Training | https://digitalcloud.training

CloudFront Origin Access Identity (OAI)

```
{
    "Version": "2008-10-17",
    "Id": "PolicyForCloudFrontPrivateContent",
    "Statement": [
        {
            "Sid": "1",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::cloudfront:user/CloudFront Origin Access Identity E11A2JL2H306JJ"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::mybucket/*"
        }
    ]
}
```

The policy restricts access to the **OAI**

HTTP GET **https://d1schtd9zdwrm1.cloudfront.net**

Access is blocked by **bucket policy**

GET **https://mybucket.s3.amazonaws.com/beach.jpg**

© Digital Cloud Training | https://digitalcloud.training

CloudFront Origin Access Control (OAC)

- Like a OAI but supports additional use cases
- AWS recommend using the OAC instead of an OAI
- Requires an S3 bucket policy that allows the CloudFront service principal

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Sid": "AllowCloudFrontServicePrincipalReadOnly",
        "Effect": "Allow",
        "Principal": {
            "Service": "cloudfront.amazonaws.com"
        },
        "Action": "s3:GetObject",
        "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
        "Condition": {
            "StringEquals": {
                "AWS:SourceArn": "arn:aws:cloudfront::111122223333:distribution/EDFDVBD6EXAMPLE"
            }
        }
    }
}
```

© Digital Cloud Training | https://digitalcloud.training

CloudFront with S3 Static Website and OAI

© Digital Cloud Training | https://digitalcloud.training

Amazon Route 53 DNS

© Digital Cloud Training | https://digitalcloud.training

Amazon Route 53

Amazon Route 53

Domain Registration

.net
.com
.org

Hosted zone

example.com
dctlabs.com

Health Checks

EC2 Instances

Traffic Flow

© Digital Cloud Training | https://digitalcloud.training

DigitalCloud TRAINING

DNS Resolution with Route 53

© Digital Cloud Training | https://digitalcloud.training

Route 53 DNS Record Types

Routing Policy	What it does
Simple	Simple DNS response providing the IP address associated with a name
Failover	If primary is down (based on health checks), routes to secondary destination
Geolocation	Uses geographic location you're in (e.g. Europe) to route you to the closest region
Geoproximity	Routes you to the closest region within a geographic area
Latency	Directs you based on the lowest latency route to resources
Multivalue answer	Returns several IP addresses and functions as a basic load balancer
Weighted	Uses the relative weights assigned to resources to determine which to route to

© Digital Cloud Training | https://digitalcloud.training

SECTION 6

Infrastructure as Code and PaaS

© Digital Cloud Training | https://digitalcloud.training

AWS CloudFormation

© Digital Cloud Training | https://digitalcloud.training

AWS CloudFormation

Infrastructure patterns are defined in a **template** file using **code**

CloudFormation **builds** your infrastructure according to the **template**

AWS CloudFormation

```
"AWSTemplateFormatVersion" : "2010-09-09",

  "Description" : "AWS CloudFormation Sample Template WordPress_Multi_AZ: WordPress is web

  "Parameters" : {
    "VpcId" : {
      "Type" : "AWS::EC2::VPC::Id",
      "Description" : "VpcId of your existing Virtual Private Cloud (VPC)",
      "ConstraintDescription" : "must be the VPC Id of an existing Virtual Private Cloud."
    },

    "Subnets" : {
      "Type" : "List<AWS::EC2::Subnet::Id>",
      "Description" : "The list of SubnetIds in your Virtual Private Cloud (VPC)",
      "ConstraintDescription" : "must be a list of at least two existing subnets associated
    },
```

© Digital Cloud Training | https://digitalcloud.training

AWS CloudFormation - Benefits

- Infrastructure is provisioned consistently, with fewer mistakes (human error)
- Less time and effort than configuring resources manually
- You can use version control and peer review for your CloudFormation templates
- Free to use (you're only charged for the resources provisioned)
- Can be used to manage updates and dependencies
- Can be used to rollback and delete the entire stack as well

© Digital Cloud Training | https://digitalcloud.training

AWS CloudFormation

Component	Description
Templates	The JSON or YAML text file that contains the instructions for building out the AWS environment
Stacks	The entire environment described by the template and created, updated, and deleted as a single unit
StackSets	AWS CloudFormation StackSets extends the functionality of stacks by enabling you to create, update, or delete stacks across multiple accounts and regions with a single operation
Change Sets	A summary of proposed changes to your stack that will allow you to see how those changes might impact your existing resources before implementing them

© Digital Cloud Training | https://digitalcloud.training

Stacks, Stack Sets, and Nested Stacks

© Digital Cloud Training | https://digitalcloud.training

CloudFormation Stacks

- Deployed resources based on **templates**
- You create, update and delete ***stacks*** using templates
- Stacks are deployed through the Management Console, CLI or APIs
- Stack creation errors:
  - Automatic rollback on error is enabled by default
  - You will be charged for resources provisioned even if there is an error

© Digital Cloud Training | https://digitalcloud.training

CloudFormation StackSets

- Create, update, or delete stacks across **multiple accounts** and **Regions** with a single operation
- An administrator account is the AWS account in which you create StackSets
- A stack set is managed by signing into the AWS administrator account in which it was created
- A target account is the account into which you create, update, or delete one or more stacks in your stack set

© Digital Cloud Training | https://digitalcloud.training

CloudFormation Nested Stacks

- Nested stacks allow re-use of CloudFormation code for common use cases
- For example, a standard configuration for a load balancer, web server, application server etc.
- Create a standard template for each common use case and reference it from within your CloudFormation template

© Digital Cloud Training | https://digitalcloud.training

Deploy Infrastructure as Code

© Digital Cloud Training | https://digitalcloud.training

CloudFormation Template Deep Dive

© Digital Cloud Training | https://digitalcloud.training

CloudFormation Templates

- A template is a **YAML** or **JSON** template used to describe the end-state of the infrastructure you are provisioning or changing
- After creating the template, you upload it to CloudFormation directly or using Amazon S3
- CloudFormation reads the template and makes the API calls on your behalf
- The resulting resources are called a "Stack"
- **Logical IDs** – used to reference resources within the template
- **Physical IDs** – identify resources outside of AWS CloudFormation templates, but only after the resources have been created

© Digital Cloud Training | https://digitalcloud.training

Intrinsic Functions

- AWS CloudFormation provides several built-in functions that help you manage your stacks
- Use intrinsic functions in your templates to assign values to properties that are not available until runtime
- The full list can be found here:
- https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/intrinsic-function-reference.html

© Digital Cloud Training | https://digitalcloud.training

Intrinsic Functions - Ref

- The intrinsic function Ref returns the value of the specified parameter or resource
- When you specify a parameter's logical name, it returns the value of the parameter
- When you specify a resource's logical name, it returns a value that you can typically use to refer to that resource, such as a physical ID

© Digital Cloud Training | https://digitalcloud.training

Intrinsic Functions - Ref

- The following resource declaration for an Elastic IP address needs the instance ID of an EC2 instance and uses the Ref function to specify the instance ID of the MyEC2Instance resource:

```yaml
MyEIP:
  Type: "AWS::EC2::EIP"
  Properties:
    InstanceId: !Ref MyEC2Instance
```

© Digital Cloud Training | https://digitalcloud.training

Intrinsic Functions - Fn::GetAtt

- The Fn::GetAtt intrinsic function returns the value of an attribute from a resource in the template
- Full syntax (YAML):
  - *Fn::GetAtt: [logicalNameOfResource, attributeName]*
- Short form (YAML):
  - *!GetAtt logicalNameOfResource.attributeName*

© Digital Cloud Training | https://digitalcloud.training

Intrinsic Functions - Fn::GetAtt

The example template returns the following for the ELB named MyELB:

- SourceSecurityGroup.OwnerAlias
- SourceSecurityGroup.GroupName

```yaml
AWSTemplateFormatVersion: 2010-09-09
Resources:
  myELB:
    Type: AWS::ElasticLoadBalancing::LoadBalancer
    Properties:
      AvailabilityZones:
        - eu-west-1a
      Listeners:
        - LoadBalancerPort: '80'
          InstancePort: '80'
          Protocol: HTTP
  myELBIngressGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: ELB ingress group
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: '80'
          ToPort: '80'
          SourceSecurityGroupOwnerId: !GetAtt myELB.SourceSecurityGroup.OwnerAlias
          SourceSecurityGroupName: !GetAtt myELB.SourceSecurityGroup.GroupName
```

© Digital Cloud Training | https://digitalcloud.training

Intrinsic Functions - Fn::FindInMap

- The intrinsic function Fn::FindInMap returns the value corresponding to keys in a two-level map that is declared in the Mappings section
- Full syntax (YAML):
  - *Fn::FindInMap: [MapName, TopLevelKey, SecondLevelKey]*
- Short form (YAML):
  - *!FindInMap [MapName, TopLevelKey, SecondLevelKey]*

© Digital Cloud Training | https://digitalcloud.training

Intrinsic Functions - Fn::FindInMap

- The example shows how to use Fn::FindInMap for a template with a Mappings section that contains a single map (RegionMap) that associates AMIs with AWS regions:

```yaml
Mappings:
  RegionMap:
    us-east-1:
      HVM64: "ami-0ff8a91507f77f867"
      HVMG2: "ami-0a584ac55a7631c0c"
    us-west-1:
      HVM64: "ami-0bdb828fd58c52235"
      HVMG2: "ami-066ee5fd4a9ef77f1"
Resources:
  myEC2Instance:
    Type: "AWS::EC2::Instance"
    Properties:
      ImageId: !FindInMap
        - RegionMap
        - !Ref 'AWS::Region'
        - HVM64
      InstanceType: m1.small
```

© Digital Cloud Training | https://digitalcloud.training

Template Sections - Resources

- The required Resources section declares the AWS resources that you want to include in the stack, such as an Amazon EC2 instance or an Amazon S3 bucket
- This is a mandatory section
- Resources are declared and can reference each other

```yaml
Resources:
 MyEC2Instance:
   Type: "AWS::EC2::Instance"
   Properties:
     ImageId: "ami-0ff8a91507f77f867"
```

© Digital Cloud Training | https://digitalcloud.training

Template Sections - Parameters

- Use the *optional* Parameters section to customize your templates
- Parameters enable you to input custom values to your template each time you create or update a stack
- Useful for template reuse

```yaml
Parameters:
  InstanceTypeParameter:
    Type: String
    Default: t2.micro
    AllowedValues:
      - t2.micro
      - m1.small
      - m1.large
    Description: Enter t2.micro, m1.small, or m1.large. Default is t2.micro.
```

© Digital Cloud Training | https://digitalcloud.training

Template Sections - Mappings

- The *optional* Mappings section matches a key to a corresponding set of named values

```yaml
RegionMap:
    us-east-1:
      HVM64: ami-0ff8a91507f77f867
      HVMG2: ami-0a584ac55a7631c0c
    us-west-1:
      HVM64: ami-0bdb828fd58c52235
      HVMG2: ami-066ee5fd4a9ef77f1
```

Exam tip: with mappings you can set values based on a region. You can create a mapping that uses the region name as a key and contains the values you want to specify for each specific region

© Digital Cloud Training | https://digitalcloud.training

Template Sections - Outputs

- The *optional* Outputs section declares output values that you can import into other stacks (to create cross-stack references), return in response (to describe stack calls), or view on the AWS CloudFormation console
- In the following example YAML code, the output named StackVPC returns the ID of a VPC, and then exports the value for cross-stack referencing with the name VPCID appended to the stack's name

```yaml
Outputs:
  StackVPC:
    Description: The ID of the VPC
    Value: !Ref MyVPC
    Export:
      Name: !Sub "${AWS::StackName}-VPCID"
```

© Digital Cloud Training | https://digitalcloud.training

Template Sections - Conditions

- The *optional* Conditions section contains statements that define the circumstances under which entities are created or configured
- In the sample YAML code below, resources are created only if the EnvType parameter is equal to prod:

```yaml
Conditions:
  CreateProdResources: !Equals [ !Ref EnvType, prod ]
```

© Digital Cloud Training | https://digitalcloud.training

Template Sections - Transform

- The *optional* Transform section specifies one or more macros that AWS CloudFormation uses to process your template
- The transform section can be used to reference additional code stored in S3, such as Lambda code or reusable snippets of CloudFormation code

© Digital Cloud Training | https://digitalcloud.training

Template Sections - Transform

- The AWS::Serverless transform specifies the version of the AWS Serverless Application Model (AWS SAM) to use
- This model defines the AWS SAM syntax that you can use and how AWS CloudFormation processes it
- The AWS::Include transform works with template snippets that are stored separately from the main AWS CloudFormation template
- In the following example, the template uses AWS SAM syntax to simplify the declaration of a Lambda function and its execution role:

```yaml
Transform: AWS::Serverless-2016-10-31
Resources:
  MyServerlessFunctionLogicalID:
    Type: AWS::Serverless::Function
    Properties:
      Handler: index.handler
      Runtime: nodejs8.10
      CodeUri: 's3://testBucket/mySourceCode.zip'
```

© Digital Cloud Training | https://digitalcloud.training

Complex VPC Stack

© Digital Cloud Training | https://digitalcloud.training

AWS Elastic Beanstalk

© Digital Cloud Training | https://digitalcloud.training

AWS Elastic Beanstalk

© Digital Cloud Training | https://digitalcloud.training

AWS Elastic Beanstalk

- Supports Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker web applications
- Supports the following languages and development stacks:
  - Apache Tomcat for Java applications
  - Apache HTTP Server for PHP applications
  - Apache HTTP Server for Python applications
  - Nginx or Apache HTTP Server for Node.js applications
  - Passenger or Puma for Ruby applications
  - Microsoft IIS 7.5, 8.0, and 8.5 for .NET applications
  - Java SE
  - Docker
  - Go

© Digital Cloud Training | https://digitalcloud.training

AWS Elastic Beanstalk

There are several **layers**

Applications:

- Contain environments, environment configurations, and application versions
- You can have multiple application versions held within an application

APPLICATION

© Digital Cloud Training | https://digitalcloud.training

AWS Elastic Beanstalk

Application version

- A specific reference to a section of deployable code
- The application version will point typically to an Amazon S3 bucket containing the code

© Digital Cloud Training | https://digitalcloud.training

AWS Elastic Beanstalk

Environments:

- An application version that has been deployed on AWS resources
- The resources are configured and provisioned by AWS Elastic Beanstalk
- The environment is comprised of all the resources created by Elastic Beanstalk and not just an EC2 instance with your uploaded code

© Digital Cloud Training | https://digitalcloud.training

Web Servers and Workers

- **Web servers** are standard applications that listen for and then process HTTP requests, typically over port 80
- **Workers** are specialized applications that have a background processing task that listens for messages on an Amazon SQS queue
- **Workers** should be used for long-running tasks

© Digital Cloud Training | https://digitalcloud.training

AWS Elastic Beanstalk

© Digital Cloud Training | https://digitalcloud.training

Creating and Updating Environments

© Digital Cloud Training | https://digitalcloud.training

Elastic Beanstalk Deployment Options

© Digital Cloud Training | https://digitalcloud.training

Elastic Beanstalk Deployment Options

All at once

- Deploys the new version to all instances simultaneously

Rolling

- Update a batch of instances, and then move onto the next batch once the first batch is healthy

Rolling with additional batch

- Like Rolling but launches new instances in a batch ensuring that there is full availability

Immutable

- Launches new instances in a new ASG and deploys the version update to these instances before swapping traffic to these instances once healthy

Blue/green

- Create a new "stage" environment and deploy updates there

© Digital Cloud Training | https://digitalcloud.training

All at Once Update

All at Once Update

- Deploys the new version to all instances simultaneously
- All of your instances are out of service while the deployment takes place
- Fastest deployment
- Good for quick iterations in development environment
- You will experience an outage while the deployment is taking place - not ideal for mission-critical systems
- If the update fails, you need to roll back the changes by re-deploying the original version to all of your instances
- No additional cost

© Digital Cloud Training | https://digitalcloud.training

Rolling Update

Rolling Update

- Update a few instances at a time (batch), and then move onto the next batch once the first batch is healthy
- Downtime affects 1 batch at a time
- Application is running both versions simultaneously
- Each batch of instances is taken out of service while the deployment takes place
- Your environment capacity will be reduced by the number of instances in a batch while the deployment takes place
- Not ideal for performance-sensitive systems
- If the update fails, you need to perform an additional rolling update to roll back the changes
- No additional cost
- Long deployment time

© Digital Cloud Training | https://digitalcloud.training

Rolling with Additional Batch Update

© Digital Cloud Training | https://digitalcloud.training

Rolling with Additional Batch Update

- Like Rolling but launches new instances in a batch ensuring that there is full availability
- Application is running at capacity
- Can set the batch size
- Application is running both versions simultaneously
- Small additional cost
- Additional batch is removed at the end of the deployment
- Longer deployment
- Good for production environments

© Digital Cloud Training | https://digitalcloud.training

Immutable Update

© Digital Cloud Training | https://digitalcloud.training

Immutable Update

- Launches new instances in a new ASG and deploys the version update to these instances before swapping traffic to these instances once healthy
- Zero downtime
- New code is deployed to new instances using an ASG
- High cost as double the number of instances running during updates
- Longest deployment
- Quick rollback in case of failures
- Great for production environments

© Digital Cloud Training | https://digitalcloud.training

Blue/Green Update

© Digital Cloud Training | https://digitalcloud.training

Blue/Green Update

- This is not a feature within Elastic Beanstalk
- You create a new "staging" environment and deploy updates there
- The new environment (green) can be validated independently, and you can roll back if there are issues
- Route 53 can be setup using weighted policies to redirect a a percentage of traffic to the staging environment
- Using Elastic Beanstalk, you can "swap URLs" when done with the environment test
- Zero downtime

© Digital Cloud Training | https://digitalcloud.training

Update App and Maintain Availability

© Digital Cloud Training | https://digitalcloud.training

Advanced Configuration and SSL/TLS

© Digital Cloud Training | https://digitalcloud.training

Configuration Files - .ebextensions

- You can add AWS Elastic Beanstalk configuration files (.ebextensions) to your web application's source code to configure your environment and customize the AWS resources that it contains
- Configuration files are YAML or JSON-formatted documents with a .config file extension that you place in a folder named .ebextensions and deploy in your application source bundle

```yaml
option_settings:
  aws:elasticbeanstalk:environment:
    LoadBalancerType: network
```

© Digital Cloud Training | https://digitalcloud.training

Configuration Files - .ebextensions

- The option_settings section of a configuration file defines values for configuration options
- The Resources section lets you further customize the resources in your application's environment and define additional AWS resources beyond the functionality provided by configuration options
- Additional sections of a configuration file let you configure the EC2 instances that are launched in your environment
- These include packages, sources, files, users, groups, commands, container_commands, and services

© Digital Cloud Training | https://digitalcloud.training

Using HTTPS with Elastic Beanstalk

- SSL/TLS certificates can be assigned to an environment's Elastic Load Balancer
- Can use AWS Certificate Manager (ACM)
- The connections between clients and the load balancer are secured
- Backend connections between the load balancer and EC2 instances are not secured
- You can configure the certificate through the console or through .ebextensions:

```
option_settings:
  aws:elbv2:listener:443:
    ListenerEnabled: 'true'
    Protocol: HTTPS
    SSLCertificateArns: arnXXX
```

© Digital Cloud Training | https://digitalcloud.training

Using HTTPS with Elastic Beanstalk

- For end-to-end encryption you can encrypt the backend connections as well
- Can use a self-signed certificate on the EC2 instances
- Configuration can be made using .ebextensions (load balancer dependent)
- For example, using the .ebextensions/https-reencrypt-alb.config configuration file

```
option_settings:
  aws:elbv2:listener:443:
    DefaultProcess: https
    ListenerEnabled: 'true'
    Protocol: HTTPS
  aws:elasticbeanstalk:environment:process:https:
    Port: '443'
    Protocol: HTTPS
```

© Digital Cloud Training | https://digitalcloud.training

HTTP to HTTPS Redirection

To configure HTTP to HTTPS redirection, do one of the following:

- **Configure the instance web servers** - configure EC2 web servers to respond to HTTP traffic with an HTTP redirection response status (platform-dependent config)
- **Configure the load balancer** – configure the ALB to send redirection responses to HTTP traffic

© Digital Cloud Training | https://digitalcloud.training

SECTION 7

AWS Lambda and AWS SAM

© Digital Cloud Training | https://digitalcloud.training

Serverless Services and Event-Driven Architecture

© Digital Cloud Training | https://digitalcloud.training

Serverless Services and Event-Driven Architecture

© Digital Cloud Training | https://digitalcloud.training

Serverless Services

- With serverless there are **no instances** to manage
- You don’t need to provision hardware
- There is no management of operating systems or software
- Capacity provisioning and patching is handled automatically
- Provides automatic scaling and high availability
- Can be very cheap!

© Digital Cloud Training | https://digitalcloud.training

AWS Lambda

© Digital Cloud Training | https://digitalcloud.training

AWS Lambda

© Digital Cloud Training | https://digitalcloud.training

AWS Lambda

- AWS Lambda executes code only when needed and scales automatically
- You pay only for the compute time you consume (you pay nothing when your code is not running)
- Benefits of AWS Lambda:
  - No servers to manage
  - Continuous scaling
  - Millisecond billing
  - Integrates with almost all other AWS services

© Digital Cloud Training | https://digitalcloud.training

AWS Lambda

- Primary use cases for AWS Lambda:
  - Data processing
  - Real-time file processing
  - Real-time stream processing
  - Build serverless backends for web, mobile, IOT, and 3rd party API requests

© Digital Cloud Training | https://digitalcloud.training

Lambda Function Invocations

Synchronous:

- CLI, SDK, API Gateway
- Wait for the function to process the event and return a response
- Error handling happens client side (retries, exponential backoff etc.)

Asynchronous:

- S3, SNS, CloudWatch Events etc.
- Event is queued for processing and a response is returned immediately
- Lambda retries up to 3 times
- Processing must be idempotent (due to retries)

Event source mapping:

- SQS, Kinesis Data Streams, DynamoDB Streams
- Lambda does the polling (polls the source)
- Records are processed in order (except for SQS standard)

SQS can also **trigger** Lambda

DigitalCloud TRAINING

Invoking Lambda Functions

© Digital Cloud Training | https://digitalcloud.training

Invoking Lambda Functions

- Lambda functions can be invoked directly through:
  - The Lambda console
  - A function URL HTTP(S) endpoint
  - The Lambda API
  - An AWS SDK
  - The AWS CLI
  - AWS Toolkits
- Lambda can be invoked by other AWS services
- Lambda can also be invoked when reading from a stream or queue
- Functions are invoked **synchronously** or **asynchronously**

© Digital Cloud Training | https://digitalcloud.training

Synchronous and Asynchronous Invocation

- With **synchronous** invocation, you wait for the function to process the event and return a response
- With **asynchronous** invocation, Lambda queues the event for processing and returns a response immediately
- For asynchronous invocation, Lambda handles retries and can send invocation records to a destination
- A trigger can be configured to invoke a function in response to lifecycle events, external requests, or on a schedule
- To process items from a stream or queue, you can create an **event source mapping**

© Digital Cloud Training | https://digitalcloud.training

Synchronous Invocation

- With **synchronous** invocation, you wait for the function to process the event and return a response
- Lambda runs the function and waits for a response
- When the function execution ends, Lambda returns the response from the function's code
- To invoke a function synchronously with the AWS CLI, use the invoke command

```
$ aws lambda invoke --function-name my-function --payload '{ "key": "value" }' response.json
{
"ExecutedVersion": "$LATEST",
"StatusCode": 200
}
```

© Digital Cloud Training | https://digitalcloud.training

Synchronous Invocation

- The payload is a string that contains an event in JSON format (base64 encoded)
- The name of the file where the AWS CLI writes the response from the function is response.json
- To get logs for an invocation from the command line, use the --log-type option. The response includes a LogResult field that contains up to 4 KB of base64-encoded logs from the invocation

```
$ aws lambda invoke --function-name my-function out --log-type Tail
{
    "StatusCode": 200,
    "LogResult": "U1RBUlQgUmVxdWVzdElkOiA4N2QwNDRiOC1mMTU0LTExZTgtOGNkYS0yOTc0YzVlNGZiMjEgVmVyc2lvb...",
    "ExecutedVersion": "$LATEST"
}
```

© Digital Cloud Training | https://digitalcloud.training

Asynchronous Invocation

- With **asynchronous** invocation, Lambda queues the event for processing and returns a response immediately
- For asynchronous invocation, Lambda handles retries and can send invocation records to a destination
- For asynchronous invocation, Lambda places the event in a queue and returns a success response without additional information
- To invoke a function asynchronously, set the invocation type parameter to Event

```
$ aws lambda invoke --function-name my-function --invocation-type Event --payload '{ "key": "value" }' response.json
{
"StatusCode": 202
}
```

© Digital Cloud Training | https://digitalcloud.training

Event Source Mapping

- To process items from a stream or queue, you can create an event source mapping
- An event source mapping is an AWS Lambda resource that reads from an event source and invokes a Lambda function
- You can use event source mappings to process items from a stream or queue
- An event source mapping uses permissions in the function's execution role to read and manage items in the event source
- Event source mappings are used for processing events in:
  - Amazon SQS queues
  - Amazon Kinesis streams
  - Amazon DynamoDB streams

© Digital Cloud Training | https://digitalcloud.training

Amazon SQS Event Source Mapping

Note: Supported poll-based services are DynamoDB, Kinesis, and SQS

© Digital Cloud Training | https://digitalcloud.training

Event Source Mapping

- The configuration of the event source mapping for stream and queue-based services (DynamoDB, Kinesis, and Amazon SQS), is made on the Lambda side
- For other services such as Amazon S3 and SNS, the function is invoked **asynchronously** and the configuration is made on the source (S3/SNS) rather than Lambda

© Digital Cloud Training | https://digitalcloud.training

Event Notifications

- You can use Lambda to process event notifications from Amazon S3
- Amazon S3 can send an event to a Lambda function when an object is created or deleted
- You configure notification settings on a bucket, and grant Amazon S3 permission to invoke a function on the function's resource-based permissions policy
- Amazon S3 invokes your function asynchronously with an event that contains details about the object

© Digital Cloud Training | https://digitalcloud.training

S3 Event Notification

© Digital Cloud Training | https://digitalcloud.training

Invoking Functions

© Digital Cloud Training | https://digitalcloud.training

Invoking AWS Lambda from the CLI

© Digital Cloud Training | https://digitalcloud.training

Create
Event Source Mapping

© Digital Cloud Training | https://digitalcloud.training

Create Event Source Mapping

© Digital Cloud Training | https://digitalcloud.training

Lambda Versions and Aliases

© Digital Cloud Training | https://digitalcloud.training

Lambda Versions

- Versioning means you can have multiple versions of your function
- The function version includes the following information:
  - The function code and all associated dependencies
  - The Lambda runtime that executes the function
  - All the function settings, including the environment variables
  - A unique Amazon Resource Name (ARN) to identify this version of the function
- You can use versions to manage the deployment of your AWS Lambda functions
- For example, you can publish a new version of a function for beta testing without affecting users of the stable production version

© Digital Cloud Training | https://digitalcloud.training

Lambda Versions

- You work on $LATEST which is the latest version of the code - this is mutable (changeable)

MyFunction:$LATEST

- When you're ready to publish a Lambda function you create a version (these are numbered)

MyFunction:1

- Numbered versions are assigned a number starting with 1 and subsequent versions are incremented by 1

MyFunction:2

- Versions are immutable (code cannot be edited)

© Digital Cloud Training | https://digitalcloud.training

Lambda Versions

- Each version has its own ARN
- This allows you to effectively manage them for different environments like Production, Staging or Development
- A **qualified** ARN has a version suffix:

```
arn:aws:lambda:us-east-1:555511112222:function:myfunction:2
```

- An **unqualified** ARN does not have a version suffix:

```
arn:aws:lambda:us-east-1:555511112222:function:myfunction
```

- You cannot create an alias from an unqualified ARN
- When you invoke a function using an unqualified ARN, Lambda implicitly invokes $LATEST

© Digital Cloud Training | https://digitalcloud.training

Lambda Aliases

© Digital Cloud Training | https://digitalcloud.training

Lambda Aliases

- Lambda aliases are pointers to a specific Lambda version
- Using an alias, you can invoke a function without having to know which version of the function is being referenced
- Aliases are mutable (changeable)

Specify the version to point the alias at

An additional version can be specified

20% of traffic is being direct to version 2

© Digital Cloud Training | https://digitalcloud.training

Lambda Aliases

- Aliases also have static ARNs but can point to any version of the same function

```
arn:aws:lambda:us-east-1:555511112222:function:myfunction:myappalias
```

- Aliases enable stable configuration of event triggers / destinations
- Aliases enable blue / green deployment by assigning weights to Lambda version
- You must create a version for an alias, you cannot use $LATEST

© Digital Cloud Training | https://digitalcloud.training

Using Versions and Aliases

© Digital Cloud Training | https://digitalcloud.training

Deployment Packages and Environment Variables

© Digital Cloud Training | https://digitalcloud.training

Lambda Deployment Packages

- A Lambda function's code consists of scripts or compiled programs and their dependencies
- A deployment package is used to deploy function code to Lambda
- Lambda supports two types of deployment packages:
  - Container images
  - .zip file archives

© Digital Cloud Training | https://digitalcloud.training

Deployment Packages – Container Images

- A container image includes:
  - The base operating system
  - The runtime
  - Lambda extensions
  - Application code and its dependencies
- Container images are uploaded to the Amazon Elastic Container Registry (ECR)
- The image is then deployed to the Lambda function

© Digital Cloud Training | https://digitalcloud.training

Deployment Packages – .zip archives

- A .zip file archive includes your application code and its dependencies
- The deployment package is uploaded from Amazon S3 or your computer
- There are limits to the size of zip archives:
  - 50 MB (zipped, for direct upload)
  - 250 MB (unzipped)
  - 3 MB (console editor)

© Digital Cloud Training | https://digitalcloud.training

Deployment through CloudFormation

- The AWS::Lambda::Function resource creates a Lambda function
- The function code zip file must be stored in Amazon S3
- The S3 bucket must be in the same region where you're running CloudFormation
- You also need a CloudFormation template file
- Set the package type to:
  - Image for container images
  - Zip for .zip archives

© Digital Cloud Training | https://digitalcloud.training

Lambda Layers

- You can configure your Lambda function to pull in additional code and content in the form of layers
- A layer is a ZIP archive that contains libraries, a custom runtime, or other dependencies
- With layers, you can use libraries in your function without needing to include them in your deployment package
- A function can use up to 5 layers at a time
- Layers are extracted to the /opt directory in the function execution environment
- Each runtime looks for libraries in a different location under /opt, depending on the language

© Digital Cloud Training | https://digitalcloud.training

Lambda Layers

- To add layers to your function, use the update-function-configuration command
- The following example adds two layers: one from the same account as the function, and one from a different account:

```
$ aws lambda update-function-configuration --function-name my-function \
--layers arn:aws:lambda:us-east-2:123456789012:layer:my-layer:3 \
arn:aws:lambda:us-east-2:210987654321:layer:their-layer:2
{
    "FunctionName": "test-layers",
    "FunctionArn": "arn:aws:lambda:us-east-2:123456789012:function:my-function",
    "Runtime": "nodejs12.x",
    "Role": "arn:aws:iam::123456789012:role/service-role/lambda-role",
    "Layers": [
        {
            "Arn": "arn:aws:lambda:us-east-2:123456789012:layer:my-layer:3",
            "CodeSize": 169
        },
        {
            "Arn": "arn:aws:lambda:us-east-2:210987654321:layer:their-layer:2",
            "CodeSize": 169
        }
    ],
    "RevisionId": "81cc64f5-5772-449a-b63e-12330476bcc4",
    ...
}
```

© Digital Cloud Training | https://digitalcloud.training

Lambda Environment Variables

- Environment variables can be used to adjust your function's behavior without updating code
- An environment variable is a pair of strings that is stored in a function's version-specific configuration
- Environment variables are defined on the unpublished version of a function
- When you publish a version, the environment variables are locked for that version
- Environment variables are key/value pairs

© Digital Cloud Training | https://digitalcloud.training

Lambda Environment Variables

- The following example sets two environment variables on a function named my-function

```
aws lambda update-function-configuration --function-name my-function \
    --environment "Variables={BUCKET=my-bucket,KEY=file.txt}"
```

- When viewing the function configuration with get-function-configuration we can see the following output:

```
{
    "FunctionName": "my-function",
    "FunctionArn": "arn:aws:lambda:us-east-2:123456789012:function:my-function",
    "Runtime": "nodejs12.x",
    "Role": "arn:aws:iam::123456789012:role/lambda-role",
    "Environment": {
        "Variables": {
            "BUCKET": "my-bucket",
            "KEY": "file.txt"
        }
    },
    "RevisionId": "0894d3c1-2a3d-4d48-bf7f-abade99f3c15",
    ...
}
```

© Digital Cloud Training | https://digitalcloud.training

Lambda Function Limits

Resource	Quota
Memory allocation	128 MB to 10,240 MB, in 1-MB increments
Function timeout (max execution)	900 seconds (15 minutes)
Environment variables	4 KB for all the function's environment variables
Layers	Up to 5
Burst concurrency	500-300 depending on the Region
Invocation payload	6 MB (synchronous) 256 KB (asynchronous
Deployment package (.zip archive) size	50 MB (zipped, for direct upload) 250 MB (unzipped) 3 MB (console)
Container image code package size	10 GB
/tmp directory storage	512 MB to 10,240 MB, in 1-MB increments

© Digital Cloud Training | https://digitalcloud.training

Using Environment Variables

© Digital Cloud Training | https://digitalcloud.training

Destinations and Dead-Letter Queues

© Digital Cloud Training | https://digitalcloud.training

Success and Failure Destinations

- Send invocation records to a destination when your function is invoked

Works for asynchronous invocations and Stream invocations (Kinesis/DDB stream)

Choose the condition as success of failure

Destination type is SQS queue, SNS topic, Lambda function, or EventBridge event bus

© Digital Cloud Training | https://digitalcloud.training

Success and Failure Destinations

- The execution record contains details about the request and response in JSON format
- Information sent includes:
  - Version
  - Timestamp
  - Request context
  - Request payload
  - Response context
  - Response payload

© Digital Cloud Training | https://digitalcloud.training

Dead-Letter Queue (DLQ)

- A DLQ saves unprocessed events for further processing
- Applies to **asynchronous** invocations
- The DLQ can be an Amazon SQS queue or an Amazon SNS topic
- When editing the asynchronous configuration, you can specify the number of retries:

Lambda will retry processing up to **2 times**. After **6 hours** the event is discarded and can be sent to a DLQ

Destinations and DLQ

© Digital Cloud Training | https://digitalcloud.training

Reserved and Provisioned Concurrency

© Digital Cloud Training | https://digitalcloud.training

Lambda Function Concurrency

Additional functions are initialized up to the **burst** or **account limit**

Function **invocation** → Lambda → Function is **executed**

If the concurrency **limit** is **exceeded** throttling occurs with error **"Rate exceeded"** and a 429 **"TooManyRequestsException"**

Burst **concurrency** quotas:

- 3000 – US West (Oregon), US East (N. Virginia), Europe (Ireland)
- 1000 – Asia Pacific (Tokyo), Europe (Frankfurt), US East (Ohio)
- 500 – Other Regions

© Digital Cloud Training | https://digitalcloud.training

Concurrency Limits

- The default concurrency limit per AWS Region is 1,000 invocations at any given time
- The default burst concurrency quota per Region is between 500 and 3,000, which varies per Region
- There is no maximum concurrency limit for Lambda functions (depends on use case)
- To avoid throttling request limit increases at least 2 weeks ahead of time

© Digital Cloud Training | https://digitalcloud.training

Reserved Concurrency

- Reserved concurrency guarantees a set number of concurrent executions will be available for a critical function
- You can reserve up to the Unreserved account concurrency value, minus 100 for functions that don't have reserved concurrency
- To throttle a function, set the reserved concurrency to zero. This stops any events from being processed until you remove the limit

© Digital Cloud Training | https://digitalcloud.training

Provisioned Concurrency

- When provisioned concurrency is allocated, the function scales with the same burst behavior as standard concurrency
- After it's allocated, provisioned concurrency serves incoming requests with very low latency
- When all provisioned concurrency is in use, the function scales up normally to handle any additional requests
- Application Auto Scaling takes this a step further by providing autoscaling for provisioned concurrency

Monitoring, Logging, and Tracing

© Digital Cloud Training | https://digitalcloud.training

Performance Monitoring - CloudWatch

- Lambda sends metrics to Amazon CloudWatch for performance monitoring

© Digital Cloud Training | https://digitalcloud.training

Logging - CloudWatch

- Execution logs are stored in Amazon CloudWatch Logs
- The Lambda function execution role must have permissions (IAM) to allow writes to CloudWatch Logs

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "logs:CreateLogGroup",
            "Resource": "arn:aws:logs:us-east-1:555511112222:*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:us-east-1:555511112222:log-group:/aws/lambda/myfunction:*"
            ]
        }
    ]
}
```

© Digital Cloud Training | https://digitalcloud.training

Tracing with AWS X-Ray

- You can use AWS X-Ray to visualize the components of your application, identify performance bottlenecks, and troubleshoot requests that resulted in an error
- Your Lambda functions send trace data to X-Ray, and X-Ray processes the data to generate a service map and searchable trace summaries

© Digital Cloud Training | https://digitalcloud.training

Tracing with AWS X-Ray

- The AWS X-Ray Daemon is a software application that gathers raw segment data and relays it to the AWS X-Ray service
- The daemon works in conjunction with the AWS X-Ray SDKs so that data sent by the SDKs can reach the X-Ray service
- When you trace your Lambda function, the X-Ray daemon automatically runs in the Lambda environment to gather trace data and send it to X-Ray
- The function needs permissions to write to X-Ray in the execution role

© Digital Cloud Training | https://digitalcloud.training

Lambda in a VPC and ALB Targets

© Digital Cloud Training | https://digitalcloud.training

Connecting Lambda to an Amazon VPC

© Digital Cloud Training | https://digitalcloud.training

Connecting Lambda to an Amazon VPC

- You must connect to a private subnet with a NAT Gateway for Internet access (no public IP)
- Careful with DNS resolution of public hostnames as it could add to function running time (cost)
- Cannot be connected to a dedicated tenancy VPC
- Only connect to a VPC if you need to, it can slow down function execution
- To connect to a VPC, your function's execution role must have the following permissions:
  - ec2:CreateNetworkInterface
  - ec2:DescribeNetworkInterfaces
  - ec2:DeleteNetworkInterface
- These permissions are included in the AWSLambdaVPCAccessExecutionRole managed policy

© Digital Cloud Training | https://digitalcloud.training

Lambda Function as a Target for an ALB

- Application Load Balancers (ALBs) support AWS Lambda functions as targets
- You can register your Lambda functions as targets and configure a listener rule to forward requests to the target group for your Lambda function

© Digital Cloud Training | https://digitalcloud.training

Lambda Function as a Target for an ALB

© Digital Cloud Training | https://digitalcloud.training

Lambda Function as a Target for an ALB

There are some limits to understand:

- The Lambda function and target group must be in the same account and in the same Region
- The maximum size of the request body that you can send to a Lambda function is 1 MB
- The maximum size of the response JSON that the Lambda function can send is 1 MB
- WebSockets are not supported. Upgrade requests are rejected with an HTTP 400 code
- Local Zones are not supported

© Digital Cloud Training | https://digitalcloud.training

Security for Lambda Functions

© Digital Cloud Training | https://digitalcloud.training

Security for AWS Lambda Functions

AWS recommend to use **Secrets Manager** instead of environment variables

AWS Secrets Manager

Amazon CloudWatch Logs

DynamoDB Table

Lambda API endpoints only support **TLS** connections

Developer Client

AWS Lambda Function

IAM Role

Function **execution role** must provide permissions to AWS services

Environment Variable

Files, deployment packages and environment variables are encrypted **at rest** with a **KMS key**

© Digital Cloud Training | https://digitalcloud.training

DigitalCloud TRAINING

AWS Signer

- AWS Signer is a fully managed code-signing service
- Used to ensure the trust and integrity of code
- Code is validated against a digital signature
- With Lambda you can ensure only trusted code runs in Lambda functions
- Signer is used to create digitally signed packages for deployment
- IAM policies can enforce that functions can be created only if they have code signing enabled
- If a developer leaves you can revoke all versions of the signing profile so the code cannot run

© Digital Cloud Training | https://digitalcloud.training

Best Practices for Developing Lambda Functions

© Digital Cloud Training | https://digitalcloud.training

Best Practices – Function Code

Separate the Lambda handler from your core logic

- This allows you to make a more unit-testable function

This example code uses **Node.js** and separates core logic from the Lambda handler

```
exports.myHandler = function(event, context, callback) {
    var foo = event.foo;
    var bar = event.bar;
    var result = MyLambdaFunction (foo, bar);

    callback(null, result);
}

function MyLambdaFunction (foo, bar) {
    // MyLambdaFunction logic here
}
```

© Digital Cloud Training | https://digitalcloud.training

Best Practices – Function Code

Take advantage of execution environment reuse to improve the performance of your function

- Initialize SDK clients and database connections outside of the function handler
- Cache static assets locally in the /tmp directory
- Subsequent invocations processed by the same instance of your function can reuse these resources
- This saves cost by reducing function run time

Use a keep-alive directive to maintain persistent connections

- Lambda purges idle connections over time and attempting to reuse an idle connection can result in an error

© Digital Cloud Training | https://digitalcloud.training

Best Practices – Function Code

Use environment variables to pass operational parameters to your function

- For example, use environment variables instead of hard-coding S3 bucket names in code

Control the dependencies in your function's deployment package

- Libraries and runtimes in the Lambda execution environment can be updated and may cause issues
- To avoid errors, package dependencies in the deployment package

© Digital Cloud Training | https://digitalcloud.training

Best Practices – Function Code

Minimize your deployment package size to its runtime necessities

- This reduces the amount of time it takes for deployment packages to be downloaded and unpacked ahead of invocation

Avoid using recursive code in your function

- Don’t use code that calls itself until some arbitrary criteria is met
- Can cause large volumes of invocations and increased costs

© Digital Cloud Training | https://digitalcloud.training

AWS Serverless Application Model (SAM)

© Digital Cloud Training | https://digitalcloud.training

AWS Serverless Application Model (SAM)

- Provides a **shorthand syntax** to express functions, APIs, databases, and event source mappings
- Can be used to create Lambda functions, API endpoints, DynamoDB tables, and other resources

```yaml
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
Resources:
  HelloWorldFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: hello-world/
      Handler: app.lambdaHandler
      Runtime: nodejs12.x
```

SAM Template (YAML)

Commands:

1. sam package
2. sam deploy

Application Code

S3 Bucket

AWS CloudFormation

Stack

Change set

SAM uploads source files to bucket and initiates CloudFormation

© Digital Cloud Training | https://digitalcloud.training

AWS Serverless Application Model (SAM)

- A SAM template file is a YAML configuration that represents the architecture of a serverless application
- You use the template to declare all the AWS resources that comprise your serverless application in one place
- AWS SAM templates are an extension of AWS CloudFormation templates
- Any resource that you can declare in an AWS CloudFormation template can also be declared in an AWS SAM template

© Digital Cloud Training | https://digitalcloud.training

AWS SAM Commands

```
sam package -t template.yaml --s3-bucket dctlabs --output-template-file packaged-template.yaml
```

```
sam deploy --template-file packaged-template.yaml --stack-name my-cf-stack
```

Alternatively, run:

```
aws cloudformation package --template-file template.yaml --s3-bucket dctlabs --output-template-file packaged-template.yaml
```

```
aws cloudformation deploy --template-file packaged-template.yaml --stack-name my-cf-stack
```

© Digital Cloud Training | https://digitalcloud.training

AWS SAM Transform Header

The “Transform” header indicates it’s a SAM template:

- Transform: 'AWS::Serverless-2016-10-31'

There are several resources types:

- AWS::Serverless::Function (AWS Lambda)
- AWS::Serverless::Api (API Gateway)
- AWS::Serverless::SimpleTable (DynamoDB)
- AWS::Serverless::Application (AWS Serverless Application Repository)
- AWS::Serverless::HttpApi (API Gateway HTTP API)
- AWS::Serverless::LayerVersion (Lambda layers)

© Digital Cloud Training | https://digitalcloud.training

Run SAM App Using CloudShell

© Digital Cloud Training | https://digitalcloud.training

The Serverless Application Repository

© Digital Cloud Training | https://digitalcloud.training

SECTION 8

Amazon DynamoDB

© Digital Cloud Training | https://digitalcloud.training

Amazon DynamoDB

© Digital Cloud Training | https://digitalcloud.training

Amazon DynamoDB

- Fully managed NoSQL database service
- Key/value store and document store
- It is a non-relational, key-value type of database
- Fully serverless service
- Push button scaling

Data is stored in **partitions** which are replicated across multiple AZs in a Region

DynamoDB Table

© Digital Cloud Training | https://digitalcloud.training

Amazon DynamoDB

- DynamoDB provides low latency (milliseconds)
- Microsecond latency can be achieved with DynamoDB Accelerator (DAX)
- All data is stored on SSD storage
- Data is replicated across multiple AZs in a Region
- DynamoDB Global Tables synchronizes tables across Regions

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Features

DynamoDB Feature	Benefit
Serverless	Fully managed, fault tolerant, service
Highly available	99.99% availability SLA – 99.999% for Global Tables!
NoSQL type of database with Name / Value structure	Flexible schema, good for when data is not well structured or unpredictable
Horizontal scaling	Seamless scalability to any scale with push button scaling or Auto Scaling
DynamoDB Streams	Captures a time-ordered sequence of item-level modifications in a DynamoDB table and durably stores the information for up to 24 hours. Often used with Lambda and the Kinesis Client Library (KCL)
DynamoDB Accelerator (DAX)	Fully managed in-memory cache for DynamoDB that increases performance (microsecond latency)
Transaction options	Strongly consistent or eventually consistent reads, support for ACID transactions
Backup	Point-in-time recovery down to the second in last 35 days; On-demand backup and restore
Global Tables	Fully managed multi-region, multi-master solution

DigitalCloud TRAINING

DynamoDB Core Components

The basic DynamoDB components are:

- **Tables**
- **Items**
- **Attributes**

userid	orderid	book	price	date
user001	1000092	ISBN100..	9.99	2020.04..
user002	1000102	ISBN100..	24.99	2020.03..
user003	1000168	ISBN2X0..	12.50	2020.04..

© Digital Cloud Training | https://digitalcloud.training

DynamoDB API

- API operations are categorized as **control plane** or **data plane**
- Example **control plane** API actions:
  - CreateTable – Creates a new table
  - DescribeTable – Returns information about a table, such as its primary key schema, throughput settings, and index information
  - ListTables – Returns the names of all your tables in a list
  - UpdateTable – Modifies the settings of a table or its indexes
  - DeleteTable – Removes a table and all its dependent objects from DynamoDB

© Digital Cloud Training | https://digitalcloud.training

DynamoDB API

- Data plane API actions can be performed using PartiQL (SQL compatible), or classic DynamoDB CRUD APIs
- Example **data plane** API actions (DynamoDB CRUD):
  - PutItem – Writes a single item to a table
  - BatchWriteItem – Writes up to 25 items to a table
  - GetItem – Retrieves a single item from a table
  - BatchGetItem – Retrieves up to 100 items from one or more tables
  - UpdateItem – Modifies one or more attributes in an item
  - DeleteItem – Deletes a single item from a table

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Supported Data Types

- DynamoDB supports many data types
- They can be categorized as follows:
  - **Scalar Types** – A scalar type can represent exactly one value. The scalar types are number, string, binary, Boolean, and null
  - **Document Types** – A document type can represent a complex structure with nested attributes, such as you would find in a JSON document. The document types are list and map
  - **Set Types** – A set type can represent multiple scalar values. The set types are string set, number set, and binary set

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Table Classes

DynamoDB Standard – default and recommended for most workloads

DynamoDB Standard-Infrequent Access (DynamoDB Standard-IA)

Lower cost storage for tables that store infrequently accessed data, such as:

- Application logs
- Old social media posts
- E-commerce order history
- Past gaming achievements

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Access Control

- All authentication and access control is managed using IAM
- DynamoDB supports **identity-based** policies:
  - Attach a permissions policy to a user or a group in your account
  - Attach a permissions policy to a role (grant cross-account permissions)
- DynamoDB **doesn't** support **resource-based** policies
- You can use a special IAM condition to restrict user access to only their own records

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Access Control

- The primary DynamoDB resources are tables
- Also supports additional resource types, indexes, and streams
- You can create indexes and streams only in the context of an existing DynamoDB table (subresources)
- These resources and subresources have unique ARNs associated with them, as shown in the following table:

Resource Type	ARN Format
Table	arn:aws:dynamodb:region:account-id:table/table-name
Index	arn:aws:dynamodb:region:account-id:table/table-name/index/index-name
Stream	arn:aws:dynamodb:region:account-id:table/table-name/stream/stream-label

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Access Control

- The following example policy grants permissions for one DynamoDB action (dynamodb:ListTables):

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListTables",
            "Effect": "Allow",
            "Action": [
                "dynamodb:ListTables"
            ],
            "Resource": "*"
        }
    ]
}
```

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Access Control

- The following example policy grants permissions for three DynamoDB actions:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DescribeQueryScanBooksTable",
            "Effect": "Allow",
            "Action": [
                "dynamodb:DescribeTable",
                "dynamodb:Query",
                "dynamodb:Scan"
            ],
            "Resource": "arn:aws:dynamodb:us-west-2:account-id:table/Books"
        }
    ]
}
```

In this example the actions are **restricted** to the specified **resource** (DynamoDB table)

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Partitions and Primary Keys

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Partitions and Primary Keys

- Amazon DynamoDB stores data in partitions
- A partition is an allocation of storage for a table that is automatically replicated across multiple AZs within an AWS Region
- DynamoDB manages partitions for you
- DynamoDB allocates sufficient partitions to support provisioned throughput requirements
- DynamoDB allocates additional partitions to a table in the following situations:
  - If you increase the table's provisioned throughput settings beyond what the existing partitions can support
  - If an existing partition fills to capacity and more storage space is required

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Partitions and Primary Keys

- There are two types of Primary key – **Partition keys** and **composite keys**
- **Partition key** - unique attribute (e.g. user ID)
  - Value of the Partition key is input to an internal hash function which determines the partition or physical location on which the data is stored
  - If you are using the Partition key as your Primary key, then no two items can have the same partition key

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Partitions and Primary Keys

Attributes

Partition Key

postid	subject	message	lastposttime	replies
1000000572	Hello World	This code..	2020-04..	12
1000001982	Advice on..	How can..	2020-03..	39
1000000239	Help!	I need…	2020-01..	52

© Digital Cloud Training | https://digitalcloud.training

DigitalCloud TRAINING

DynamoDB Partitions and Primary Keys

Composite key - Partition key + Sort key in combination

- Example is user posting to a forum. Partition key would be the user ID, Sort key would be the timestamp of the post
- 2 items may have the same Partition key, but they must have a different Sort key
- All items with the same Partition key are stored together, then sorted according to the Sort key value
- Allows you to store multiple items with the same partition key

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Partitions and Primary Keys

This is known as a **composite key** as it has a **partition key** + **sort key**

Primary Key

Attributes

Partition Key	Sort Key					
clientid	created	sku	category	size	color	weight
john@example.com	1583975308	SKU-S523	T-Shirt	Small	Red	Light
chris@example.com	1583975613	SKU-J091	Pen		Blue	
chris@example.com	1583975449	SKU-A234	Mug			
sarah@example.com	1583976311	SKU-R873	Chair			4011
jenny@example.com	1583976323	SKU-I019	Plate	30		

Data structure can be **unpredictable**

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Partitions and Primary Keys

- DynamoDB evenly distributes provisioned throughput—**read capacity units** (RCUs) and **write capacity units** (WCUs) among partitions
- If your access pattern exceeds 3000 RCU or 1000 WCU for a single partition key value, your requests might be throttled
- Reading or writing above the limit can be caused by these issues:
  - Uneven distribution of data due to the wrong choice of partition key
  - Frequent access of the same key in a partition (the most popular item, also known as a hot key)
  - A request rate greater than the provisioned throughput

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Partitions and Primary Keys

Best practices for partition keys:

- Use **high-cardinality** attributes – e.g. e-mailid, employee_no, customerid, sessionid, orderid, and so on
- Use **composite attributes** – e.g. customerid+productid+countrycode as the partition key and order_date as the sort key
- **Cache** popular items – use DynamoDB accelerator (DAX) for caching reads
- Add **random numbers** or digits from a predetermined range for write-heavy use cases
- For example, add a random suffix to an invoice number such as INV00023-04593

© Digital Cloud Training | https://digitalcloud.training

Practice Creating DynamoDB Tables

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Consistency Models and Transactions

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Consistency Models

- DynamoDB supports **eventually consistent** and **strongly consistent** reads
- **Eventually consistent** reads:
  - When you read data from a DynamoDB table, the response might not reflect the results of a recently completed write operation
  - The response might include some stale data
  - If you repeat your read request after a short time, the response should return the latest data

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Consistency Models

Strongly consistent read:

- DynamoDB returns a response with the most up-to-date data, reflecting the updates from all prior write operations that were successful
- A strongly consistent read might not be available if there is a network delay or outage. In this case, DynamoDB may return a server error (HTTP 500)
- Strongly consistent reads may have higher latency than eventually consistent reads
- Strongly consistent reads are not supported on global secondary indexes
- Strongly consistent reads use more throughput capacity than eventually consistent reads

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Consistency Models

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Consistency Models

- With a **strongly consistent** read, data will always be returned when reading after a successful write
- DynamoDB uses eventually consistent reads by default
- You can configure strongly consistent reads with the GetItem, Query and Scan APIs by setting the --consistent-read (or ConsistentRead) parameter to true

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Transactions

- With DynamoDB transactions DynamoDB makes coordinated, all-or-nothing changes to multiple items both within and across tables
- Transactions provide atomicity, consistency, isolation, and durability (ACID) in DynamoDB
- Enables reading and writing of multiple items across multiple tables as an all or nothing operation
- Checks for a pre-requisite condition before writing to a table

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Transaction Write API

- With the transaction write API, you can group multiple Put, Update, Delete, and ConditionCheck actions
- You can then submit the actions as a single TransactWriteItems operation that either succeeds or fails as a unit
- The same is true for multiple Get actions, which you can group and submit as a single TransactGetItems operation

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Transactions

- There is no additional cost to enable transactions for DynamoDB tables
- You pay only for the reads or writes that are part of your transaction
- DynamoDB performs **two** underlying **reads** or **writes** of every item in the transaction: one to prepare the transaction and one to commit the transaction

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Transactions

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Transactions

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Capacity Units (RCU/WCU)

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Provisioned Capacity

- Provisioned capacity is the default setting
- You specify the reads and write per second
- Can enabled auto scaling for dynamic adjustments
- Capacity is specified using:
  - **Read Capacity Units** (RCUs)
  - **Write Capacity Units** (WCUs)

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Provisioned Capacity

Read Capacity Units (RCUs):

- Each API call to read data from your table is a read request
- Read requests can be strongly consistent, eventually consistent, or transactional
- For items up to 4 KB in size, one RCU equals:
  - One strongly consistent read request per second
  - Two eventually consistent read requests per second
  - 0.5 transactional read requests per second
- Items larger than 4 KB require additional RCUs

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Provisioned Capacity

Calculating RCUs (examples):

Requirement	RCUs Needed
10 strongly consistent reads per second of 4 KB each	10 * 4 KB / 4 KB = **10 RCU**
10 strongly consistent reads per second of 11 KB each	10 * 12 KB / 4 KB = **30 RCU**
20 eventually consistent reads per second of 12 KB each	(20 / 2) * (12 / 4) – **30 RCU**
36 eventually consistent reads per second of 16 KB each	(36 / 2) * (16 / 4) = **72 RCU**

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Provisioned Capacity

Write Capacity Units (WCUs):

- Each API call to write data to your table is a write request
- For items up to 1 KB in size, one WCU can perform:
  - One standard write request per second
  - 0.5 transactional writes requests (one transactional write requires two WCUs)
- Items larger than 1 KB require additional WCUs

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Provisioned Capacity

Calculating WCUs (examples):

Requirement	WCUs Needed
10 standard writes per second of 4 KB each	10 * 4 = **40 WCU**
12 standard writes per second of 9.5 KB each (9.5 gets rounded up)	12 * 10 = **120 WCU**
10 transactional writes per second of 4 KB each	10 * 2 * 4 = **80 WCU**
12 transactional writes per second of 9.5 KB each	12 * 2 * 10 = **240 WCU**

© Digital Cloud Training | https://digitalcloud.training

DynamoDB On-Demand Capacity

- With on-demand, you don't need to specify your requirements
- DynamoDB instantly scales up and down based on the activity of your application
- Great for unpredictable / spikey workloads or new workloads that aren't well understood
- You pay for what you use (pay per request)

© Digital Cloud Training | https://digitalcloud.training

DynamoDB
Performance and Throttling

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Performance and Throttling

- Throttling occurs when the configured RCU or WCU are exceeded
- You may receive the following error:

  ProvisionedThroughputExceededException
- This error indicates that the request rate is too high for the read / write capacity provisioned for the table
- The AWS SDKs for DynamoDB automatically retry requests that receive this exception
- The request is eventually successful, unless the retry queue is too large to finish

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Performance and Throttling

Possible causes of performance issues:

- **Hot keys** – one partition key is being read too often
- **Hot partitions** - when data access is imbalanced, a "hot" partition can receive a higher volume of read and write traffic compared to other partitions
- **Large items** – large items consume more RCUs and WCUs

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Performance and Throttling

Resolution:

- **Reduce the frequency of requests** and use exponential backoff
- Try to design your application for **uniform activity** across all logical partition keys in the table and its secondary indexes
- **Use burst capacity effectively** - DynamoDB retains up to 5 minutes (300 seconds) of unused read and write capacity which can be consumed quickly
- **Ensure Adaptive Capacity is enabled** (default) – this feature minimizes throttling due to throughput exceptions

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Scan and Query API

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Scan API

- The Scan operation returns one or more items and item attributes by **accessing** every item in a table or a secondary index
- To have DynamoDB return fewer items, you can provide a FilterExpression operation
- A single Scan operation reads up to the maximum number of items set (if using the Limit parameter), or a maximum of 1 MB
- Scan API calls can use a lot of RCUs as they access every item in the table

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Scan API

- Scan operations proceed sequentially
- Applications can request a parallel Scan operation by providing the Segment and TotalSegments parameters
- Scan uses **eventually consistent** reads when accessing the data in a table
- If you need a consistent copy of the data, as of the time that the Scan begins, you can set the ConsistentRead parameter to true

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Scan API

All items and **all attributes** are returned

Developer

Results ←

Scan API request →

Attributes

Partition Key

postid	subject	message	lastposttime	replies
1000000572	Hello World	This code..	2020-04..	12
1000001982	Advice on..	How can..	2020-03..	39
1000000239	Help!	I need…	2020-01..	52

Every item in the table is accessed (consumes lots of RCUs)

© Digital Cloud Training | https://digitalcloud.training

DigitalCloud TRAINING

DynamoDB Scan API with Projection Expression

- In this example the scan will return all posts in the forum that were posted within a date range and that have more than 50 replies:

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Scan API

Select attributes are returned

Results

Scan API with projection expression

Developer

Primary Key		Attributes				
Partition Key	Sort Key					
clientid	created	sku	category	size	color	weight
john@example.com	2020-03-9T08:12Z	SKU-S523	T-Shirt	Small	Red	Light
chris@example.com	2020-03-10T14:30Z	SKU-J091	Pen		Blue	
chris@example.com	2020-03-10T15:30Z	SKU-A234	Mug	12		
sarah@example.com	2020-03-12T7:42Z	SKU-R873	Chair	94		4011
jenny@example.com	2020-03-13T18:29Z	SKU-I019	Plate	30		

Every item in the table is accessed (consumes lots of RCUs)

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Query API

- A query operation finds items in your table based on the **primary key** attribute and a **distinct value** to search for
- For example, you might search for a user ID value and all attributes related to that item would be returned
- You can use an optional sort key name and value to refine the results
- For example, if your sort key is a timestamp, you can refine the query to only select items with a timestamp of the last 7 days
- All attributes are returned for the items by default
- You can also use the ProjectionExpression parameter if you want the query to only return the attributes you want to see
- By default, queries are eventually consistent
- To use strongly consistent, you need to explicitly set this in the query

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Query API

Primary Key

Attributes

Partition Key

Sort Key

Select items and **attributes** are returned

Results

Query API request with projection expression

Developer

clientid	created	sku	category	size	color	weight
john@example.com	2020-03-9T08:12Z	SKU-S523	T-Shirt	Small	Red	Light
chris@example.com	2020-03-10T14:30Z	SKU-J091	Pen		Blue	
chris@example.com	2020-03-10T15:30Z	SKU-A234	Mug	12		
sarah@example.com	2020-03-12T7:42Z	SKU-R873	Chair	94		4011
jenny@example.com	2020-03-13T18:29Z	SKU-I019	Plate	30		

Select items with **select attributes** are accessed (consumes fewer RCUs)

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Query API

- In this example the query returns only items with the client id of chris@example.com that were created within a certain date range and that are in the category pen:

Searching DynamoDB – Scan API

© Digital Cloud Training | https://digitalcloud.training

Searching DynamoDB – Query API

© Digital Cloud Training | https://digitalcloud.training

DynamoDB
LSI and GSI

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Local Secondary Index (LSI)

- Provides an **alternative sort key** to use for scans and queries
- Can create up to 5 LSIs per table
- Must be created at table creation time
- You cannot add, remove, or modify it later
- It has the same partition key as your original table (different sort key)
- Gives you a different view of your data, organized by alternative sort key
- Any queries based on this sort key are much faster using the index than the main table

DigitalCloud TRAINING

DynamoDB Local Secondary Index (LSI)

Primary Table

Partition Key	Sort Key
clientid	created
john@example.com	2020-03-9T08:12Z
chris@example.com	2020-03-10T14:30Z
chris@example.com	2020-03-10T15:30Z
sarah@example.com	2020-03-12T7:42Z
jenny@example.com	2020-03-13T18:29Z

Index is created from the primary table

LSI

Partition Key	Sort Key	
clientid	sku	
john@example.com	SKU-S523	
chris@example.com	SKU-J091	
chris@example.com	SKU-A234	
sarah@example.com	SKU-R873	
jenny@example.com	SKU-I019	

Attributes can be optionally "projected"

Sort Key is different on the LSI

© Digital Cloud Training | https://digitalcloud.training

DigitalCloud TRAINING

DynamoDB Local Secondary Index (LSI)

- In this example querying the main table, we must use the partition key clientid and the sort key created:

- In this example with an LSI, we can query the index for any orders made by chris@example.com for the product SKU-1922:

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Global Secondary Index (GSI)

- Used to speed up queries on **non-key attributes**
- You can create when you create your table or at any time
- Can specify a **different partition key** as well as a **different sort key**
- Gives a completely different view of the data
- Speeds up any queries relating to this alternative partition and sort key

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Global Secondary Index (GSI)

Primary Table

Partition Key	Sort Key
clientid	created
john@example.com	2020-03-9T08:12Z
chris@example.com	2020-03-10T14:30Z
chris@example.com	2020-03-10T15:30Z
sarah@example.com	2020-03-12T7:42Z
jenny@example.com	2020-03-13T18:29Z

Index is created from the primary table

GSI

Partition Key	Sort Key	
sku	qty	
SKU-S523	1	
SKU-J091	10	
SKU-A234	4	
SKU-R873	2	
SKU-I019	6	

Attributes can be optionally "projected"

© Digital Cloud Training | https://digitalcloud.training

DigitalCloud TRAINING

DynamoDB Global Secondary Index (GSI)

- In this example, we can query the index for orders of SKU-T122 where the quantity is greater than 1:

Create LSI and GSI

© Digital Cloud Training | https://digitalcloud.training

DynamoDB
Optimistic Locking and Conditional Updates

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Optimistic Locking

- Optimistic locking is a strategy to ensure that the client-side item that you are updating (or deleting) is the same as the item in Amazon DynamoDB
- Protects database writes from being overwritten by the writes of others, and vice versa

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Optimistic Locking

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Optimistic Locking

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Conditional Updates

- To manipulate data in an Amazon DynamoDB table, you use the PutItem, UpdateItem, and DeleteItem operations
- You can optionally specify a condition expression to determine which items should be modified
- If the condition expression evaluates to true, the operation succeeds; otherwise, the operation fails

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Conditional Updates

- This example CLI command allows the write to proceed only if the item in question does not already have the same key:

```
aws dynamodb put-item --table-name ProductCatalog --item file://item.json \
    --condition-expression "attribute_not_exists(Id)"
```

- This example CLI command uses attribute_not_exists to delete a product only if it does not have a Price attribute:

```
aws dynamodb delete-item --table-name ProductCatalog --key '{"Id": {"N": "456"}}' \
    --condition-expression "attribute_not_exists(Price)"
```

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Conditional Updates

- This example CLI command only deletes an item if the ProductCategory is either "Sporting Goods" or "Gardening Supplies" and the price is between 500 and 600:

```
aws dynamodb delete-item \
    --table-name ProductCatalog \
    --key '{"Id":{"N":"456"}}' \
    --condition-expression "(ProductCategory IN (:cat1, :cat2)) and (Price between :lo and :hi)" \
    --expression-attribute-values file://values.json
```

© Digital Cloud Training | https://digitalcloud.training

Adding a Time to Live (TTL) to Items

© Digital Cloud Training | https://digitalcloud.training

DynamoDB Time to Live (TTL)

- TTL lets you define when items in a table expire so that they can be automatically deleted from the database
- With TTL enabled on a table, you can set a timestamp for deletion on a per-item basis
- No extra cost and does not use WCU / RCU
- Helps reduce storage and manage the table size over time

© Digital Cloud Training | https://digitalcloud.training

Amazon DynamoDB Streams

© Digital Cloud Training | https://digitalcloud.training

Amazon DynamoDB Streams

1. Application inserts / updates /deletes item
2. A record is written to the DynamoDB stream
3. A Lambda function is triggered
4. The Lambda function writes to CloudWatch Logs

© Digital Cloud Training | https://digitalcloud.training

Amazon DynamoDB Streams

- DynamoDB Streams captures a **time-ordered sequence** of **item-level modifications** in any DynamoDB table
- The information is stored in a log for up to 24 hours
- Applications can access this log and view the data items as they appeared before and after they were modified, in near-real time
- You can also use the CreateTable or UpdateTable API operations to enable or modify a stream

© Digital Cloud Training | https://digitalcloud.training

Amazon DynamoDB Streams

- The StreamSpecification parameter determines how the stream is configured:
- StreamEnabled — Specifies whether a stream is enabled (true) or disabled (false) for the table
- StreamViewType — Specifies the information that will be written to the stream whenever data in the table is modified:
  - **KEYS_ONLY** — Only the key attributes of the modified item
  - **NEW_IMAGE** — The entire item, as it appears after it was modified
  - **OLD_IMAGE** — The entire item, as it appeared before it was modified
  - **NEW_AND_OLD_IMAGES** — Both the new and the old images of the item

© Digital Cloud Training | https://digitalcloud.training

Amazon DynamoDB Accelerator (DAX)

© Digital Cloud Training | https://digitalcloud.training

Amazon DynamoDB Accelerator (DAX)

- DAX is a managed service that provides in-memory acceleration for DynamoDB tables
- Improves performance from **milliseconds** to **microseconds**, even at millions of requests per second
- Provides managed cache invalidation, data population, and cluster management
- DAX is used to improve **READ performance** (not writes)
- You do not need to modify application logic, since DAX is compatible with existing DynamoDB API calls

© Digital Cloud Training | https://digitalcloud.training

Amazon DynamoDB Accelerator (DAX)

© Digital Cloud Training | https://digitalcloud.training

Amazon DynamoDB Accelerator (DAX)

- You can enable DAX with just a few clicks in the AWS Management Console or using the AWS SDK
- Just as with DynamoDB, you only pay for the capacity you provision
- Provisioned through clusters and charged by the node (runs on EC2 instances)
- Pricing is per node-hour consumed and is dependent on the instance type you select

© Digital Cloud Training | https://digitalcloud.training

DAX vs ElastiCache

- DAX is optimized for DynamoDB
- With ElastiCache you have more management overhead (e.g. invalidation)
- With ElastiCache you need to modify application code to point to cache
- ElastiCache supports more datastores

© Digital Cloud Training | https://digitalcloud.training

Amazon DynamoDB Global Tables

© Digital Cloud Training | https://digitalcloud.training

Amazon DynamoDB Global Tables

- DynamoDB global tables is a fully managed solution for deploying a multi-region, multi-master database
- When you create a global table, you specify the AWS Regions where you want the table to be available
- DynamoDB performs all the necessary tasks to create identical tables in these regions, and propagate ongoing data changes to all of them

© Digital Cloud Training | https://digitalcloud.training

Amazon DynamoDB Global Tables

© Digital Cloud Training | https://digitalcloud.training

Enable Global Table

© Digital Cloud Training | https://digitalcloud.training

SECTION 9

Application Integration and APIs

© Digital Cloud Training | https://digitalcloud.training

Amazon Simple Queue Service (SQS)

© Digital Cloud Training | https://digitalcloud.training

Decoupling with SQS Queues

© Digital Cloud Training | https://digitalcloud.training

Amazon SQS

- Amazon SQS is **pull-based**, not push-based (like SNS)
- Messages are up to 256KB in size
- Larger messages can be sent using the Amazon SQS Extended Client Library for Java
- Messages can be kept in the queue from 1 minute to 14 days
- Default retention period is 4 days
- Amazon SQS guarantees that your messages will be processed at least once

© Digital Cloud Training | https://digitalcloud.training

Amazon SQS Queue Types

© Digital Cloud Training | https://digitalcloud.training

Amazon SQS Queue Types

Standard Queue	FIFO Queue
Unlimited Throughput: Standard queues support a nearly unlimited number of transactions per second (TPS) per API action.	**High Throughput:** FIFO queues support up to 300 messages per second (300 send, receive, or delete operations per second). When you batch 10 messages per operation (maximum), FIFO queues can support up to 3,000 messages per second
Best-Effort Ordering: Occasionally, messages might be delivered in an order different from which they were sent	**First-In-First-out Delivery:** The order in which messages are sent and received is strictly preserved
At-Least-Once Delivery: A message is delivered at least once, but occasionally more than one copy of a message is delivered	**Exactly-Once Processing:** A message is delivered once and remains available until a consumer processes and deletes it. Duplicates are not introduced into the queue

© Digital Cloud Training | https://digitalcloud.training

Amazon SQS Queue Types

- FIFO queues require the **Message Group ID** and **Message Deduplication ID** parameters to be added to messages
- **Message Group ID** (MessageGroupId):
  - The tag that specifies that a message belongs to a specific message group
  - Messages that belong to the same message group are guaranteed to be processed in a FIFO manner
  - Messages with a different Group ID may be received out of order
- **Message Deduplication ID** (MessageDeduplicationId)
  - The token used for deduplication of messages within the deduplication interval
  - The de-duplication interval is 5 minutes
  - Generated as the SHA-256 with the message body

© Digital Cloud Training | https://digitalcloud.training

Amazon SQS Dead Letter Queue

Message not processed successfully (**ReceiveCount** exceeds **maxReceiveCount** for queue)

Dead-letter queue is a **standard** or **FIFO** queue that has been specified as a **dead-letter queue**

© Digital Cloud Training | https://digitalcloud.training

Amazon SQS Dead Letter Queue

- The main task of a dead-letter queue is handling message failure
- A dead-letter queue lets you set aside and isolate messages that can't be processed correctly to determine why their processing didn't succeed
- It is not a queue type, it is a **standard** or **FIFO** queue that has been specified as a dead-letter queue in the configuration of **another** standard or FIFO queue

Enable Redrive Policy

Specify the queue to use as a dead-letter queue

Specify the maximum receives before a message is sent to the dead-letter queue

© Digital Cloud Training | https://digitalcloud.training

Amazon SQS Dead Letter Queue

- Messages are moved to the dead-letter queue when the ReceiveCount for a message exceeds the maxReceiveCount for a queue
- Dead-letter queues should not be used with standard queues when your application will keep retrying transmission
- Dead-letter queues will break the order of messages in FIFO queues

© Digital Cloud Training | https://digitalcloud.training

Amazon SQS Delay Queue

© Digital Cloud Training | https://digitalcloud.training

Amazon SQS Delay Queue

When to use a delay queue:

- Large distributed applications which may need to introduce a delay in processing
- You need to apply a delay to an entire queue of messages
- For example, adding a delay of a few seconds to allow updates to sales or stock control databases before sending a notification to a customer confirming an online transaction

© Digital Cloud Training | https://digitalcloud.training

Amazon SQS Visibility Timeout

© Digital Cloud Training | https://digitalcloud.training

Amazon SQS Visibility Timeout

- The amount of time a message is invisible in the queue after a reader picks it up
- Provided the job is processed before the visibility timeout expires, the message will then be deleted from the queue
- If the job is not processed within the visibility timeout, the message will become visible again and another reader will process it
- This could result in the same message being delivered twice
- Default visibility timeout is 30 seconds
- Maximum is 12 hours

The length of time (in seconds) that a message received from a queue will be **invisible** to other receiving components

© Digital Cloud Training | https://digitalcloud.training

SQS Long Polling vs Short Polling

© Digital Cloud Training | https://digitalcloud.training

SQS Long Polling vs Short Polling

- SQS **Long polling** is a way to retrieve messages from SQS queues – waits for messages to arrive
- SQS **Short polling** returns immediately (even if the message queue is empty)
- SQS Long polling can lower costs
- SQS Long polling can be enabled at the queue level or at the API level using WaitTimeSeconds
- SQS Long polling is in effect when the Receive Message Wait Time is a value greater than 0 seconds and up to 20 seconds

The maximum amount of time that a long polling receive call will wait for a message to become available before returning an empty response.

© Digital Cloud Training | https://digitalcloud.training

Working with SQS Queues

© Digital Cloud Training | https://digitalcloud.training

Amazon SQS API and Client Library

© Digital Cloud Training | https://digitalcloud.training

Amazon SQS API

ChangeMessageVisibility

- Changes the visibility timeout of a specified message in a queue to a new value
- Default is 30 seconds; minimum is 0 seconds; maximum is 12 hours

GetQueueAttributes and SetQueueAttributes

- Gets/sets attributes for the specified queue
- Lots of possible values and recommend reviewing them in the AWS API reference
- Key attributes for the exam:
  - DelaySeconds – Configures a delay queue. 0/s to 900/s (default 0/s)
  - ReceiveMessageWaitTimeSeconds – Sets short/long polling. 0/s to 20/s (default 0s)
  - VisibilityTimeout - The visibility timeout for the queue. 0/ to 43,200/s (12 hours) (default 30/s)

© Digital Cloud Training | https://digitalcloud.training

Amazon SQS API

ReceiveMessage

- Retrieves one or more messages (up to 10), from the specified queue
- Using the WaitTimeSeconds parameter enables long-poll support

SendMessage

- DelaySeconds parameter delays a message
- MessageDeduplicationId parameter adds a deduplication ID (FIFO only)
- MessageGroupId parameter adds a tag for a message group (FIFO only)

© Digital Cloud Training | https://digitalcloud.training

Amazon SQS Extended Client Library

Producer uses the **SQS Extended Client Library for Java** to send a message with a reference to an S3 object

Consumer receives message with reference to **S3 object**

© Digital Cloud Training | https://digitalcloud.training

Amazon SQS Extended Client Library

- Maximum messages size in SQS is **256 KB**
- You can use Amazon S3 and the **Amazon SQS Extended Client Library for Java** to manage Amazon SQS messages
- Useful for storing and consuming messages up to **2 GB** in size
- You can use the Amazon SQS Extended Client Library for Java library to do the following:
  - Specify whether messages are always stored in Amazon S3 or only when the size of a message exceeds 256 KB
  - Send a message that references a single message object stored in an Amazon S3 bucket
  - Get the corresponding message object from an Amazon S3 bucket
  - Delete the corresponding message object from an Amazon S3 bucket

© Digital Cloud Training | https://digitalcloud.training

Amazon SNS

© Digital Cloud Training | https://digitalcloud.training

Amazon SNS Notifications

© Digital Cloud Training | https://digitalcloud.training

Amazon SNS

- Amazon SNS is a highly available, durable, secure, fully managed pub/sub messaging service
- Amazon SNS provides topics for high-throughput, push-based, many-to-many messaging
- Publisher systems can fan out messages to a large number of subscriber endpoints:
- Endpoints include:
  - Amazon SQS queues
  - AWS Lambda functions
  - HTTP/S webhooks
  - Mobile push
  - SMS
  - Email

© Digital Cloud Training | https://digitalcloud.training

Amazon SNS

- Multiple recipients can be grouped using Topics
- A topic is an “access point” for allowing recipients to dynamically subscribe for identical copies of the same notification
- One topic can support deliveries to multiple endpoint types
- Simple APIs and easy integration with applications
- Flexible message delivery over multiple transport protocols

© Digital Cloud Training | https://digitalcloud.training

Amazon SNS + Amazon SQS Fan-Out

- You can subscribe one or more Amazon SQS queues to an Amazon SNS topic
- Amazon SQS manages the subscription and any necessary permissions
- When you publish a message to a topic, Amazon SNS sends the message to every subscribed queue

© Digital Cloud Training | https://digitalcloud.training

Simple Serverless Application

© Digital Cloud Training | https://digitalcloud.training

Simple Serverless Application

© Digital Cloud Training | https://digitalcloud.training

AWS Step Functions

© Digital Cloud Training | https://digitalcloud.training

AWS Step Functions

© Digital Cloud Training | https://digitalcloud.training

AWS Step Functions

- AWS Step Functions is used to build distributed applications as a series of steps in a visual workflow.
- You can quickly build and run state machines to execute the steps of your application

How it works:

1. Define the steps of your workflow in the **JSON-based Amazon States Language**. The visual console automatically graphs each step in the order of execution
2. Start an execution to visualize and verify the steps of your application are operating as intended. The console highlights the real-time status of each step and provides a detailed history of every execution
3. AWS Step Functions **operates and scales** the steps of your **application** and **underlying compute** for you to help ensure your application executes reliably under increasing demand

© Digital Cloud Training | https://digitalcloud.training

Create a State Machine

© Digital Cloud Training | https://digitalcloud.training

Amazon EventBridge

© Digital Cloud Training | https://digitalcloud.training

Amazon EventBridge

EventBridge used to be known as **CloudWatch Events**

© Digital Cloud Training | https://digitalcloud.training

Amazon EventBridge Example 1

© Digital Cloud Training | https://digitalcloud.training

Amazon EventBridge Example 1

Event matching pattern

You can use pre-defined pattern provided by a service or create a custom pattern

- (●) Pre-defined pattern by service
- ( ) Custom pattern

Service provider

AWS services or custom/partner services

AWS

Service name

The name of partner service selected as the event source

EC2

Event type

The type of events as the source of the matching pattern

EC2 Instance State-change Notification

- ( ) Any state
- (●) Specific state(s)

terminated ×

- ( ) Any instance
- (●) Specific instance Id(s)

i-1234567890abcdef0 Remove

Add

Event pattern Copy Edit

```
1 {
2   "source": ["aws.ec2"],
3   "detail-type": ["EC2 Instance State-change N
4   "detail": {
5     "state": ["terminated"],
6     "instance-id": ["i-1234567890abcdef0"]
7   }
8 }
```


```
{
   "version": "0",
   "id": "6a7e8feb-b491-4cf7-a9f1-bf3703467718",
   "detail-type": "EC2 Instance State-change Notification",
   "source": "aws.ec2",
   "account": "111122223333",
   "time": "2017-12-22T18:43:48Z",
   "region": "us-west-1",
   "resources": [
      "arn:aws:ec2:us-west-1:123456789012:instance/i-1234567890abcdef0"
 ],
 "detail": {
   "instance-id": "i-1234567890abcdef0",
   "state": "terminated"
   }
 }
```


© Digital Cloud Training | https://digitalcloud.training

Amazon EventBridge Example 2

```
{
   "source": ["aws.cloudtrail"],
   "detail-type": ["AWS API Call via CloudTrail"],
   "detail": {
     "eventSource": ["cloudtrail.amazonaws.com"],
     "eventName": ["s3:PutBucketPolicy"]
   }
 }
```

© Digital Cloud Training | https://digitalcloud.training

Create Event Bus and Rule

© Digital Cloud Training | https://digitalcloud.training

Amazon EventBridge Example 1

Event matching pattern

You can use pre-defined pattern provided by a service or create a custom pattern

- (●) Pre-defined pattern by service
- ( ) Custom pattern

Service provider

AWS services or custom/partner services

AWS

Service name

The name of partner service selected as the event source

EC2

Event type

The type of events as the source of the matching pattern

EC2 Instance State-change Notification

- ( ) Any state
- (●) Specific state(s)

terminated ×

- ( ) Any instance
- (●) Specific instance Id(s)

i-1234567890abcdef0 Remove

Add

Event pattern Copy Edit

```
1 {
2   "source": ["aws.ec2"],
3   "detail-type": ["EC2 Instance State-change N
4   "detail": {
5     "state": ["terminated"],
6     "instance-id": ["i-1234567890abcdef0"]
7   }
8 }
```


```
{
   "version": "0",
   "id": "6a7e8feb-b491-4cf7-a9f1-bf3703467718",
   "detail-type": "EC2 Instance State-change Notification",
   "source": "aws.ec2",
   "account": "111122223333",
   "time": "2017-12-22T18:43:48Z",
   "region": "us-west-1",
   "resources": [
      "arn:aws:ec2:us-west-1:123456789012:instance/i-1234567890abcdef0"
 ],
 "detail": {
   "instance-id": "i-1234567890abcdef0",
   "state": "terminated"
   }
 }
```


© Digital Cloud Training | https://digitalcloud.training

Amazon API Gateway

© Digital Cloud Training | https://digitalcloud.training

Amazon API Gateway Overview

© Digital Cloud Training | https://digitalcloud.training

Amazon API Gateway API Types

Amazon API Gateway supports:

- **REST APIs** - support OIDC and OAuth 2.0 authorization, and come with built-in support for CORS and automatic deployments
- **HTTP APIs** - designed for low-latency, cost-effective integrations with AWS services, including AWS Lambda, and HTTP endpoints
- **WebSocket APIs** – deployed as a stateful frontend for an AWS service (such as Lambda or DynamoDB) or for an HTTP endpoint
- REST APIs and HTTP APIs support authorizers for AWS Lambda, IAM, and Amazon Cognito
- WebSocket APIs support IAM authorization and Lambda authorizers

© Digital Cloud Training | https://digitalcloud.training

Amazon API Gateway Deployment Types

Edge-optimized endpoint

Key benefits:

- Reduced latency for requests from around the world

Regional endpoint

Key benefits:

- Reduced latency for requests that originate in the same region
- Can also configure your own CDN and protect with WAF

Private endpoint

Key benefits:

- Securely expose your REST APIs only to other services within your VPC or connect via Direct Connect

© Digital Cloud Training | https://digitalcloud.training

Edge-Optimized APIs

- An edge-optimized API endpoint is best for geographically distributed clients
- API requests are routed to the nearest CloudFront Point of Presence (POP)
- This is the default endpoint type for API Gateway REST APIs
- Edge-optimized APIs capitalize the names of HTTP headers

© Digital Cloud Training | https://digitalcloud.training

Regional APIs

- A regional API endpoint is intended for clients in the same region
- Reduces connection overhead for connections from the same Region
- Any custom domain name that you use is specific to the region where the API is deployed
- If you deploy a regional API in multiple regions, it can have the same custom domain name in all regions

© Digital Cloud Training | https://digitalcloud.training

API Gateway Private REST APIs

- Private REST APIs can only be accessed from within a VPC using an interface VPC endpoint

© Digital Cloud Training | https://digitalcloud.training

Amazon API Gateway Features

API Gateway Feature	Benefit
Support for RESTful APIs and WebSocket APIs	With API Gateway, you can create RESTful APIs using either HTTP APIs or REST APIs
Private integrations with AWS ELB & AWS Cloud Map	Route requests to private resources in your VPC. Using HTTP APIs, you can build APIs for services behind private ALBs, private NLBs, and IP-based services registered in AWS Cloud Map, such as ECS tasks
Metering	Define plans that meter and restrict third-party developer access to APIs
Security	API Gateway provides multiple tools to authorize access to APIs and control service operation access
Resiliency	Manage traffic with throttling so that backend operations can withstand traffic spikes
Operations Monitoring	API Gateway provides a metrics dashboard to monitor calls to services
Lifecycle Management	Operate multiple API versions and multiple stages for each version simultaneously so that existing applications can continue to call previous versions after new API versions are published
AWS Authorization	Support for signature version 4 for REST APIs and WebSocket APIs, IAM access policies, and authorization with bearer tokens (e.g. JWT, SAML) using Lambda functions

DigitalCloud TRAINING

Methods, Integrations, and Mapping Templates

© Digital Cloud Training | https://digitalcloud.training

Methods and Resources

- A **resource** represents a path in your API
- **Methods** are created within resources
- A method represents a client-facing interface by which the client calls the API to access back-end resources
- A method is created for a specific HTTP verb, or you can use ANY

PetStore API

© Digital Cloud Training | https://digitalcloud.training

Methods and Integrations

© Digital Cloud Training | https://digitalcloud.training

Method Requests / Responses

- A Method resource is integrated with an Integration resource
- **Method request:**
  - The public interface of a REST API method in API Gateway that defines the parameters and body that an app developer must send in requests to access the backend through the API
- **Method response:**
  - The public interface of a REST API that defines the status codes, headers, and body models that an app developer should expect in responses from the API

© Digital Cloud Training | https://digitalcloud.training

Integration Requests / Responses

- API methods are integrated with backend endpoints using API integrations
- Backend endpoints are known as “integration endpoints”
- **Integration request:**
- The internal interface of a WebSocket API route or REST API method in API Gateway
- Maps the body of a route request or the parameters and body of a method request to the formats required by the backend
- **Integration response:**
- The internal interface of a WebSocket API route or REST API method in API Gateway
- Maps the status codes, headers, and payload that are received from the backend to the response format that is returned to a client app

© Digital Cloud Training | https://digitalcloud.training

Integration Types

- You choose an API integration type according to the types of integration endpoint you work with and how you want data to pass to and from the integration endpoint
- For a Lambda function, you can have the **Lambda proxy integration**, or the **Lambda custom integration**
- For an HTTP endpoint, you can have the **HTTP proxy integration** or the **HTTP custom integration**
- For an AWS service action, you have the AWS integration of the **non-proxy** type only
- API Gateway also supports the mock integration, where API Gateway serves as an integration endpoint to respond to a method request

© Digital Cloud Training | https://digitalcloud.training

Integration Types

AWS:

- This type of integration lets an API expose AWS service actions
- Must configure both the integration request and integration response and set up necessary data mappings from the method request to the integration request, and from the integration response to the method response

© Digital Cloud Training | https://digitalcloud.training

Integration Types

AWS_PROXY:

- This integration relies on direct interactions between the client and the integrated Lambda function
- With this type of integration, also known as the Lambda proxy integration, you do not set the integration request or the integration response
- API Gateway passes the incoming request from the client as the input to the backend Lambda function
- The integrated Lambda function takes the input of this format and parses the input from all available sources, including request headers, URL path variables, query string parameters, and applicable body
- The function returns the result following this output format

© Digital Cloud Training | https://digitalcloud.training

Integration Types

AWS_PROXY:

© Digital Cloud Training | https://digitalcloud.training

Integration Types

HTTP:

- This type of integration lets an API expose HTTP endpoints in the backend
- With the HTTP integration, also known as the HTTP custom integration, you must configure both the integration request and integration response
- You must set up necessary data mappings from the method request to the integration request, and from the integration response to the method response

© Digital Cloud Training | https://digitalcloud.training

Integration Types

MOCK:

- This type of integration lets API Gateway return a response without sending the request further to the backend
- Used to test the integration set up without incurring charges for using the backend and to enable collaborative development of an API
- A team can isolate their development effort by setting up simulations of API components owned by other teams by using the MOCK integrations

© Digital Cloud Training | https://digitalcloud.training

Mapping Templates

- An API's method request can take a payload in a different format from the corresponding integration request payload, as required in the backend
- Similarly, the backend may return an integration response payload different from the method response payload, as expected by the frontend
- API Gateway lets you use **mapping templates** to map the payload from a method request to the corresponding integration request and from an integration response to the corresponding method response
- A mapping template is a script expressed in Velocity Template Language (VTL) and applied to the payload using JSONPath expressions

© Digital Cloud Training | https://digitalcloud.training

Mapping Templates

© Digital Cloud Training | https://digitalcloud.training

API Gateway Stages and Deployments

© Digital Cloud Training | https://digitalcloud.training

API Gateway Stages and Deployments

- **Deployments** are a snapshot of the APIs resources and methods
- Deployments must be created and associated with a **stage** for anyone to access the API
- A stage is a logical reference to a lifecycle state of your REST or WebSocket API (e.g. ‘dev’, ‘prod’, ‘beta’, ‘v2’)
- API stages are identified by API ID and stage name
- Stage variables are like environment variables for API Gateway
  - Stage variables can be used in:
  - Lambda function ARN
  - HTTP endpoint
  - Parameter mapping templates

© Digital Cloud Training | https://digitalcloud.training

API Gateway Stages and Deployments

© Digital Cloud Training | https://digitalcloud.training

API Gateway Stages and Deployments

Use cases for stage variables:

- Configure HTTP endpoints your stages talk to (dev, test, prod etc.)
- Pass configuration parameters to AWS Lambda through mapping templates
- Stage variables are passed to the “context” object in Lambda
- Stage variables are used with Lambda aliases
- You can create a stage variable to indicate the corresponding Lambda alias
- You can create canary deployments for any stage and choose the % of traffic the canary channel receives

© Digital Cloud Training | https://digitalcloud.training

Swagger and Open API 3

- You can import existing **Swagger** / **Open API 3.0** definitions (written in YAML or JSON) to API Gateway
- This is a common way of defining REST APIs using API definition as code
- Can also export current APIs as Swagger / Open API 3.0 definition
- Uses the API Gateway Import API feature to import an API from an external definition
- You specify the options using a mode query parameter in the request URL

© Digital Cloud Training | https://digitalcloud.training

Swagger and Open API 3

Example of a template for a **Swagger 2.0 API**

```
{
  "swagger": "2.0",
  "info": {
    "version": "2020-03-15T07:45:21Z",
    "title": "MyLambdaAPI"
  },
  "host": "a90w7s09q6.execute-api.ap-southeast-2.amazonaws.com",
  "basePath": "/development",
  "schemes": [
    "https"
  ],
  "paths": {
    "/": {
      "get": {
        "produces": [
          "application/json"
        ],
        "responses": {
          "200": {
            "description": "200 response",
            "schema": {
              "$ref": "#/definitions/Empty"
            }
          }
        }
      }
```

© Digital Cloud Training | https://digitalcloud.training

Build an API with Lambda Proxy Integration

© Digital Cloud Training | https://digitalcloud.training

API Gateway
Caching and Throttling

© Digital Cloud Training | https://digitalcloud.training

API Gateway - Caching

- You can add caching to API calls by provisioning an Amazon API Gateway cache and specifying its size in gigabytes
- Caching allows you to cache the endpoint's response
- Caching can reduce number of calls to the backend and improve latency of requests to the API

© Digital Cloud Training | https://digitalcloud.training

API Gateway Caching

- API Gateway caches responses for a specific amount of time (time to live or TTL)
- The default TTL is 300 seconds (min 0, max 3600)
- Caches are defined per stage
- You can encrypt caches
- The cache capacity is between 0.5GB to 237GB
- It is possible to override cache settings for specific methods
- You can flush the entire cache (invalidate it) immediately if required
- Clients can invalidate the cache with the header: Cache-Control: max-age=0

© Digital Cloud Training | https://digitalcloud.training

API Gateway - Throttling

- API Gateway sets a limit on a steady-state rate and a burst of request submissions against all APIs in your account
- Limits:
  - By default, API Gateway limits the steady-state request rate to 10,000 requests per second
  - The maximum concurrent requests is 5,000 requests across all APIs within an AWS account
  - If you go over 10,000 requests per second or 5,000 concurrent requests, you will receive a 429 Too Many Requests error response
- Upon catching such exceptions, the client can resubmit the failed requests in a way that is rate limiting, while complying with the API Gateway throttling limits

© Digital Cloud Training | https://digitalcloud.training

API Gateway Throttling

- Amazon API Gateway provides two basic types of throttling-related settings:
- **Server-side** throttling limits are applied across all clients. These limit settings exist to prevent your API—and your account—from being overwhelmed by too many requests
- **Per-client** throttling limits are applied to clients that use API keys associated with your usage policy as client identifier
- API Gateway throttling-related settings are applied in the following order:
  - Per-client per-method limits that you set for an API stage in a usage plan
  - Per-client limits that you set in a usage plan
  - Default per-method limits and individual per-method limits that you set in API stage settings
  - Account-level throttling

© Digital Cloud Training | https://digitalcloud.training

Configure Caching and Throttling

© Digital Cloud Training | https://digitalcloud.training

API Gateway
Usage Plans and API Keys

© Digital Cloud Training | https://digitalcloud.training

Usage Plans and API Keys

Users connect to **specific public endpoint** with **API key** that is configured in a usage plan

Can also configure **per-method** throttling limits

© Digital Cloud Training | https://digitalcloud.training

Usage Plans and API Keys

- A usage plan specifies who can access one or more deployed API stages and methods — and how much and how fast they can access them
- You can use a usage plan to configure throttling and quota limits, which are enforced on individual client API keys
- The plan uses API keys to identify API clients and meters access to the associated API stages for each key
- It also lets you configure throttling limits and quota limits that are enforced on individual client API keys
- You can use API keys together with usage plans or Lambda authorizers to control access to your APIs

© Digital Cloud Training | https://digitalcloud.training

Usage Plans and API Keys

© Digital Cloud Training | https://digitalcloud.training

Microservice with API, Lambda, and DynamoDB

© Digital Cloud Training | https://digitalcloud.training

Microservice with Lambda, API Gateway and DynamoDB

© Digital Cloud Training | https://digitalcloud.training

API Gateway Access Control

© Digital Cloud Training | https://digitalcloud.training

API Gateway Access Control

There are several mechanisms for controlling and managing access to an API:

- Resource-based policies
- Identity-based policies
- IAM Tags
- Endpoint policies for interface VPC endpoints
- Lambda authorizers
- Amazon Cognito user pools

© Digital Cloud Training | https://digitalcloud.training

Resource Based Policies

- Amazon API Gateway resource policies are JSON policy documents that you attach to an API
- Control whether a specified principal can invoke the API

© Digital Cloud Training | https://digitalcloud.training

IAM Permissions

```
{
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Permission",
        "Action": [
          "execute-api:Execution-operation"
        ],
        "Resource": [
          "arn:aws:execute-api:region:account-id:api-id/stage/METHOD_HTTP_VERB/Resource-path"
        ]
      }
    ]
  }
```

Allow or deny

Supported operations

ARN of deployed API

© Digital Cloud Training | https://digitalcloud.training

Lambda Authorizer

- A Lambda authorizer is an API Gateway feature that uses a Lambda function to control access to your API
- API Gateway calls the Lambda authorizer, which takes the caller's identity as input and returns an IAM policy as output
- There are two types of Lambda authorizers:
- A **token-based** Lambda authorizer receives the caller's identity in a bearer token, such as a JSON Web Token (JWT) or an OAuth token.
- A **request parameter-based** Lambda authorizer receives the caller's identity in a combination of headers, query string parameters, stageVariables, and $context variables
- For WebSocket APIs, only request parameter-based authorizers are supported

© Digital Cloud Training | https://digitalcloud.training

Lambda Authorizer

1. Client calls API method, passing bearer token or request parameters
2. API Gateway calls the Lambda authorizer
3. Lambda function authenticates the user:
   - Call OAuth provider to get token
   - Call SAML provider to get assertion
   - Generate IAM Policy based on request parameters
4. If successful, Lambda grants access and returns IAM policy and a principal identifier
5. API Gateway evaluates the policy:
   - If access is allowed, execute the method
   - If access is denied, return status code (e.g. 403)

© Digital Cloud Training | https://digitalcloud.training

Cognito User Pool Authorizer

- A user pool is a user directory in Amazon Cognito
- With a user pool, users can sign into a web or mobile app through Amazon Cognito
- Users can also sign in through social identity providers like Google, Facebook, Amazon, or Apple, and through SAML identity providers
- You can use an Amazon Cognito user pool to control who can access your API in Amazon API Gateway
- You create an authorizer of the COGNITO_USER_POOLS type and then configure an API method to use that authorizer

© Digital Cloud Training | https://digitalcloud.training

Cognito User Pool Authorizer

© Digital Cloud Training | https://digitalcloud.training

SECTION 10

Containers on Amazon ECS/EKS

© Digital Cloud Training | https://digitalcloud.training

Amazon Elastic Container Service (ECS)

© Digital Cloud Training | https://digitalcloud.training

Amazon ECS

```
{
"containerDefinitions": [
                {
                  "name": "wordpress",
                  "links": [
                    "mysql"
                ],
                "image": "wordpress",
                "essential": true,
                "portMappings": [
                  {
                    "containerPort": 80,
                    "hostPort": 80
                     }
                  ],
                  "memory": 500,
                  "cpu": 10
                }
```

© Digital Cloud Training | https://digitalcloud.training

Amazon ECS Key Features

- **Serverless with AWS Fargate** – managed for you and fully scalable
- **Fully managed container orchestration** – control plane is managed for you
- **Docker support** – run and manage Docker containers with integration into the Docker Compose CLI
- **Windows container support** – ECS supports management of Windows containers
- **Elastic Load Balancing integration** – distribute traffic across containers using ALB or NLB
- **Amazon ECS Anywhere (NEW)** – enables the use of Amazon ECS control plane to manage on-premises implementations

© Digital Cloud Training | https://digitalcloud.training

Amazon ECS Components

Elastic Container Service (ECS)	Description
Cluster	Logical grouping of tasks or services
Container instance	EC2 instance running the the ECS agent
Task Definition	Blueprint that describes how a docker container should launch
Task	A running container using settings in a Task Definition
Service	Defines long running tasks – can control task count with Auto Scaling and attach an ELB

© Digital Cloud Training | https://digitalcloud.training

Amazon ECS Clusters

- ECS Clusters are a logical grouping of container instances that you can place tasks on
- A default cluster is created but you can then create multiple clusters to separate resources
- ECS allows the definition of a specified number (desired count) of tasks to run in the cluster

© Digital Cloud Training | https://digitalcloud.training

Amazon ECS Clusters

- Clusters can contain tasks using the Fargate and EC2 launch type
- EC2 launch type clusters can contain different container instance types
- Each container instance may only be part of one cluster at a time
- Clusters are region specific
- You can create IAM policies for your clusters to allow or restrict users' access to specific clusters

© Digital Cloud Training | https://digitalcloud.training

ECS Container Instances and Container Agent

- You can use any AMI that meets the Amazon ECS AMI specification
- The EC2 instances used as container hosts must run an ECS agent
- The ECS container agent allows container instances to connect to the cluster
- The container agent runs on each infrastructure resource on an ECS cluster

© Digital Cloud Training | https://digitalcloud.training

Amazon ECS Images

- Containers are created from a read-only template called an **image** which has the instructions for creating a Docker container
- Images are built from a **Dockerfile**
- Only Docker containers are supported on ECS
- Images are stored in a registry such as DockerHub or Amazon Elastic Container Registry (ECR)
- ECR is a managed AWS Docker registry service that is secure, scalable and reliable
- ECR supports private Docker repositories with resource-based permissions using AWS IAM in order to access repositories and images
- You can use the Docker CLI to push, pull and manage images

© Digital Cloud Training | https://digitalcloud.training

Amazon ECS Tasks and Task Definitions

- A task definition is required to run Docker containers in Amazon ECS
- A task definition is a text file in JSON format that describes one or more containers, up to a maximum of 10
- Task definitions use Docker images to launch containers
- You specify the number of tasks to run (i.e. the number of containers)

Task Definition

```
{
"containerDefinitions": [
                {
                  "name": "wordpress",
                  "links": [
                    "mysql"
                ],
                "image": "wordpress",
                "essential": true,
                "portMappings": [
                  {
                    "containerPort": 80,
                    "hostPort": 80
                     }
                  ],
                  "memory": 500,
                  "cpu": 10
                }
```

© Digital Cloud Training | https://digitalcloud.training

Amazon ECS Tasks and Task Definitions

Some of the parameters you can specify in a task definition include:

- Which Docker images to use with the containers in your task
- How much CPU and memory to use with each container
- Whether containers are linked together in a task
- The Docker networking mode to use for the containers in your task
- What (if any) ports from the container are mapped to the host container instances
- Whether the task should continue if the container finished or fails
- The commands the container should run when it is started
- Environment variables that should be passed to the container when it starts
- Data volumes that should be used with the containers in the task
- IAM role the task should use for permissions

© Digital Cloud Training | https://digitalcloud.training

Amazon ECS Launch Types

© Digital Cloud Training | https://digitalcloud.training

Launch Types – EC2 and Fargate

Registry:
ECR, Docker Hub, Self-hosted

Registry:
ECR, Docker Hub

EC2 Launch Type

- You explicitly provision EC2 instances
- You're responsible for managing EC2 instances
- Charged per running EC2 instance
- EFS and EBS integration
- You handle cluster optimization
- More granular control over infrastructure

Fargate Launch Type

- Fargate automatically provisions resources
- Fargate provisions and manages compute
- Charged for running tasks
- No EBS integration
- Fargate handles cluster optimization
- Limited control, infrastructure is automated

© Digital Cloud Training | https://digitalcloud.training

Amazon ECS Launch Types

Fargate Launch Type

- Run containers without the need to provision and manage the backend infrastructure
- AWS Fargate is the serverless way to host your Amazon ECS workloads

EC2 Launch Type

- Run containers on a cluster of Amazon EC2 instances that you manage

External Launch Type

- Run containers on your on-premises servers or virtual machines (VMs) – uses Amazon ECS Anywhere

© Digital Cloud Training | https://digitalcloud.training

Amazon ECS Launch Types

Amazon EC2	Amazon Fargate
You explicitly provision EC2 instances	The control plane asks for resources and Fargate automatically provisions
You're responsible for upgrading, patching, care of EC2 pool	Fargate provisions compute as needed
You must handle cluster optimization	Fargate handles cluster optimization
More granular control over infrastructure	Limited control, as infrastructure is automated

© Digital Cloud Training | https://digitalcloud.training

Amazon ECS and IAM Roles

© Digital Cloud Training | https://digitalcloud.training

ECS and IAM Roles

AmazonEC2ContainerServiceforEC2Role

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeTags",
                "ecs:CreateCluster",
                "ecs:DeregisterContainerInstance",
                "ecs:DiscoverPollEndpoint",
                "ecs:Poll",
                "ecs:RegisterContainerInstance",
                "ecs:StartTelemetrySession",
                "ecs:UpdateContainerInstancesState",
                "ecs:Submit*",
                "ecr:GetAuthorizationToken",
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:BatchGetImage",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        }
    ]
}
```

NOTE: container instances have access to **all of the permissions** that are supplied to the **container instance role** through instance metadata

© Digital Cloud Training | https://digitalcloud.training

ECS and IAM Roles

Create an ECS Cluster with EC2 Launch Type

© Digital Cloud Training | https://digitalcloud.training

Launch Task and Service with ALB

© Digital Cloud Training | https://digitalcloud.training

ECS Task Placement Strategies

© Digital Cloud Training | https://digitalcloud.training

ECS Task Placement Strategies

- A task placement strategy is an algorithm for selecting instances for task placement or tasks for termination
- Task placement strategies can be specified when either running a task or creating a new service
- This is relevant only to the EC2 launch type
- Amazon ECS supports the following task placement strategies:

binpack - place tasks based on the least available amount of CPU or memory. This minimizes the number of instances in use

random - place tasks randomly

© Digital Cloud Training | https://digitalcloud.training

ECS Task Placement Strategies

spread - place tasks evenly based on the specified value

- Accepted values are instanceId or host (same effect)
- Or any platform or custom attribute that is applied to a container instance, such as attribute:ecs.availability-zone
- Service tasks are spread based on the tasks from that service
- Standalone tasks are spread based on the tasks from the same task group

© Digital Cloud Training | https://digitalcloud.training

ECS Task Placement Strategies

The following strategy distributes tasks evenly across **Availability Zones**:

```
"placementStrategy": [
    {
        "field": "attribute:ecs.availability-zone",
        "type": "spread"
    }
]
```

© Digital Cloud Training | https://digitalcloud.training

ECS Task Placement Strategies

The following strategy distributes tasks evenly across **all instances**:

```
"placementStrategy": [
    {
        "field": "instanceId",
        "type": "spread"
    }
]
```

© Digital Cloud Training | https://digitalcloud.training

ECS Task Placement Strategies

The following strategy **bin packs** tasks based on **memory**:

```
"placementStrategy": [
    {
        "field": "memory",
        "type": "binpack"
    }
]
```

© Digital Cloud Training | https://digitalcloud.training

ECS Task Placement Strategies

The following strategy distributes tasks evenly across **Availability Zones** and then **bin packs** tasks based on **memory** within each **Availability Zone**:

```
"placementStrategy": [
    {
        "field": "attribute:ecs.availability-zone",
        "type": "spread"
    },
    {
        "field": "memory",
        "type": "binpack"
    }
]
```

© Digital Cloud Training | https://digitalcloud.training

ECS Task Placement Strategies

- A task placement constraint is a rule that is considered during task placement
- Amazon ECS supports the following types of task placement constraints:
  - distinctInstance - Place each task on a different container instance
  - memberOf - Place tasks on container instances that satisfy an expression

© Digital Cloud Training | https://digitalcloud.training

Cluster Query Language

- Cluster queries are expressions that enable you to group objects
- For example, you can group container instances by attributes such as **Availability Zone**, **instance type**, or **custom metadata**
- Expressions have the following syntax: subject operator [argument]
- **Example 1:** The following expression selects instances with the specified instance type:

```
attribute:ecs.instance-type == t2.small
```

- **Example 2:** The following expression selects instances in the us-east-1a or us-east-1b Availability Zone:

```
attribute:ecs.availability-zone in [us-east-1a, us-east-1b]
```

© Digital Cloud Training | https://digitalcloud.training

Cluster Query Language

- **Example 3:** The following expression selects instances that are hosting tasks in the service:production group:

```
task:group == service:production
```

© Digital Cloud Training | https://digitalcloud.training

Scaling Amazon ECS

© Digital Cloud Training | https://digitalcloud.training

Auto Scaling for ECS

Two types of scaling:

1. Service auto scaling
2. Cluster auto scaling

- **Service auto scaling** automatically adjusts the desired task count up or down using the Application Auto Scaling service
- **Service auto scaling** supports target tracking, step, and scheduled scaling policies
- **Cluster auto scaling** uses a Capacity Provider to scale the number of EC2 cluster instances using EC2 Auto Scaling

© Digital Cloud Training | https://digitalcloud.training

Service Auto Scaling

- 1. Metric reports CPU > 80%
- 2. CloudWatch notifies Application Auto Scaling
- 3. ECS launches additional task

© Digital Cloud Training | https://digitalcloud.training

Service Auto Scaling

- Amazon ECS Service Auto Scaling supports the following types of scaling policies:
  - **Target Tracking Scaling Policies**—Increase or decrease the number of tasks that your service runs based on a target value for a specific CloudWatch metric
  - **Step Scaling Policies**—Increase or decrease the number of tasks that your service runs in response to CloudWatch alarms. Step scaling is based on a set of scaling adjustments, known as step adjustments, which vary based on the size of the alarm breach
  - **Scheduled Scaling**—Increase or decrease the number of tasks that your service runs based on the date and time

© Digital Cloud Training | https://digitalcloud.training

Cluster Auto Scaling

- 1. Metric reports target capacity > 80%
- 2. CloudWatch notifies ASG
- 3. AWS launches additional container instance

A **Capacity provider reservation** metric measures the total percentage of cluster resources needed by all ECS workloads in the cluster

© Digital Cloud Training | https://digitalcloud.training

DigitalCloud TRAINING

Cluster Auto Scaling

- Uses an ECS resource type called a **Capacity Provider**
- A Capacity Provider can be associated with an EC2 **Auto Scaling Group** (ASG)
- ASG can automatically scale using:
  - **Managed scaling** - with an automatically-created scaling policy on your ASG
  - **Managed instance termination protection** - which enables container-aware termination of instances in the ASG when scale-in happens

© Digital Cloud Training | https://digitalcloud.training

Amazon ECS with ALB

© Digital Cloud Training | https://digitalcloud.training

Amazon ECS with ALB

© Digital Cloud Training | https://digitalcloud.training

Amazon Elastic Container Registry (ECR)

© Digital Cloud Training | https://digitalcloud.training

Amazon Elastic Container Registry (ECR)

- Amazon ECR is a fully-managed container registry
- Integrated with Amazon ECS and Amazon EKS
- Supports Open Container Initiative (OCI) and Docker Registry HTTP API V2 standards
- You can use Docker tools and Docker CLI commands such as push, pull, list, and tag
- Can be accessed from any Docker environment – in the cloud, on-premises, or on you machine

© Digital Cloud Training | https://digitalcloud.training

Amazon Elastic Container Registry (ECR)

- Container images and artifacts are stored in S3
- You can use namespaces to organize repositories
- Public repositories allow everyone to access container images
- Access control applies to private repositories:
  - **IAM access control** - Set policies to define access to container images in private repositories
  - **Resource-based policies** - Access control down to the individual API action such as create, list, describe, delete, and get

© Digital Cloud Training | https://digitalcloud.training

Amazon ECR Components

ECR Component	Description
Registry	An Amazon ECR private registry is provided to each AWS account; you can create one or more repositories in your registry and store images in them
Authorization token	Your client must authenticate to Amazon ECR registries as an AWS user before it can push and pull images
Repository	An Amazon ECR repository contains your Docker images, OCI images, and OCI compatible artifacts
Repository policy	You can control access to your repositories and the images within them with repository policies
Image	You can push and pull container images to your repositories

© Digital Cloud Training | https://digitalcloud.training

Amazon Elastic Container Registry (ECR)

- **Lifecycle policies** - manage the lifecycle of the images in your repositories
- **Image scanning** - identify software vulnerabilities in your container images
- **Cross-Region and cross-account replication** – replicate images across accounts/Region
- **Pull through cache rules** - cache repositories in remote public registries in your private Amazon ECR registry

© Digital Cloud Training | https://digitalcloud.training

Amazon Elastic Container Registry (ECR)

© Digital Cloud Training | https://digitalcloud.training

Pushing an Image to a Private Repository

- Users must have the following IAM permissions:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:CompleteLayerUpload",
                "ecr:GetAuthorizationToken",
                "ecr:UploadLayerPart",
                "ecr:InitiateLayerUpload",
                "ecr:BatchCheckLayerAvailability",
                "ecr:PutImage"
            ],
            "Resource": "*"
        }
    ]
}
```

The **Resource** element can also be used to scope to a specific repository ARN

© Digital Cloud Training | https://digitalcloud.training

Pushing an Image to a Private Repository

- First, authenticate the Docker client to ECR:

```
aws ecr get-login-password --region region | docker login --username AWS --password-stdin
aws_account_id.dkr.ecr.region.amazonaws.com
```

- Tag your image with the Amazon ECR registry, repository, and image tag name to use:

```
docker tag e9ae3c220b23 aws_account_id.dkr.ecr.region.amazonaws.com/my-repository:tag
```

- Push the image using the docker push command:

```
docker push aws_account_id.dkr.ecr.region.amazonaws.com/my-repository:tag
```

© Digital Cloud Training | https://digitalcloud.training

ECS-LAB-1 - Create Image and Push to ECR Repository

© Digital Cloud Training | https://digitalcloud.training

ECS Lab Part 1

1. **Create Image and Push to ECR Repository**
2. Create Task Definition and ALB
3. Create Fargate Cluster and Service
4. CodeDeploy Application and Pipeline
5. Implement Blue/Green Update to ECS

© Digital Cloud Training | https://digitalcloud.training

ECS Lab Part 1

© Digital Cloud Training | https://digitalcloud.training

ECS-LAB-2 - Create Task Definition and ALB

© Digital Cloud Training | https://digitalcloud.training

ECS Lab Part 2

1. Create Image and Push to ECR Repository
2. **Create Task Definition and ALB**
3. Create Fargate Cluster and Service
4. CodeDeploy Application and Pipeline
5. Implement Blue/Green Update to ECS

© Digital Cloud Training | https://digitalcloud.training

ECS Lab Part 2

© Digital Cloud Training | https://digitalcloud.training

ECS-LAB-3 - Create Fargate Cluster and Service

© Digital Cloud Training | https://digitalcloud.training

ECS Lab Part 3

1. Create Image and Push to ECR Repository
2. Create Task Definition and ALB
3. **Create Fargate Cluster and Service**
4. CodeDeploy Application and Pipeline
5. Implement Blue/Green Update to ECS

© Digital Cloud Training | https://digitalcloud.training

ECS Lab Part 3

© Digital Cloud Training | https://digitalcloud.training

Amazon EKS

© Digital Cloud Training | https://digitalcloud.training

Amazon EKS

- Amazon Elastic Kubernetes Service (Amazon EKS) is a managed service
- Use when you need to **standardize** container orchestration across multiple environments using a **managed Kubernetes** implementation
- **Hybrid Deployment** - manage Kubernetes clusters and applications across hybrid environments (AWS + On-premises)
- **Batch Processing** - run sequential or parallel batch workloads on your EKS cluster using the Kubernetes Jobs API. Plan, schedule and execute batch workloads
- **Machine Learning** - use Kubeflow with EKS to model your machine learning workflows and efficiently run distributed training jobs using the latest EC2 GPU-powered instances, including Inferentia
- **Web Applications** - build web applications that automatically scale up and down and run in a highly available configuration across multiple Availability Zones

© Digital Cloud Training | https://digitalcloud.training

Amazon EKS

© Digital Cloud Training | https://digitalcloud.training

Amazon EKS Auto Scaling

Cluster Auto Scaling:

- **Vertical Pod Autoscaler** - automatically adjusts the CPU and memory reservations for your pods to help "right size" your applications
- **Horizontal Pod Autoscaler** - automatically scales the number of pods in a deployment, replication controller, or replica set based on that resource's CPU utilization

Workload Auto Scaling:

- Amazon EKS supports two **autoscaling** products:
  - Kubernetes Cluster Autoscaler
  - Karpenter open source autoscaling project.
- The cluster autoscaler uses AWS scaling groups, while Karpenter works directly with the Amazon EC2 fleet

© Digital Cloud Training | https://digitalcloud.training

Amazon EKS Pod Networking

- Amazon EKS supports native VPC networking with the Amazon VPC Container Network Interface (CNI) plugin for Kubernetes
- This plugin assigns a private IPv4 or IPv6 address from your VPC to each pod
- The VPC CNI plugin for Kubernetes is deployed with each of your Amazon EC2 nodes in a Daemonset with the name aws-node
- The plugin consists of two components:
  - **L-IPAM daemon** – Responsible for creating network interfaces and attaching the network interfaces to Amazon EC2 instances, assigning secondary IP addresses to network interfaces, and maintaining a warm pool of IP addresses on each node for assignment to Kubernetes pods when they are scheduled
  - **CNI plugin** – Responsible for wiring the host network (for example, configuring the network interfaces and virtual Ethernet pairs) and adding the correct network interface to the pod namespace

© Digital Cloud Training | https://digitalcloud.training

Amazon EKS and Elastic Load Balancing

- Amazon EKS supports Network Load Balancers and Application Load Balancers
- The AWS Load Balancer Controller manages AWS Elastic Load Balancers for a Kubernetes cluster
- Install the AWS Load Balancer Controller using Helm V3 or later or by applying a Kubernetes manifest
- The controller provisions the following resources:
  - An AWS Application Load Balancer (ALB) when you create a Kubernetes Ingress
  - An AWS Network Load Balancer (NLB) when you create a Kubernetes service of type LoadBalancer
- In the past, the Kubernetes network load balancer was used for instance targets, but the AWS Load balancer Controller was used for IP targets
- With the AWS Load Balancer Controller version 2.3.0 or later, you can create NLBs using either target type

DigitalCloud TRAINING

AWS Copilot

© Digital Cloud Training | https://digitalcloud.training

AWS Copilot

- Command line interface for launching and managing containers

AWS Copilot

- Used with ECS, Fargate, and AWS AppRunner
- Based on Infrastructure as Code (IaC) templates
- Simple commands to build containerized environments

```
# Build a copilot environment
copilot init
# Show information about environments and services
copilot app show
# Show information about environments
copilot env ls
# List of all the services in an application
copilot svc ls
# Show service status
copilot svc status
```

© Digital Cloud Training | https://digitalcloud.training

SECTION 11

AWS Developer Tools (CI/CD)

© Digital Cloud Training | https://digitalcloud.training

CI/CD Overview

© Digital Cloud Training | https://digitalcloud.training

Continuous Integration

Continuous Integration and Continuous Delivery

© Digital Cloud Training | https://digitalcloud.training

Continuous Integration and Continuous Delivery

CODE	BUILD & TEST	DEPLOY
	AWS CodePipeline	
AWS CodeCommit	AWS CodeBuild	AWS CodeDeploy
GitHub	Jenkins	ANSIBLE

© Digital Cloud Training | https://digitalcloud.training

AWS CodeCommit and CodePipeline

© Digital Cloud Training | https://digitalcloud.training

AWS CodeCommit

- AWS CodeCommit is a fully-managed **source control** service that hosts secure Git-based repositories
- Git is an Open-Source distributed source control system:
  - Centralized repository for all of your code, binaries, images, and libraries
  - Tracks and manages code changes
  - Maintains version history
  - Manages updates from multiple sources
  - Enables collaboration

© Digital Cloud Training | https://digitalcloud.training

AWS CodeCommit

- CodeCommit repositories are private
- CodeCommit scales seamlessly
- CodeCommit is integrated with Jenkins, CodeBuild and other CI tools
- You can transfer your files to and from AWS CodeCommit using HTTPS or SSH
- Repositories are automatically encrypted at rest through AWS Key Management Service (AWS KMS) using customer-specific keys

© Digital Cloud Training | https://digitalcloud.training

AWS CodeCommit

- You need to configure your Git client to communicate with CodeCommit repositories
- IAM supports CodeCommit with three types of credentials:
  - **Git credentials** - an IAM-generated user name and password pair you can use to communicate with CodeCommit repositories over HTTPS
  - **SSH keys** - a locally generated public-private key pair that you can associate with your IAM user to communicate with CodeCommit repositories over SSH
  - **AWS access keys** - which you can use with the credential helper included with the AWS CLI to communicate with CodeCommit repositories over HTTPS

AWS CodePipeline

- Fully managed continuous delivery service that helps you automate your release pipelines for fast and reliable application and infrastructure updates
- Automates the build, test, and deploy phases of your release process every time there is a code change, based on the release model you define
- CodePipeline provides tooling integrations for many AWS and third-party software at each stage of the pipeline including:
  - **Source stage** – S3, CodeCommit, Github, ECR, Bitbucket Cloud (beta).
  - **Build** – CodeBuild, Jenkins.
  - **Deploy stage** – CloudFormation, CodeDeploy, ECS, Elastic Beanstalk, AWS Service Catalog, S3.

© Digital Cloud Training | https://digitalcloud.training

AWS CodePipeline

Pipelines

- A workflow that describes how software changes go through the release process

Artifacts

- Files or changes that will be worked on by the actions and stages in the pipeline
- Each pipeline stage can create "artifacts"
- Artifacts are passed, stored in Amazon S3 and then passed on to the next stage

Stages

- Pipelines are broken up into stages
- Each stage can have sequential actions and or parallel actions
- Stage examples would be build, test, deploy, load test etc.
- Manual approval can be defined at any stage

DigitalCloud TRAINING

AWS CodePipeline

Actions

- Stages contain at least one action
- Actions affect artifacts and will have artifacts as either an input, and output, or both

Transitions

- The progressing from one stage to another inside of a pipeline

© Digital Cloud Training | https://digitalcloud.training

CodePipeline with Elastic Beanstalk

© Digital Cloud Training | https://digitalcloud.training

Install Git
(and Learn the Basics)

© Digital Cloud Training | https://digitalcloud.training

Create CodeCommit Repository

© Digital Cloud Training | https://digitalcloud.training

Create Pipeline and Application

© Digital Cloud Training | https://digitalcloud.training

CodePipeline with Elastic Beanstalk

© Digital Cloud Training | https://digitalcloud.training

Amazon CodeGuru

© Digital Cloud Training | https://digitalcloud.training

Amazon CodeGuru

- Provides intelligent recommendations for improving application performance, efficiency, and code quality
- Made up of two services:
- **Amazon CodeGuru Reviewer**
- **Amazon CodeGuru Profiler**

© Digital Cloud Training | https://digitalcloud.training

Amazon CodeGuru Reviewer

- Reviews Java and Python code and offers suggestions for improvement
- Suggestions are best on best practices
- Finds complex issues such as resource leak and security analysis
- Integrations with Secrets Manager to use a secrets detector that finds unprotected secrets in code
- Supports the following source providers:
  - AWS CodeCommit
  - Bitbucket
  - GitHub
  - GitHub Enterprise Cloud
  - GitHub Enterprise Server
  - Amazon S3

© Digital Cloud Training | https://digitalcloud.training

Amazon CodeGuru Profiler

- Collects runtime performance data from your live applications
- Provides recommendations that can help you fine-tune your application performance
- CodeGuru Profiler provides different visualizations to identify:
  - What code is running on the CPU
  - How much time is consumed
  - Ways to reduce CPU utilization
- Profiling includes:
  - Latency and CPU utilization issues in your application
  - Ways to reduce the infrastructure costs of running an application
  - Identify application performance issues
  - Understand your application's heap utilization over time

© Digital Cloud Training | https://digitalcloud.training

AWS CodeBuild

© Digital Cloud Training | https://digitalcloud.training

AWS CodeBuild

- AWS CodeBuild is a fully managed continuous integration (CI) service
- Compiles source code, runs tests, and produces software packages that are ready to deploy
- CodeBuild scales continuously and processes multiple builds concurrently
- You pay based on the time it takes to complete the builds
- CodeBuild takes source code from GitHub, CodeCommit, CodePipeline, S3 etc.
- Build instructions can be defined in the code (**buildspec.yml**)
- Output logs can be sent to Amazon S3 & Amazon CloudWatch Logs

© Digital Cloud Training | https://digitalcloud.training

AWS CodeBuild Components

- **Build project** - defines how CodeBuild will run a build defines settings including:
- Location of the source code
- The build environment to use
- The build commands to run
- Where to store the output of the build
- **Build environment** - the operating system, language runtime, and tools that CodeBuild uses for the build
- **Build Specification** - a YAML file that describes the collection of commands and settings for CodeBuild to run a build

© Digital Cloud Training | https://digitalcloud.training

AWS CodeBuild – buildspec.yml file

```yaml
version: 0.2

phases:
    install:
        commands:
            - echo "Entered the install phase..."
    pre_build:
        commands:
            - echo "Entered the pre_build phase..."
    build:
        commands:
            - echo "Entered the build phase..."
            - echo "Build started on `date`"
            - find production.txt
    post_build:
        commands:
            - echo "Entered the post_build phase..."
            - echo "Build completed on `date`"
```

© Digital Cloud Training | https://digitalcloud.training

Buildspec Phases

Phases	Required?	Description
phases/*/run-as	Optional	Specifies a Linux user that runs its commands
phases/*/on-failure	Optional	Specifies the action to take if failure occurs during the phase (ABORT or CONTINUE)
phases/*/finally	Optional	Runs commands after commands in the commands block (even if the the commands in the commands block fail)
phases/install	Optional	Commands to run during installation
phases/pre_build	Optional	Commands to run before the build
phases/build	Optional	Commands to run during the build
phases/post_build	Optional	Commands to run after the build

© Digital Cloud Training | https://digitalcloud.training

CI/CD with CodeBuild

© Digital Cloud Training | https://digitalcloud.training

Add Build Stage to Pipeline

© Digital Cloud Training | https://digitalcloud.training

CI/CD with CodeBuild

© Digital Cloud Training | https://digitalcloud.training

AWS CodeDeploy

© Digital Cloud Training | https://digitalcloud.training

AWS CodeDeploy

- CodeDeploy is a deployment service that automates application deployments
- Deploys to Amazon EC2 instances, on-premises instances, serverless Lambda functions, and Amazon ECS
- You can deploy a nearly unlimited variety of application content, including:
  - Serverless AWS Lambda functions
  - Web and configuration files
  - Executables
  - Packages
  - Scripts
  - Multimedia files

© Digital Cloud Training | https://digitalcloud.training

AWS CodeDeploy

- An AWS CodeDeploy application contains information about what to deploy and how to deploy it
- Need to choose the compute platform:
  - EC2/On-premises
  - AWS Lambda
  - Amazon ECS

© Digital Cloud Training | https://digitalcloud.training

AWS CodeDeploy

EC2/On-Premises:

- Amazon EC2, on-premises servers, or both
- Traffic is directed using an **in-place** or **blue/green** deployment type

© Digital Cloud Training | https://digitalcloud.training

AWS CodeDeploy

AWS Lambda:

- Used to deploy applications that consist of an updated version of a Lambda function
- You can manage the way in which traffic is shifted to the updated Lambda function versions during a deployment by choosing a **canary**, **linear**, or **all-at-once** configuration

© Digital Cloud Training | https://digitalcloud.training

AWS CodeDeploy

Amazon ECS:

- Used to deploy an Amazon ECS containerized application as a task set
- CodeDeploy performs a **blue/green** deployment by installing an updated version of the application as a new replacement task set
- CodeDeploy reroutes production traffic from the original application task set to the replacement task set
- The original task set is terminated after a successful deployment
- You can manage the way in which traffic is shifted to the updated task set during a deployment by choosing a **canary**, **linear**, or **all-at-once** configuration

© Digital Cloud Training | https://digitalcloud.training

AWS CodeDeploy

© Digital Cloud Training | https://digitalcloud.training

Blue/Green Traffic Shifting

- **AWS Lambda:** Traffic is shifted from one version of a Lambda function to a new version of the same Lambda function
- **Amazon ECS:** Traffic is shifted from a task set in your Amazon ECS service to an updated, replacement task set in the same Amazon ECS service
- **EC2/On-Premises:** Traffic is shifted from one set of instances in the original environment to a replacement set of instances
- **Note:** All AWS Lambda and Amazon ECS deployments are blue/green. An EC2/On-Premises deployment can be in-place or blue/green

© Digital Cloud Training | https://digitalcloud.training

Blue/Green Traffic Shifting

For Amazon ECS and AWS Lambda there are three ways traffic can be shifted during a deployment:

- **Canary** - Traffic is shifted in two increments. You can choose from predefined canary options that specify the percentage of traffic shifted to your updated Amazon ECS task set / Lambda function in the first increment and the interval, in minutes, before the remaining traffic is shifted in the second increment
- **Linear** - Traffic is shifted in equal increments with an equal number of minutes between each increment. You can choose from predefined linear options that specify the percentage of traffic shifted in each increment and the number of minutes between each increment
- **All-at-once** - All traffic is shifted from the original Amazon ECS task set / Lambda function to the updated Amazon ECS task set / Lambda function all at once

© Digital Cloud Training | https://digitalcloud.training

ECS-LAB-4 - CodeDeploy Application and Pipeline

© Digital Cloud Training | https://digitalcloud.training

ECS Lab Part 4

1. Create Image and Push to ECR Repository
2. Create Task Definition and ALB
3. Create Fargate Cluster and Service
4. **CodeDeploy Application and Pipeline**
5. Implement Blue/Green Update to ECS

© Digital Cloud Training | https://digitalcloud.training

ECS Lab Part 4

© Digital Cloud Training | https://digitalcloud.training

ECS Lab Part 4

© Digital Cloud Training | https://digitalcloud.training

ECS-LAB-5 - Implement Blue/Green Update to ECS

© Digital Cloud Training | https://digitalcloud.training

ECS Lab Part 5

1. Create Image and Push to ECR Repository
2. Create Task Definition and ALB
3. Create Fargate Cluster and Service
4. CodeDeploy Application and Pipeline
5. **Implement Blue/Green Update to ECS**

© Digital Cloud Training | https://digitalcloud.training

ECS Lab Part 4

© Digital Cloud Training | https://digitalcloud.training

ECS Lab Part 4

© Digital Cloud Training | https://digitalcloud.training

AWS CodeStar

© Digital Cloud Training | https://digitalcloud.training

AWS CodeStar

- AWS CodeStar enables you to quickly develop, build, and deploy applications on AWS
- It is a pre-configured continuous delivery toolchain for developing, building, testing, and deploying
- Use project templates to develop applications on services such as:
  - Amazon EC2
  - AWS Lambda
  - Elastic Beanstalk
- You can use code editors such as Visual Studio, Eclipse or the AWS CLI

© Digital Cloud Training | https://digitalcloud.training

AWS CodeStar

- Uses IAM to manage developer identities
- Built-in role-based policies for secure team access
- Share projects with three access levels:
  - Owners
  - Contributors
  - Viewers
- Application code is stored in CodeCommit
- Compiles and packages source code with CodeBuild
- A pre-configured pipeline is used through CodePipeline
- Automated deployments with CodeDeploy and CloudFormation

© Digital Cloud Training | https://digitalcloud.training

AWS CodeStar

© Digital Cloud Training | https://digitalcloud.training

AWS Cloud9

© Digital Cloud Training | https://digitalcloud.training

AWS Cloud9

- AWS Cloud9 is an integrated development environment (IDE)
- Used by developers to write, run, and debug code
- Editor provides syntax highlighting, code completion, and error checking
- Terminal is used to navigate the file system, run commands, and manage code
- Provides collaboration features that allow multiple developers to work on the same codebase simultaneously
- Provides a range of debugging tools to identify and fix errors in code
- Integrates with many AWS services including AWS Lambda, Amazon EC2, and AWS CodePipeline

© Digital Cloud Training | https://digitalcloud.training

AWS Amplify and AppSync

© Digital Cloud Training | https://digitalcloud.training

AWS Amplify

- Tools and features for building full-stack applications on AWS
- Build web and mobile backends, and web frontend UIs
- AWS Amplify Studio is a visual interface for building web and mobile apps:
  - Use the visual interface to define a data model, user authentication, and file storage without backend expertise
  - Easily add AWS services not available within Amplify Studio using the AWS Cloud Development Kit (CDK)
  - Connect mobile and web apps using Amplify Libraries for iOS, Android, Flutter, React Native, and web (JavaScript)
- AWS Amplify Hosting is a fully managed CI/CD and hosting service for fast, secure, and reliable static and server-side rendered apps

© Digital Cloud Training | https://digitalcloud.training

AWS AppSync

- AWS AppSync is a fully managed service that makes it easy to develop GraphQL APIs
- Applications can securely access, manipulate, and receive real-time updates from multiple data sources such as databases or APIs

AWS AppSync

- AWS AppSync automatically scales a GraphQL API execution engine up and down to meet API request volumes
- Uses GraphQL, a data language that enables client apps to fetch, change and subscribe to data from servers
- AWS AppSync lets you specify which portions of your data should be available in a real-time manner using GraphQL Subscriptions
- AWS AppSync supports AWS Lambda, Amazon DynamoDB, and Amazon Elasticsearch
- Server-side data caching capabilities reduce the need to directly access data sources
- AppSync is fully managed and eliminates the operational overhead of managing cache clusters

© Digital Cloud Training | https://digitalcloud.training

AWS AppSync

Example of using **AppSync** and **Amplify** to simplify access to microservices

Amplify is used to build and host the WebStore application and create backend services

AppSync creates a unified API layer for integrating the microservices

© Digital Cloud Training | https://digitalcloud.training

SECTION 12

Databases and Analytics

© Digital Cloud Training | https://digitalcloud.training

Amazon Relational Database Service (RDS)

© Digital Cloud Training | https://digitalcloud.training

Amazon RDS

RDS supports the following database engines:

- Amazon Aurora
- MySQL
- MariaDB
- Oracle
- Microsoft SQL Server
- PostgreSQL

© Digital Cloud Training | https://digitalcloud.training

Amazon RDS Scaling Up (vertically)

© Digital Cloud Training | https://digitalcloud.training

Disaster Recovery (DR) and Scaling Out (Horizontally)

© Digital Cloud Training | https://digitalcloud.training

Amazon RDS

- RDS uses EC2 instances, so you must choose an instance family/type
- Relational databases are known as Structured Query Language (SQL) databases
- RDS is an Online Transaction Processing (OLTP) type of database
- Easy to setup, highly available, fault tolerant, and scalable
- Common use cases include online stores and banking systems
- You can encrypt your Amazon RDS instances and snapshots at rest by enabling the encryption option for your Amazon RDS DB instance (during creation)
- Encryption uses AWS Key Management Service (KMS)

© Digital Cloud Training | https://digitalcloud.training

Amazon RDS

- Amazon RDS supports the following database engines:
  - SQL Server
  - Oracle
  - MySQL Server
  - PostgreSQL
  - Aurora
  - MariaDB
- Scales up by increasing instance size (compute and storage)
- Read replicas option for read heavy workloads (scales out for reads/queries only)
- Disaster recovery with Multi-AZ option

© Digital Cloud Training | https://digitalcloud.training

Amazon RDS Backup and Recovery

© Digital Cloud Training | https://digitalcloud.training

Amazon RDS Automated Backups

Restore can be to **any** point in time during the **retention period**

New DB Instance

Restore

Backup window **Info**

Select the period you want automated backups of the database to be created by Amazon RDS.

- (●) Select window
- ( ) No preference

Start time				Duration	
00 ▼	:	00 ▼	UTC	0.5 ▼	hours

DB Instance

Backup

Snapshot

Retention is **0–35** days

© Digital Cloud Training | https://digitalcloud.training

DigitalCloud TRAINING

Amazon RDS Manual Backups (Snapshot)

- Backs up the entire DB instance, not just individual databases
- For single-AZ DB instances there is a brief suspension of I/O
- For Multi-AZ SQL Server, I/O activity is briefly suspended on primary
- For Multi-AZ MariaDB, MySQL, Oracle and PostgreSQL the snapshot is taken from the standby
- Snapshots do not expire (no retention period)

© Digital Cloud Training | https://digitalcloud.training

Amazon RDS Maintenance Windows

- Operating system and DB patching can require taking the database offline
- These tasks take place during a maintenance window
- By default a weekly maintenance window is configured
- You can choose your own maintenance window

© Digital Cloud Training | https://digitalcloud.training

Create Amazon RDS Database

© Digital Cloud Training | https://digitalcloud.training

Read Replicas and Multi-AZ

© Digital Cloud Training | https://digitalcloud.training

Amazon Aurora

© Digital Cloud Training | https://digitalcloud.training

Amazon Aurora

- Amazon Aurora is an AWS database offering in the RDS family
- Amazon Aurora is a MySQL and PostgreSQL-compatible relational database built for the cloud
- Amazon Aurora is up to five times faster than standard MySQL databases and three times faster than standard PostgreSQL databases
- Amazon Aurora features a distributed, fault-tolerant, self-healing storage system that auto-scales up to 128TB per database instance

© Digital Cloud Training | https://digitalcloud.training

Amazon Aurora

Aurora Fault Tolerance

- Fault tolerance across 3 AZs
- Single logical volume
- Aurora Replicas scale-out read requests
- Can **promote** Aurora Replica to be a new primary or create new primary
- Can use **Auto Scaling** to add replicas

© Digital Cloud Training | https://digitalcloud.training

Amazon Aurora Key Features

Aurora Feature	Benefit
High performance and scalability	Offers high performance, self-healing storage that scales up to 128TB, point-in-time recovery and continuous backup to S3
DB compatibility	Compatible with existing MySQL and PostgreSQL open source databases
Aurora Replicas	In-region read scaling and failover target – up to 15 (can use Auto Scaling)
MySQL Read Replicas	Cross-region cluster with read scaling and failover target – up to 5 (each can have up to 15 Aurora Replicas)
Global Database	Cross-region cluster with read scaling (fast replication / low latency reads). Can remove secondary and promote
Multi-Master	Scales out writes within a region. In preview currently and will not appear on the exam
Serverless	On-demand, autoscaling configuration for Amazon Aurora - does not support read replicas or public IPs (can only access through VPC or Direct Connect - not VPN)

© Digital Cloud Training | https://digitalcloud.training

Amazon Aurora Replicas

Feature	Aurora Replica	MySQL Replica
Number of replicas	Up to 15	Up to 5
Replication type	Asynchronous (milliseconds)	Asynchronous (seconds)
Performance impact on primary	Low	High
Replica location	In-region	Cross-region
Act as failover target	Yes (no data loss)	Yes (potentially minutes of data loss)
Automated failover	Yes	No
Support for user-defined replication delay	No	Yes
Support for different data or schema vs. primary	No	Yes

© Digital Cloud Training | https://digitalcloud.training

Amazon RDS Security

© Digital Cloud Training | https://digitalcloud.training

Amazon RDS Security

© Digital Cloud Training | https://digitalcloud.training

DigitalCloud TRAINING

Amazon RDS Security

- Encryption **at rest** can be enabled – includes DB storage, backups, read replicas and snapshots
- You can only enable encryption for an Amazon RDS DB instance when you create it, not after the DB instance is created
- DB instances that are encrypted can't be modified to disable encryption
- Uses AES 256 encryption and encryption is transparent with minimal performance impact
- RDS for Oracle and SQL Server is also supported using Transparent Data Encryption (TDE) (may have performance impact)
- AWS KMS is used for managing encryption keys

© Digital Cloud Training | https://digitalcloud.training

Amazon RDS Security

- You can't have:
  - An **encrypted** read replica of an **unencrypted** DB instance
  - An **unencrypted** read replica of an **encrypted** DB instance
- Read replicas of encrypted primary instances are encrypted
- The same KMS key is used if in the same Region as the primary
- If the read replica is in a different Region, a different KMS key is used
- You can't restore an unencrypted backup or snapshot to an encrypted DB instance

© Digital Cloud Training | https://digitalcloud.training

Amazon RDS Security

© Digital Cloud Training | https://digitalcloud.training

Create Encrypted Copy of RDS Database

© Digital Cloud Training | https://digitalcloud.training

Amazon RDS Security

© Digital Cloud Training | https://digitalcloud.training

Amazon ElastiCache

© Digital Cloud Training | https://digitalcloud.training

Amazon ElastiCache

- Fully managed implementations **Redis** and **Memcached**
- ElastiCache is a **key/value** store
- In-memory database offering high performance and low latency
- Can be put in front of databases such as RDS and DynamoDB
- ElastiCache nodes run on Amazon EC2 instances, so you must choose an instance family/type

© Digital Cloud Training | https://digitalcloud.training

Amazon ElastiCache

Feature	Memcached	Redis (cluster mode disabled)	Redis (cluster mode enabled)
Data persistence	No	Yes	Yes
Data types	Simple	Complex	Complex
Data partitioning	Yes	No	Yes
Encryption	No	Yes	Yes
High availability (replication)	No	Yes	Yes
Multi-AZ	Yes, place nodes in multiple AZs. No failover or replication	Yes, with auto-failover. Uses read replicas (0-5 per shard)	Yes, with auto-failover. Uses read replicas (0-5 per shard)
Scaling	Up (node type); out (add nodes)	Up (node type); out (add replica)	Up (node type); out (add shards)
Multithreaded	Yes	No	No
Backup and restore	No (and no snapshots)	Yes, automatic and manual snapshots	Yes, automatic and manual snapshots

© Digital Cloud Training | https://digitalcloud.training

Amazon ElastiCache Use Cases

- Data that is relatively **static** and **frequently accessed**
- Applications that are tolerant of stale data
- Data is slow and expensive to get compared to cache retrieval
- Require push-button scalability for memory, writes and reads
- Often used for storing session state

© Digital Cloud Training | https://digitalcloud.training

Amazon ElastiCache Examples

Use Case	Benefit
Web session store	In cases with load-balanced web servers, store web session information in Redis so if a server is lost, the session info is not lost, and another web server can pick it up
Database caching	Use Memcached in front of AWS RDS to cache popular queries to offload work from RDS and return results faster to users
Leaderboards	Use Redis to provide a live leaderboard for millions of users of your mobile app
Streaming data dashboards	Provide a landing spot for streaming sensor data on the factory floor, providing live real-time dashboard displays

© Digital Cloud Training | https://digitalcloud.training

Scaling ElastiCache

© Digital Cloud Training | https://digitalcloud.training

Amazon ElastiCache - Scalability

Memcached

- Add nodes to a cluster
- Scale vertically (node type) – must create a **new cluster** manually

Redis

- Cluster mode **disabled**:
  - Add replica or change node type – creates a new cluster and migrates data
- Cluster mode **enabled**:
  - Online resharding to add or remove shards; vertical scaling to change node type
  - Offline resharding to add or remove shards change node type or upgrade engine (more flexible than online)

© Digital Cloud Training | https://digitalcloud.training

Amazon ElastiCache Memcached

© Digital Cloud Training | https://digitalcloud.training

Amazon ElastiCache Redis (Cluster mode enabled)

© Digital Cloud Training | https://digitalcloud.training

Amazon ElastiCache Redis (Cluster mode disabled)

© Digital Cloud Training | https://digitalcloud.training

Create ElastiCache Cluster

© Digital Cloud Training | https://digitalcloud.training

Amazon MemoryDB for Redis

© Digital Cloud Training | https://digitalcloud.training

Amazon MemoryDB for Redis

- Redis-compatible, durable, in-memory database service that delivers ultra-fast performance
- Entire dataset is stored in memory – entire DB solution
- Purpose-built for modern applications with microservices architectures
- Build applications using the same flexible and friendly Redis data structures, APIs, and commands
- Microsecond read and single-digit millisecond write latency and high throughput
- Data stored durably across multiple AZs using a distributed transactional log
- Supports write scaling with sharding and read scaling by adding replicas

MemoryDB for Redis vs ElastiCache

- Use ElastiCache for caching DB queries
- Use MemoryDB for a full DB solution combining DB and cache
- MemoryDB offers higher performance with lower latency
- MemoryDB offers strong consistency for primary nodes and eventual consistency for replica nodes
- With ElastiCache there can be some inconsistency and latency depending on the engine and caching strategy

© Digital Cloud Training | https://digitalcloud.training

Amazon Kinesis Core Knowledge

© Digital Cloud Training | https://digitalcloud.training

Amazon Kinesis Services

© Digital Cloud Training | https://digitalcloud.training

Amazon Kinesis Data Streams

- Producers send data to Kinesis, data is stored in Shards for 24 hours (by default, up to 7 days)
- Consumers then take the data and process it - data can then be saved into another AWS service
- Real time (~200ms)

© Digital Cloud Training | https://digitalcloud.training

Kinesis Client Library (KCL)

- The Kinesis Client Library (KCL) helps you consume and process data from a Kinesis data stream

© Digital Cloud Training | https://digitalcloud.training

Kinesis Client Library (KCL)

- Each shard is processed by exactly one KCL worker and has exactly one corresponding record processor
- One worker can process any number of shards, so it's fine if the number of shards exceeds the number of instances

Amazon Kinesis Data Streams

Order is maintained for records **within a shard**

A partition key can be specified with PutRecord to group data by shard

© Digital Cloud Training | https://digitalcloud.training

DigitalCloud TRAINING

Kinesis Data Firehose

- Producers send data to Firehose
- There are no Shards, completely automated (scalability is elastic)
- Firehose data is sent to another AWS service for storing, data can be optionally processed/transformed using AWS Lambda
- Near real-time delivery (~60 seconds latency)

© Digital Cloud Training | https://digitalcloud.training

Kinesis Data Firehose

Kinesis Data Firehose destinations:

- RedShift (via an intermediate S3 bucket)
- Elasticsearch
- Amazon S3
- Splunk
- Datadog
- MongoDB
- New Relic
- HTTP Endpoint

© Digital Cloud Training | https://digitalcloud.training

Kinesis Data Analytics

- Provides real-time SQL processing for streaming data
- Provides analytics for data coming in from Kinesis Data Streams and Kinesis Data Firehose
- Destinations can be Kinesis Data Streams, Kinesis Data Firehose, or AWS Lambda

© Digital Cloud Training | https://digitalcloud.training

Amazon Kinesis Client Library (KCL)

© Digital Cloud Training | https://digitalcloud.training

Amazon Kinesis Client Library (KCL)

- The Kinesis Client Library (KCL) helps you consume and process data from a Kinesis data stream
- The KCL is different from the Kinesis Data Streams API that is available in the AWS SDKs
  - The Kinesis Data Streams API helps you manage many aspects of Kinesis Data Streams (including creating streams, resharding, and putting and getting records)
  - The KCL provides a layer of abstraction specifically for processing data in a consumer role

Amazon Kinesis Client Library (KCL)

- Each shard is processed by exactly one KCL worker and has exactly one corresponding record processor, so you never need multiple instances to process one shard
- However, one worker can process any number of shards, so it's fine if the number of shards exceeds the number of instances

Amazon OpenSearch Service (Elasticsearch)

© Digital Cloud Training | https://digitalcloud.training

Amazon OpenSearch Service

Successor to **Amazon Elasticsearch Service**

Search, visualize, and analyze **text and unstructured data**

Amazon OpenSearch Service

Deploy **nodes** and **replicas** across AZs

Fully Managed

Petabyte Scale

Secure

Highly Available

Scalable

Deploy to **Amazon VPC** and integrates with **IAM**

© Digital Cloud Training | https://digitalcloud.training

DigitalCloud TRAINING

Amazon OpenSearch Service

- Distributed search and analytics suite
- Based on the popular open source Elasticsearch
- Supports queries using SQL syntax
- Integrates with open-source tools
- Scale by adding or removing instances
- Availability in up to three Availability Zones
- Backup using snapshots
- Encryption at-rest and in-transit

© Digital Cloud Training | https://digitalcloud.training

OpenSearch Service Deployment

- Clusters are created (Management Console, API, or CLI)
- Clusters are also known as OpenSearch Service domains
- You specify the number of instances and instance types
- Storage options include UltraWarm or Cold storage

© Digital Cloud Training | https://digitalcloud.training

Ingesting Data into OpenSearch Service Domains

OpenSearch in an Amazon VPC

- Clusters can be deployed in a VPC for secure intra-VPC communications
- VPN or proxy required to connect from the internet (public domains are directly accessible)
- Cannot use IP-based access policies

© Digital Cloud Training | https://digitalcloud.training

OpenSearch in an Amazon VPC

- Limitations of VPC deployments:
  - You can’t switch from VPC to a public endpoint. The reverse is also true
  - You can’t launch your domain within a VPC that uses dedicated tenancy
  - After you place a domain within a VPC, you can’t move it to a different VPC, but you can change the subnets and security group settings

© Digital Cloud Training | https://digitalcloud.training

The ELK Stack

- ELK stands for Elasticsearch, Logstash, and Kibana

- This is a popular combination of projects
- Aggregate logs from systems and applications, analyze these logs, and create visualizations
- Use cases include:
  - Visualizing application and infrastructure monitoring data
  - Troubleshooting
  - Security analytics

© Digital Cloud Training | https://digitalcloud.training

OpenSearch Access Control

- **Resource-based policies** – often called a domain access policy
- **Identity-based policies** – attached to users or roles (principals)
- **IP-based policies** – Restrict access to one or more IP addresses or CIDR blocks
- **Fine-grained access control** – Provides:
  - Role-based access control
  - Security at the index, document, and field level
  - OpenSearch Dashboards multi-tenancy
  - HTTP basic authentication for OpenSearch and OpenSearch Dashboards

© Digital Cloud Training | https://digitalcloud.training

OpenSearch Access Control

- Authentication options include:
  - Federation using SAML to on-premises directories
  - Amazon Cognito and social identity providers

© Digital Cloud Training | https://digitalcloud.training

OpenSearch Best Practices

- Deploy OpenSearch data instances across three Availability Zones (AZs) for the best availability
- Provision instances in multiples of three for equal distribution across AZs
- If three AZs are not available use two AZs with equal numbers of instances

© Digital Cloud Training | https://digitalcloud.training

OpenSearch Best Practices

- Use three dedicated master nodes
- Configure at least one replica for each index
- Apply restrictive resource-based access policies to the domain (or use fine-grained access control)
- Create the domain within an Amazon VPC
- For sensitive data enable node-to-node encryption and encryption at rest

© Digital Cloud Training | https://digitalcloud.training

Amazon Athena and AWS Glue

© Digital Cloud Training | https://digitalcloud.training

Amazon Athena and AWS Glue

© Digital Cloud Training | https://digitalcloud.training

Amazon Athena and AWS Glue

Amazon Athena

SQL queries can also be run against these data sources

Data from source is **mapped** in **tables** in Athena and is queryable

A **Lambda function** connects Athena to **data source**

Lambda function

© Digital Cloud Training | https://digitalcloud.training

Amazon Athena

- Athena queries data in S3 using SQL
- Can be connected to other data sources with Lambda
- Data can be in CSV, TSV, JSON, Parquet and ORC formats
- Uses a managed Data Catalog (AWS Glue) to store information and schemas about the databases and tables

© Digital Cloud Training | https://digitalcloud.training

Optimizing Athena for Performance

- **Partition your data**
- **Bucket your data** – bucket the data within a single partition
- **Use Compression** – AWS recommend using either Apache Parquet or Apache ORC
- **Optimize file sizes**
- **Optimize columnar data store generation** – Apache Parquet and Apache ORC are popular columnar data stores
- **Optimize ORDER BY and Optimize GROUP BY**
- **Use approximate functions**
- **Only include the columns that you need**

© Digital Cloud Training | https://digitalcloud.training

AWS Glue

- Fully managed extract, transform and load (ETL) service
- Used for preparing data for analytics
- AWS Glue runs the ETL jobs on a fully managed, scale-out Apache Spark environment
- AWS Glue discovers data and stores the associated metadata (e.g. table definition and schema) in the AWS Glue Data Catalog
- Works with data lakes (e.g. data on S3), data warehouses (including RedShift), and data stores (including RDS or EC2 databases)

© Digital Cloud Training | https://digitalcloud.training

AWS Glue

- You can use a **crawler** to populate the AWS Glue Data Catalog with tables
- A crawler can crawl multiple data stores in a single run
- Upon completion, the crawler creates or updates one or more tables in your Data Catalog.
- ETL jobs that you define in AWS Glue use the Data Catalog tables as sources and targets

© Digital Cloud Training | https://digitalcloud.training

SECTION 13

Management and Security

© Digital Cloud Training | https://digitalcloud.training

AWS Cloud Development Kit (AWS CDK)

© Digital Cloud Training | https://digitalcloud.training

AWS CDK

- Build apps in the cloud with standard programming languages
- Supports TypeScript, JavaScript, Python, Java, C#/.Net, and Go
- Use these languages to create reusable **Constructs**
- Compose them into **Stacks** and **Apps**
- Command line toolkit
- ‘Synthesizes’ AWS CloudFormation templates

AWS AppConfig

© Digital Cloud Training | https://digitalcloud.training

AWS AppConfig

- Create, manage, and deploy application configurations
- Capability of AWS Systems Manager
- A **configuration** is a collection of settings that influence the behavior of your application
- Applications can be hosted on:
  - Amazon EC2 instances
  - AWS Lambda
  - Mobile applications
  - IoT devices
- Reduces errors associated with configuration changes and streamlines deployment

© Digital Cloud Training | https://digitalcloud.training

AWS AppConfig

- Configurations can be stored in:
  - Amazon S3
  - AWS AppConfig
  - Systems Manager Parameter Store
  - Systems Manager Document Store
  - Bitbucket, GitHub, CodeCommit (via CodePipeline)
- Applications must be updated to check for and retrieve configuration data
- API actions include:
  - StartConfigurationSession
  - GetLatestConfiguration

© Digital Cloud Training | https://digitalcloud.training

AWS AppConfig

- Validators are used to ensure that configuration data is syntactically and semantically correct
- Validators are either:
  - JSON Schema Validators
  - AWS Lambda Validators
- Deployment type is either:
  - Linear – uses a growth factor which is a step %
  - Exponential – uses the exponential formula G*(2^N)
- Deployment strategies:
  - AppConfig.AllAtOnce – all targets at once
  - AppConfig.Linear50PercentEvery30Seconds – 50% of targets every 30 seconds

© Digital Cloud Training | https://digitalcloud.training

AWS AppConfig – Example Configurations

Enables or disables mobile payments and default payments on a per-region basis

```
{
  "allow_mobile_payments": {
    "enabled": false
  },
  "default_payments_per_region": {
    "enabled": true
  }
}
```

© Digital Cloud Training | https://digitalcloud.training

AWS AppConfig – Example Configurations

Enforces limits on how an application processes requests

```
{
    "throttle-limits": {
      "enabled": "true",
      "throttles": [
        {
          "simultaneous_connections": 12
        },
        {
          "tps_maximum": 5000
        }
      ],
      "limit-background-tasks": [
        true
      ]
    }
}
```

© Digital Cloud Training | https://digitalcloud.training

Amazon CloudWatch Overview

© Digital Cloud Training | https://digitalcloud.training

Amazon CloudWatch

CloudWatch is used for performance monitoring, alarms, log collection and automated actions

Use cases / benefits include:

- Collect performance metrics from AWS and on-premises systems
- Automate responses to operational changes
- Improve operational performance and resource optimization
- Derive actionable insights from logs
- Get operational visibility and insight

© Digital Cloud Training | https://digitalcloud.training

Amazon CloudWatch

CloudWatch Core Features:

- **CloudWatch Metrics** – services send time-ordered data points to CloudWatch
- **CloudWatch Alarms** – monitor metrics and initiate actions
- **CloudWatch Logs** – centralized collection of system and application logs
- **CloudWatch Events** – stream of system events describing changes to AWS resources and can trigger actions

© Digital Cloud Training | https://digitalcloud.training

Amazon CloudWatch Metrics

- Metrics are sent to CloudWatch for many AWS services
- EC2 metrics are sent every **5 minutes** by default (free)
- Detailed EC2 monitoring sends every **1 minute** (chargeable)
- Unified CloudWatch Agent sends system-level metrics for EC2 and on-premises servers
- System-level metrics include memory and disk usage

© Digital Cloud Training | https://digitalcloud.training

Amazon CloudWatch Metrics

- Can publish custom metrics using CLI or API
- Custom metrics are one of the following resolutions:
  - **Standard resolution** – data having a one-minute granularity
  - **High resolution** – data at a granularity of one second
- AWS metrics are standard resolution by default

© Digital Cloud Training | https://digitalcloud.training

Amazon CloudWatch Alarms

Two types of alarms

- **Metric alarm** – performs one or more actions based on a single metric
- **Composite alarm** – uses a rule expression and takes into account multiple alarms
- Metric alarm states:
  - OK – Metric is within a threshold
  - ALARM – Metric is outside a threshold
  - INSUFFICIENT_DATA – not enough data

© Digital Cloud Training | https://digitalcloud.training

Create a Custom Metric

© Digital Cloud Training | https://digitalcloud.training

Create a CloudWatch Alarm

© Digital Cloud Training | https://digitalcloud.training

Amazon CloudWatch Logs

© Digital Cloud Training | https://digitalcloud.training

Amazon CloudWatch Logs

- Gather application and system logs in CloudWatch
- Defined expiration policies and KMS encryption
- Send to:
  - Amazon S3 (export)
  - Kinesis Data Streams
  - Kinesis Data Firehose

Unified CloudWatch Agent installed on EC2 and on-premises servers

© Digital Cloud Training | https://digitalcloud.training

Review CloudWatch Logs

© Digital Cloud Training | https://digitalcloud.training

The Unified CloudWatch Agent

© Digital Cloud Training | https://digitalcloud.training

The Unified CloudWatch Agent

The unified CloudWatch agent enables you to do the following:

- Collect internal system-level metrics from Amazon **EC2 instances** across operating systems
- Collect system-level metrics from **on-premises servers**
- Retrieve custom metrics from your applications or services using the StatsD and collectd protocols
- Collect logs from Amazon EC2 instances and on-premises servers (Windows / Linux)

© Digital Cloud Training | https://digitalcloud.training

The Unified CloudWatch Agent

- Agent must be installed on the server
- Can be installed on:
  - Amazon EC2 instances
  - On-premises servers
  - Linux, Windows Server, or macOS

© Digital Cloud Training | https://digitalcloud.training

AWS CloudTrail

© Digital Cloud Training | https://digitalcloud.training

AWS CloudTrail

- CloudTrail logs **API activity** for auditing
- By default, management events are logged and retained for 90 days
- A **CloudTrail Trail** logs any events to S3 for indefinite retention
- Trail can be within Region or all Regions
- CloudWatch Events can triggered based on API calls in CloudTrail
- Events can be streamed to CloudWatch Logs

© Digital Cloud Training | https://digitalcloud.training

AWS CloudTrail

© Digital Cloud Training | https://digitalcloud.training

CloudTrail – Types of Events

- **Management events** provide information about management operations that are performed on resources in your AWS account
- **Data events** provide information about the resource operations performed on or in a resource
- **Insights events** identify and respond to unusual activity associated with write API calls by continuously analyzing CloudTrail management events

© Digital Cloud Training | https://digitalcloud.training

Create a CloudTrail Trail

© Digital Cloud Training | https://digitalcloud.training

AWS Key Management Service (KMS)

© Digital Cloud Training | https://digitalcloud.training

AWS Key Management Service (KMS)

- Create and managed **symmetric** and **asymmetric** encryption keys
- The **KMS keys** are protected by hardware security modules (HSMs)

Developer creates **customer managed** KMS keys in AWS KMS

© Digital Cloud Training | https://digitalcloud.training

KMS Keys

- KMS keys are the primary resources in AWS KMS
- Used to be known as “customer master keys” or CMKs
- The KMS key also contains the key material used to encrypt and decrypt data
- By default, AWS KMS creates the key material for a KMS key
- You can also import your own key material
- A KMS key can encrypt data up to 4KB in size
- A KMS key can generate, encrypt and decrypt Data Encryption Keys (DEKs)

© Digital Cloud Training | https://digitalcloud.training

Alternative Key Stores

External Key Store

- Keys can be stored outside of AWS to meet regulatory requirements
- You can create a KMS key in an AWS KMS external key store (XKS)
- All keys are generated and stored in an external key manager
- When using an XKS, key material never leaves your HSM

Custom Key Store

- You can create KMS keys in an AWS CloudHSM custom key store
- All keys are generated and stored in an AWS CloudHSM cluster that you own and manage
- Cryptographic operations are performed solely in the AWS CloudHSM cluster you own and manage
- Custom key stores are not available for asymmetric KMS keys

© Digital Cloud Training | https://digitalcloud.training

AWS Managed KMS Keys

- Created, managed, and used on your behalf by an AWS service that is integrated with AWS KMS
- You cannot manage these KMS keys, rotate them, or change their key policies
- You also cannot use AWS managed KMS keys in cryptographic operations directly; the service that creates them uses them on your behalf

Alias	Key ID
aws/sqs	025b9386-b1f8-4fa9-84e2-ac3220b1de59
aws/acm	2d604e85-c2d4-42dc-ab0b-0b356f5fe26e
aws/codecommit	41fea9df-e447-4992-8af7-6ddec6d81175
aws/elasticfilesystem	460c4f05-fe98-4a35-b940-3e1992f04314
aws/glue	617516fe-bf19-4da4-a743-2b13c41973e1
aws/lambda	7f513d01-784b-41b9-9c51-51621db7b5e1
aws/ebs	b9baa4f6-3e87-4256-af6a-d181940df286
aws/lightsail	bc7ba666-8e17-444a-800c-d3d4be303a97
aws/fsx	cebc434c-ee2b-4a61-9b5a-f63be9fdb068
aws/kinesis	d99014b5-09d4-480d-9b2a-3a7d7e3e9c5b

© Digital Cloud Training | https://digitalcloud.training

Data Encryption Keys

- Data keys are encryption keys that you can use to encrypt large amounts of data
- You can use AWS KMS keys to generate, encrypt, and decrypt data keys
- AWS KMS does not store, manage, or track your data keys, or perform cryptographic operations with data keys
- You must use and manage data keys outside of AWS KMS

© Digital Cloud Training | https://digitalcloud.training

KMS Keys and Automatic Rotation

Type of KMS Key	Can view	Can manage	Used only for my AWS account	Automatic rotation
Customer managed key	Yes	Yes	Yes	Optional. Every 365 days
AWS managed key	Yes	No	Yes	Required. Every 365 days
AWS owned key	No	No	No	Varies

- You cannot enable or disable key rotation for AWS owned keys
- Automatic key rotation is supported only on symmetric encryption KMS keys with key material that AWS KMS generates (**Origin = AWS_KMS**)

© Digital Cloud Training | https://digitalcloud.training

KMS Keys and Automatic Rotation

- Automatic rotation generates new key material every year (*optional* for **customer managed keys**)

Rotation only changes the **key material** used for encryption, the KMS key remains the same

© Digital Cloud Training | https://digitalcloud.training

KMS Keys and Automatic Rotation

With automatic key rotation:

- The properties of the KMS key, including its key ID, key ARN, region, policies, and permissions, do not change when the key is rotated
- You do not need to change applications or aliases that refer to the key ID or key ARN of the KMS key
- After you enable key rotation, AWS KMS rotates the KMS key automatically every year

Automatic key rotation is not supported on the following types of KMS keys:

- Asymmetric KMS keys
- HMAC KMS keys
- KMS keys in custom key stores
- KMS keys with imported key material

Note: You can rotate these KMS keys **manually**

© Digital Cloud Training | https://digitalcloud.training

Manual Rotation

- Manual rotation is creating a new KMS key with a different key ID
- You must then update your applications with the new key ID
- You can use an **alias** to represent a KMS key so you don't need to modify your application code

The **alias** is associated with the new **KMS key**

© Digital Cloud Training | https://digitalcloud.training

KMS Key Policies

- Key policies define management and usage permissions for KMS keys

```
{
  "Sid": "Allow access for Key Administrators",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:user/KMSKeyAdmin"},
  "Action": [
    "kms:Describe*",
    "kms:Put*",
    "kms:Create*",
    "kms:Update*",
    "kms:Enable*",
    "kms:Revoke*",
    "kms:List*",
    "kms:Disable*",
    "kms:Get*",
    "kms:Delete*",
    "kms:ScheduleKeyDeletion",
    "kms:CancelKeyDeletion"
  ],
  "Resource": "*"
}
```

This key policy defines the **administrative actions** that are permitted for a key administrator

© Digital Cloud Training | https://digitalcloud.training

KMS Key Policies

- Multiple policy statements can be combined to specify separate administrative and usage permissions

```
{
    "Sid": "Allow use of the key",
    "Effect": "Allow",
    "Principal": {"AWS": "arn:aws:iam::111122223333:role/EncryptionApp"},
    "Action": [
      "kms:DescribeKey",
      "kms:GenerateDataKey*",
      "kms:Encrypt",
      "kms:ReEncrypt*",
      "kms:Decrypt"
    ],
    "Resource": "*"
  }
```

This key policy defines the **cryptographic** actions for encrypting and decrypting data with the KMS key

© Digital Cloud Training | https://digitalcloud.training

KMS Key Policies

- Permissions can be specified for delegating use of the key to AWS services

```
{
  "Sid": "Allow attachment of persistent resources",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:role/EncryptionApp"},
  "Action": [
    "kms:ListGrants",
    "kms:CreateGrant",
    "kms:RevokeGrant"
  ],
  "Resource": "*",
  "Condition": {"Bool": {"kms:GrantIsForAWSResource": true}}
}
```

Grants are useful for **temporary permissions** as they can be used without modifying key policies or IAM policies

© Digital Cloud Training | https://digitalcloud.training

Additional Exam Tips

- To share snapshots with another account you must specify Decrypt and CreateGrant permissions
- The kms:ViaService condition key can be used to limit key usage to specific AWS services
- For example:

```
"Condition": {
  "StringEquals": {
    "kms:ViaService": [
      "ec2.us-west-2.amazonaws.com",
      "rds.us-west-2.amazonaws.com"
```

© Digital Cloud Training | https://digitalcloud.training

Additional Exam Tips

- Cryptographic erasure means removing the ability to decrypt data and can be achieved when using **imported key material** and deleting that key material
- You must use the DeleteImportedKeyMaterial API to remove the key material
- An InvalidKeyId exception when using SSM Parameter Store indicates the KMS key is not enabled
- Make sure you know the differences between AWS managed and customer managed KMS keys and automatic vs manual rotation

© Digital Cloud Training | https://digitalcloud.training

Create Custom KMS Keys

© Digital Cloud Training | https://digitalcloud.training

AWS KMS
API and CLI

© Digital Cloud Training | https://digitalcloud.training

AWS KMS API and CLI

Encrypt (aws kms encrypt):

- Encrypts plaintext into ciphertext by using a KMS key
- You can encrypt small amounts of arbitrary data, such as a personal identifier or database password, or other sensitive information
- You can use the Encrypt operation to move encrypted data from one AWS region to another

Decrypt (aws kms decrypt):

- Decrypts ciphertext that was encrypted by an KMS key using any of the following operations:
  - Encrypt
  - GenerateDataKey
  - GenerateDataKeyPair
  - GenerateDataKeyWithoutPlaintext
  - GenerateDataKeyPairWithoutPlaintext

© Digital Cloud Training | https://digitalcloud.training

AWS KMS API and CLI

Re-encrypt (aws kms re-encrypt):

- Decrypts ciphertext and then re-encrypts it entirely within AWS KMS
- You can use this operation to change the KMS key under which data is encrypted, such as when you manually rotate a KMS key or change the KMS key that protects a ciphertext
- You can also use it to re-encrypt ciphertext under the same KMS key, such as to change the encryption context of a ciphertext

Enable-key-rotation:

- Enables automatic rotation of the key material for the specified symmetric KMS key
- You cannot perform this operation on a KMS key in a different AWS account

© Digital Cloud Training | https://digitalcloud.training

AWS KMS API and CLI

GenerateDataKey (aws kms generate-data-key):

- Generates a unique symmetric data key
- This operation returns a plaintext copy of the data key and a copy that is encrypted under a KMS key that you specify
- You can use the plaintext key to encrypt your data outside of AWS KMS and store the encrypted data key with the encrypted data

GenerateDataKeyWithoutPlaintext (generate-data-key-without-plaintext):

- Generates a unique symmetric data key
- This operation returns a data key that is encrypted under a KMS key that you specify
- To request an asymmetric data key pair, use the GenerateDataKeyPair or GenerateDataKeyPairWithoutPlaintext operations

DigitalCloud TRAINING

Throttling and Caching

- AWS KMS has two types of quotas:
  - Resource quotas
  - Request quotas
- If you exceed a resource limit, requests to create an additional resource of that type generate an LimitExceededException error message
- Request quotas apply to API actions such as Encrypt, Decrypt, ReEncrypt, and GenerateDataKey
- If you exceed a request quota you may receive a throttling error:

```
You have exceeded the rate at which you may call KMS. Reduce the frequency of your calls.
(Service: AWSKMS; Status Code: 400; Error Code: ThrottlingException; Request ID: <ID>
```

© Digital Cloud Training | https://digitalcloud.training

Throttling and Caching

- To prevent throttling, you can:
  - Implement a backoff and retry strategy
  - Request a service quota increase
  - Implement data key caching
- Data key caching stores data keys and related cryptographic material in a cache
- Useful if your application:
  - Can reuse data keys
  - Generates numerous data keys
  - Runs cryptographic operations that are unacceptably slow, expensive, limited, or resource-intensive
- You can create a local cache using the AWS Encryption SDK and the LocalCryptoMaterialsCache feature

© Digital Cloud Training | https://digitalcloud.training

AWS Certificate Manager (ACM)

© Digital Cloud Training | https://digitalcloud.training

AWS Certificate Manager (ACM)

- Create, store and renew SSL/TLS X.509 certificates
- Single domains, multiple domain names and wildcards
- Integrates with several AWS services including:
  - **Elastic Load Balancing**
  - **Amazon CloudFront**
  - **AWS Elastic Beanstalk**
  - **AWS Nitro Enclaves**
  - **AWS CloudFormation**

© Digital Cloud Training | https://digitalcloud.training

AWS Certificate Manager (ACM)

- **Public certificates** are signed by the AWS public Certificate Authority
- You can also create a Private CA with ACM
- Can then issue private certificates
- You can also import certificates from third-party issuers

© Digital Cloud Training | https://digitalcloud.training

AWS Systems Manager

© Digital Cloud Training | https://digitalcloud.training

AWS Systems Manager

- Manages many AWS resources including Amazon EC2, Amazon S3, Amazon RDS etc.
- Systems Manager Components:
  - Automation
  - Run Command
  - Inventory
  - Patch Manager
  - Session Manager
  - Parameter Store

© Digital Cloud Training | https://digitalcloud.training

AWS Systems Manager Automation

Documents define the actions to perform (YAML or JSON)

Automates **IT operations** and **management** tasks across AWS resources

Documents → Automation → Amazon RDS

```yaml
description: Creates an RDS Snapshot for an RDS instance.
schemaVersion: '0.3'
assumeRole: "{{AutomationAssumeRole}}"
parameters:
  DBInstanceIdentifier:
    type: String
    description: (Required) The DBInstanceId ID of the RDS Instance.
  DBSnapshotIdentifier:
    type: String
    description: (Optional) The DBSnapshotIdentifier ID.
    default: ''
  InstanceTags:
    type: String
    default: ''
    description: (Optional) Tags to create for instance.
  SnapshotTags:
    type: String
    default: ''
    description: (Optional) Tags to create for snapshot
```

This automation, takes a snapshot of an RDS DB instance

© Digital Cloud Training | https://digitalcloud.training

AWS Systems Manager Run Command

Document types include **command**, **automation**, **package** etc.

Runs commands on managed EC2 instances

Documents → Run Command → Amazon EC2

```
{
  "schemaVersion": "1.2",
  "description": "List missing Microsoft Windows updates.",
  "parameters": {
    "UpdateLevel": {
      "type": "String",
      "default": "Important",
      "description": "Important: .......",
      "allowedValues": [
        "None",
        "All",
        "Important",
        "Optional"
      ]
    },
```

This command checks for missing updates

© Digital Cloud Training | https://digitalcloud.training

DigitalCloud TRAINING

AWS Systems Manager Inventory

Inventory

© Digital Cloud Training | https://digitalcloud.training

AWS Systems Manager Patch Manager

- Helps you select and deploy operating system and software patches automatically across large groups of Amazon EC2 or on-premises instances
- Patch baselines:
  - Set rules to auto-approve select categories of patches to be installed
  - Specify a list of patches that override these rules and are automatically approved or rejected
- You can also schedule maintenance windows for your patches so that they are only applied during predefined times
- Systems Manager helps ensure that your software is up-to-date and meets your compliance policies

Patch Manager

© Digital Cloud Training | https://digitalcloud.training

AWS Systems Manager Compliance

- AWS Systems Manager lets you scan your managed instances for patch compliance and configuration inconsistencies
- You can collect and aggregate data from multiple AWS accounts and Regions, and then drill down into specific resources that aren't compliant
- By default, AWS Systems Manager displays data about patching and associations
- You can also customize the service and create your own compliance types based on your requirements (must use the AWS CLI, AWS Tools for Windows PowerShell, or the SDKs)

© Digital Cloud Training | https://digitalcloud.training

AWS Systems Manager Session Manager

- Secure remote management of your instances at scale without logging into your servers
- Replaces the need for bastion hosts, SSH, or remote PowerShell
- Integrates with IAM for granular permissions
- All actions taken with Systems Manager are recorded by AWS CloudTrail
- Can store session logs in an S3 and output to CloudWatch Logs
- Requires IAM permissions for EC2 instance to access SSM, S3, and CloudWatch Logs

© Digital Cloud Training | https://digitalcloud.training

AWS Systems Manager Parameter Store

- Parameter Store provides secure, hierarchical storage for configuration data management and secrets management
- Highly scalable, available, and durable
- Store data such as passwords, database strings, and license codes as parameter values
- Store values as plaintext (unencrypted data) or ciphertext (encrypted data)
- Reference values by using the unique name that you specified when you created the parameter
- No native rotation of keys (difference with AWS Secrets Manager which does it automatically)

© Digital Cloud Training | https://digitalcloud.training

AWS Systems Manager Parameter Store

© Digital Cloud Training | https://digitalcloud.training

AWS SSM Parameter Store

- Parameter Store provides secure, hierarchical storage for configuration data and secrets
- Highly scalable, available, and durable
- Store data such as passwords, database strings, and license codes as parameter values
- Store values as plaintext (unencrypted data) or ciphertext (encrypted data)
- Reference values by using the unique name that you specified when you created the parameter
- No native rotation of keys (difference with AWS Secrets Manager which does it automatically)

© Digital Cloud Training | https://digitalcloud.training

AWS Secrets Manager

© Digital Cloud Training | https://digitalcloud.training

AWS Secrets Manager

- Stores and rotate secrets safely without the need for code deployments
- Secrets Manager offers automatic rotation of credentials (built-in) for:
  - Amazon RDS (MySQL, PostgreSQL, and Amazon Aurora)
  - Amazon Redshift
  - Amazon DocumentDB
- For other services you can write your own AWS Lambda function for automatic rotation

© Digital Cloud Training | https://digitalcloud.training

AWS Secrets Manager vs SSM Parameter Store

	Secrets Manager	SSM Parameter Store
Automatic Key Rotation	Yes, built-in for some services, use Lambda for others	No native key rotation; can use custom Lambda
Key/Value Type	String or Binary (encrypted)	String, StringList, SecureString (encrypted)
Hierarchical Keys	No	Yes
Price	Charges apply per secret	Free for standard, charges for advanced

© Digital Cloud Training | https://digitalcloud.training

Working with Secrets

© Digital Cloud Training | https://digitalcloud.training

Amazon Cognito

© Digital Cloud Training | https://digitalcloud.training

Cognito User Pools

Users can also sign in using **social IdPs**

A **User Pool** is a directory for managing **sign-in** and **sign-up** for mobile applications

Cognito User Pool

SAML OIDC

Identity Providers

Token (JWT)

Client / Mobile

Token (JWT)

API Gateway used for application

API

AWS Lambda

Lambda authorizer accepts **JWT**

Cognito acts as an **Identity Broker** between the IdP and AWS

© Digital Cloud Training | https://digitalcloud.training

DigitalCloud TRAINING

Cognito Identity Pool

© Digital Cloud Training | https://digitalcloud.training

User Pools + Identity Pools

© Digital Cloud Training | https://digitalcloud.training

AWS Web Application Firewall (WAF)

© Digital Cloud Training | https://digitalcloud.training

AWS WAF

- AWS WAF is a web application firewall
- WAF lets you create rules to filter web traffic based on conditions that include IP addresses, HTTP headers and body, or custom URIs
- WAF makes it easy to create rules that block common web exploits like SQL injection and cross site scripting

© Digital Cloud Training | https://digitalcloud.training

AWS WAF

© Digital Cloud Training | https://digitalcloud.training

AWS WAF

- **Web ACLs** – You use a web access control list (ACL) to protect a set of AWS resources
- **Rules** – Each rule contains a statement that defines the inspection criteria, and an action to take if a web request meets the criteria
- **Rule groups** – You can use rules individually or in reusable rule groups

© Digital Cloud Training | https://digitalcloud.training

AWS WAF

- **IP Sets** - An IP set provides a collection of IP addresses and IP address ranges that you want to use together in a rule statement
- **Regex pattern set** - A regex pattern set provides a collection of regular expressions that you want to use together in a rule statement

© Digital Cloud Training | https://digitalcloud.training

AWS WAF

A **rule action** tells AWS WAF what to do with a web request when it ***matches*** the criteria defined in the rule:

- **Count** – AWS WAF counts the request but doesn't determine whether to allow it or block it. With this action, AWS WAF continues processing the remaining rules in the web ACL
- **Allow** – AWS WAF allows the request to be forwarded to the AWS resource for processing and response
- **Block** – AWS WAF blocks the request and the AWS resource responds with an HTTP 403 (Forbidden) status code

© Digital Cloud Training | https://digitalcloud.training

AWS WAF

Match statements compare the web request or its origin against conditions that you provide

Match Statement	Description
Geographic match	Inspects the request's country of origin
IP set match	Inspects the request against a set of IP addresses and address ranges
Regex pattern set	Compares regex patterns against a specified request component
Size constraint	Checks size constraints against a specified request component
SQLi attack	Inspects for malicious SQL code in a specified request component
String match	Compares a string to a specified request component
XSS scripting attack	Inspects for cross-site scripting attacks in a specified request component

DigitalCloud TRAINING