

Fundamental of AWS cloud (1)


AWS Global Infrastructure

- AWS operates a global infrastructure that includes a network of data centers located in various regions around the world.
- The AWS global infrastructure is made up of regions, which are geographical areas that consist of one or more availability zones.
- Each availability zone has one or more data center(s) that is designed to be redundant and isolated from other availability zones, providing users with a high level of availability and durability for their applications and data.

Regions in the console

- https://aws.amazon.com/about-aws/global-infrastructure/regions_az/

Depending Which region you are now, the availability zones are different.



US East (N. Virginia)	us-east-1
US East (Ohio)	us-east-2
US West (N. California)	us-west-1
US West (Oregon)	us-west-2
Asia Pacific (Mumbai)	ap-south-1
Asia Pacific (Osaka)	ap-northeast-3
Asia Pacific (Seoul)	ap-northeast-2
Asia Pacific (Singapore)	ap-southeast-1
Asia Pacific (Sydney)	ap-southeast-2
Asia Pacific (Tokyo)	ap-northeast-1
Canada (Central)	ca-central-1
Europe (Frankfurt)	eu-central-1
Europe (Ireland)	eu-west-1
Europe (London)	eu-west-2

AWS as island of APIs

- Services in AWS can be accessed by APIs.
- These APIs can be accessed through the **AWS Management Console** or **programmatically by calling the APIs**.
- Today we want to learn about the following services:
 - VPC
 - EC2
 - Lambda and API GW
 - Cloud9
 - S3

VPC

- Amazon Virtual Private Cloud (Amazon VPC) is a service that allows users to create a **virtual network** in the AWS cloud.
- This virtual network can be used to securely connect AWS resources to each other, as well as to on-premises resources.
- With Amazon VPC, users can create their **own network topology**, using virtual private clouds (VPCs) and subnets, and configure security and access controls for their resources
- VPCs allow users to define the **IP address range** for their virtual network (CIDR), and to create subnets within that range to segment their network into smaller, more manageable units.

Which services are included in VPC? (1)

- **(Sub)-Networking:** Amazon VPC allows you to create your own virtual network and customize its networking configuration, including IP address range, subnets, and network security groups.
- **Load Balancing:** Amazon VPC includes load balancers that can distribute incoming traffic across multiple Amazon Elastic Compute Cloud (Amazon EC2) instances or IP addresses.
- **NAT and Internet Gateways:** Amazon VPC includes a NAT gateway service that allows you to connect your Amazon VPC to the Internet, while still maintaining private IP addresses for your instances. The servers in public subnets can use IGW to connect to Internet

Which services are included in VPC? (2)

- **Security groups:** Amazon VPC allows you to define security groups that act as a virtual firewall for your Amazon EC2 instances.
- **Elastic IP addresses:** Amazon VPC allows you to allocate and assign static IP addresses to your Amazon EC2 instances and network interfaces.
- **ENI:** An Elastic Network Interface (ENI) is a virtual network interface that you can attach to an Amazon Elastic Compute Cloud (Amazon EC2) instance or a network load balancer. ENIs allow you to attach multiple IP addresses to an instance, and to move an ENI from one instance to another

VPC in the ML context

- **Securely hosting ML models:** You can use Amazon VPC to create a secure, isolated network environment for hosting and deploying your ML models. By creating a VPC and launching your ML resources (such as Amazon EC2 instances and Amazon SageMaker endpoints) within the VPC, you can protect your models from external access and ensure that they are only accessible to authorized users and applications.
- **Connecting to on-premises data sources:** You can use Amazon VPC to create a secure VPN connection between your on-premises network and your AWS resources, allowing you to access and process data stored on-premises as part of your ML workflow.

VPC in the ML context (2)

- **Running ML workloads on dedicated hardware:** You can use Amazon VPC along with EC2 service to launch instances with specialized hardware, such as GPU instances or Inferentia-based instances, to run your ML workloads. By launching these instances within a VPC, you can ensure that they have the resources and network connectivity needed to perform efficiently.
- **Protecting data in transit:** You can use Amazon VPC to encrypt data in transit between your ML resources and other services or applications, using features such as AWS PrivateLink and SSL/TLS certificates. This can help protect your data from unauthorized access or interception as it travels across the network.

Demo

- Create a VPC with two AZs
- Each AZ has two subnets, public and private subnets
- Setup IGW
- Setup Routing Tables for each subnet type

Amazon Elastic Compute Cloud

(Amazon EC2)

EC2

- **Amazon Elastic Compute Cloud** (Amazon EC2) is a web service that provides resizable compute capacity in the cloud.
- It allows you to launch virtual machine instances (also known as "**instances**") with a variety of operating systems, including Amazon Linux, Windows, and Ubuntu.
- You can choose the instance type, which determines the hardware and resource specifications of the instance, such as the number of CPU cores, the amount of memory, and the amount of storage.

Instance families and sizes

<https://aws.amazon.com/ec2/instance-types>

- Instances are organized into **families**, where each family is optimized for different workloads.
- For example, the **M5 family** is optimized for general-purpose workloads, while the **R5 family** is optimized for memory-intensive workloads.
- Within each family, there are **multiple instance sizes** to choose from, each with different hardware and resource specifications.
- Here are a few examples of instance types and their hardware and resource specifications:
 - M5.large: 2 vCPUs, 8 GiB of memory, EBS-only storage
 - C5.xlarge: 4 vCPUs, 8 GiB of memory, EBS-only storage
 - R5.xlarge: 4 vCPUs, 16 GiB of memory, EBS-only storage

EC2 in ML context

- **Training ML models:** You can use EC2 instances to train ML models using a variety of frameworks, such as TensorFlow, PyTorch, and MXNet.

You can choose an instance type and size that meets the resource and performance requirements of your model, and you can use features such as **Elastic Fabric Adapter (EFA)** and **Amazon FSx for Lustre** to accelerate training.

- **Deploying ML models:** You can use EC2 instances to host and deploy your ML models as web services or as part of a serverless architecture using AWS Lambda. You can choose an instance type and size that meets the performance and scalability requirements of your model

EC2 in ML context (2)

- **Processing and analyzing data:** You can use EC2 instances to process and analyze large amounts of data as part of your ML workflow. You can use tools such as Amazon EMR, Apache Spark, and Amazon Athena to run distributed data processing and analysis jobs on EC2 instances
- **Running ML workflows:** You can use EC2 instances to run complex ML workflows using tools such as AWS Step Functions, AWS Data Pipeline, and AWS Glue.

Demo

- Create and EC2 instance in Public subnet
- Connect to EC2 instance

AWS Lambda and API GW

What is Lambda?

- Amazon Lambda is a **serverless** compute service that runs your code in response to events and automatically manages the compute resources needed to run your code.
- With Lambda, you can run code for virtually any application or backend service, **without the need to provision or manage servers.**
- You can use Lambda to build **event-driven applications**, such as automated workflows, data processing pipelines, and real-time data streams.
- You can write Lambda functions in a variety of languages, including Node.js, Python, Java, C#, and Go.

What is Lambda? (2)

- Lambda integrates with other AWS services, such as Amazon S3, Amazon DynamoDB, and Amazon API Gateway, making it easy to build and deploy applications that use these services.
- Lambda charges you only for the compute time that you consume, and you can scale your applications up or down as needed without any additional infrastructure.

Lambda in ML context

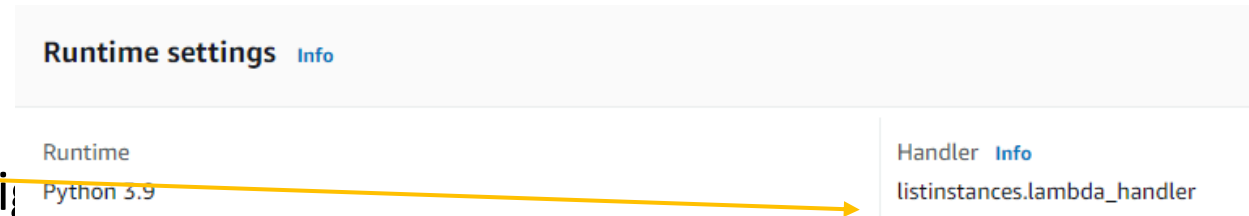
- **Running inferences on demand:** You can use Lambda to **run inferences on demand**, using ML models hosted on Amazon SageMaker. For example, you can create a Lambda function that invokes a SageMaker endpoint and returns the results of the inference to the caller. This can be useful if you want to run inferences on real-time data streams or if you want to provide an API for accessing your ML models.
- **Processing and analyzing data:** You can use Lambda to process and analyze data as part of your ML workflow. For example, you can create a Lambda function that **reads data from Amazon S3, applies some preprocessing or transformation steps, and then writes the results back to S3** or another destination. This can be useful if you want to perform data preparation tasks as part of your ML pipeline.

Lambda in ML context (2)

- **Running ML workflows:** You can use Lambda to orchestrate **complex ML workflows** using tools such as AWS Step Functions and AWS Data Pipeline. For example, you can create a Lambda function that triggers a pipeline of tasks, such as data preparation, model training, and model deployment, and that coordinates the execution of these tasks using Step Functions.
- **Running automated ML experiments:** You can use Lambda **to automate the process of running ML experiments**, using tools such as Amazon SageMaker Autopilot. For example, you can create a Lambda function that triggers an Autopilot job, and that processes the results of the job and writes them to a destination such as Amazon S3 or Amazon Redshift.
- **Running real-time data processing and analytics:** You can use Lambda **to process and analyze data in real-time** using technologies such as Amazon Kinesis and Apache Flink. For example, you can create a Lambda function that reads data from a Kinesis stream, applies some transformation or aggregation steps, and then writes the results to a destination such as S3 or Amazon Redshift.

Demo

- Create a lambda function called `listinstances`
- It uses `python 3.9` runtime
- Use `LabRole` in default execution role
- Create function
- Upload the zip file that has `listinstance.py`
- Make sure the Runtime setting has this config
- Run the code by running a test (create a test, name it and remove the content in `Event JSON` and just put `{}`)



The code creates an `EC2 client` using the `boto3.client` function and then calls the `describe_instances` method to list all the EC2 instances in your account. It then iterates through the list of instances and checks the state of each instance. If the state is 'running', it prints the instance ID, instance type, and state.

API GW

- Now we have a function that lists the running instances
- But at this point we can only run that by clicking on the “Test” in the Lambda console
- We want another service calls that Lambda function when it is needed
- We put API GW in front of that Lambda function to call it when we pull API GW URL

API GW Demo

- Connect from Internet to that function you created in previous step:
1. Go to the API Gateway console and create a new API by clicking the "Create API" button.
 2. Choose the "REST" protocol and the "New API" option, and give your API a name and description.
 3. Click the "Create API" button to create the API.
 4. Click on the "Resources" panel on the left, then click the "Actions" dropdown and select "Create Method".
 5. Choose the GET HTTP method and click the "OK" (check mark).
 6. In the "Integration Type" dropdown, select "Lambda Function" and enter the name of your Lambda function in the "Lambda Function" field.
 7. Click the "Save" button to save the integration.
 8. Click the "Actions" dropdown again and select "Deploy API".
 9. Choose a stage (e.g., "prod") and click the "Deploy" button to deploy the API.
 10. Click on in invoke URL

AWS Cloud9

What is Cloud9?

- Amazon Cloud9 is a cloud-based integrated development environment (IDE) that enables you to write, run, and debug code from your web browser.
- Cloud9 supports a wide range of programming languages and frameworks, including JavaScript, Python, PHP, Ruby, and Go.
- Cloud9 provides a collaborative workspace that allows multiple developers to work on the same codebase at the same time, with features such as real-time code collaboration, chat, and code review.
- Cloud9 integrates with other AWS services, such as Amazon EC2, Amazon S3, and AWS Lambda, making it easy to build and deploy applications using these services.
- Cloud9 is fully managed, meaning that it automatically scales and updates itself, so you can focus on writing code instead of managing infrastructure.
-

Cloud9 in ML context

- **Writing and debugging ML code:** You can use Cloud9 to write and debug ML code using frameworks such as TensorFlow, PyTorch, and MXNet. You can choose a runtime environment that includes the packages and libraries needed for your ML application, and you can use Cloud9's debugger to troubleshoot your code.
- **Collaborating on ML projects:** You can use Cloud9's collaborative workspace to work with other developers on ML projects. You can use features such as real-time code collaboration, chat, and code review to share code, discuss ideas, and review changes.

Demo

- Create a Cloud9 instance (instance size: **t3.small**, Connection: **SSH**)
- See how to use aws CLI in that instance by querying the instances in the region (that one you created earlier)

This command will list all the EC2 instances in your account, along with their details such as the instance ID, instance type, state, and public IP address.

```
aws ec2 describe-instances
```

To filter the results, you can use `--filter`:

```
aws ec2 describe-instances --filters Name=instance-state-name,Values=running
```

You can use `--output` option and specify `table` to see the result in human readable format:

```
aws ec2 describe-instances --output table
```

Demo (2)

- **Create a python code to query the instances and show how that works in Cloud9**

- You may need to install boto3 (pip install boto3)
- Run ec2-instance.py
- This code creates an EC2 client using the `boto3.client` function and then calls the `describe_instances` method to list all the EC2 instances in your account. It then iterates through the list of instances and prints the instance ID, instance type, and state for each instance.
- To filter the results to show only running instances, you can add a Filters parameter to the `describe_instances` method, like this:

```
response = ec2.describe_instances(Filters=[{'Name': 'instance-state-name', 'Values': ['running']}])
```

Amazon S3

What is S3?

- Amazon S3 is an **object storage service** that enables you to store and retrieve large amounts of data from anywhere on the Internet.
- S3 is designed for **durability, availability, and scalability**, making it a reliable and cost-effective solution for storing data of all types and sizes.
- S3 is a **pay-per-use service**, meaning that you only pay for the storage and data transfer that you consume.
- S3 offers a **range of storage classes**, including Standard, Standard-Infrequent Access, and Glacier, that enable you to store data with different performance and cost trade-offs.

What is S3? (2)

- S3 provides a number of features and capabilities that make it easy to manage and access your data, including bucket policies, versioning, and cross-region replication.
- S3 integrates with other AWS services, such as Amazon EC2, Amazon EBS, and SageMaker

S3 in ML context

- **Storing data for ML training:** You can use S3 to store large datasets that you want to use for ML training. You can also use S3's versioning feature to keep multiple versions of your data, and you can use S3's cross-region replication feature to store copies of your data in multiple regions for disaster recovery.
- **Storing trained ML models:** You can use S3 to store trained ML models that you want to use for prediction or inference later. You can save the trained model to an S3 bucket and then use the bucket's URL to access the model from other applications or services.

Demo

- Create a bucket and load the Humber logo
- Use AWS CLI in creating a bucket through Cloud9 cli

```
aws s3 mb s3://my-new-bucket
```

- Run a Python function in cloud9 to list the bucket names (`listbuckets.py`)

This code will create an s3 client using the boto3 library, and then use the `list_buckets()` method to retrieve a list of all the buckets in your AWS account. The response variable will contain a dictionary with information about the buckets, and the `bucket_names` variable will be a list containing the names of all the buckets.

Convert the above-mentioned code to run it inside a lambda function (`lambdalistbucket.py`)

Lab

- Create a new instance in a public subnet
- Install boto3 on it
- Query S3 bucket and running instances programmatically from inside of that instance