

# Module 6 : Compute

Starting date: 2081/04/29

## Section 1

Amazon Web Services (AWS) offers many compute services. This module will discuss the highlighted services.

- Amazon EC2
- Amazon Elastic Container Registry (Amazon ECR)
- Amazon Elastic Container Service ( Amazon ECS)
- AWS Elastic Beanstalk
- AWS Lambda
- Amazon Elastic Kubernetes Service ( Amazon EKS)
- AWS Fargate

| Categorizing compute services  |  |  |  |
|--|--|--|--|
| Services   | Key Concepts   | Characteristics  | Ease of Use  |
| <ul style="list-style-type: none"><li>• Amazon EC2</li></ul>   | <ul style="list-style-type: none"><li>• Infrastructure as a service (IaaS)</li><li>• Instance-based</li><li>• Virtual machines</li></ul> | <ul style="list-style-type: none"><li>• Provision virtual machines that you can manage as you choose</li></ul>   | A familiar concept to many IT professionals.   |
| <ul style="list-style-type: none"><li>• AWS Lambda</li></ul>   | <ul style="list-style-type: none"><li>• Serverless computing</li><li>• Function-based</li><li>• Low-cost</li></ul>                       | <ul style="list-style-type: none"><li>• Write and deploy code that executes on a schedule or that can be triggered by events</li><li>• Use when possible (architect for the cloud)</li></ul> | A relatively new concept for many IT staff members, but easy to use after you learn how.         |
| <ul style="list-style-type: none"><li>• Amazon ECS</li><li>• Amazon EKS</li><li>• AWS Fargate</li><li>• Amazon ECR</li></ul> | <ul style="list-style-type: none"><li>• Container-based computing</li><li>• Instance-based</li></ul>                                     | <ul style="list-style-type: none"><li>• Spin up and execute jobs more quickly</li></ul>  | AWS Fargate reduces administrative overhead, but you can use options that give you more control. |
| <ul style="list-style-type: none"><li>• AWS Elastic Beanstalk</li></ul>  | <ul style="list-style-type: none"><li>• Platform as a service (PaaS)</li><li>• For web applications</li></ul>                            | <ul style="list-style-type: none"><li>• Focus on your code (building your application)</li><li>• Can easily tie into other services—databases, Domain Name System (DNS), etc.</li></ul>      | Fast and easy to get started.  |

## Choosing the optimal computer service

Some aspects to consider:

- What is your application design?
- What are your usage patterns?
- Which configuration settings you will want to manage?

Selecting the wrong compute solution for an architecture can lead to lower performance efficiency.

- A good starting place - Understand the available compute options.

AWS,

**"compute"** refers to the processing power that allows you to run applications, programs, or tasks in the cloud. It's like the brain of your computer but in the cloud, where you can rent the amount of processing power you need without having to buy physical hardware.

**Usage** in the context of AWS refers to how much of a particular service you are using. It's a way to measure how many resources you're consuming, like how much processing power, storage, or data transfer you're using.

## Section 2 : Amazon EC2 overview

Running servers on premises is expensive, servers must be purchased, data centers must be built, staffed and maintained. Organizations must permanently provision enough hardware to handle peak workloads. After all the hardware is in place, server capacity often sits idle for significant portions of every day, which is wasteful.

Amazon **Elastic Compute Cloud** ( Amazon EC2) is a different option. It provides virtual machines where you can host the same applications you run on premises servers. It provides secure, resizable compute capacity in the cloud. Common

uses for EC2 instances include application servers, web servers, database servers, other types of servers listed here or even servers not listed here.

**Amazon Elastic Compute Cloud (Amazon EC2)** is a web service that provides resizable compute capacity in the cloud. It is designed to make web-scale cloud computing easier for developers.

Amazon EC2's simple web service interface allows you to obtain and configure capacity with minimal friction. It provides you with complete control of your computing resources and lets you run on Amazon's proven computing environment. Amazon EC2 reduces the time required to obtain and boot new server instances to minutes, allowing you to quickly scale capacity, both up and down, as your computing requirements change.

Amazon EC2 changes the economics of computing by allowing you to pay only for capacity that you actually use. Amazon EC2 provides developers the tools to build failure resilient applications and isolate themselves from common failure scenarios.

Example uses of Amazon EC2 instances

- Application server
- Web server
- Database server
- Game server
- Media server
- Catalog server
- File server
- Computing server
- Proxy server

## Amazon EC2 overview

Amazon EC2 provides virtual machines (VMs), are referred to as EC2 instances—in the cloud.

You have full administration control over the windows or Linux operating system that runs on these instances

Most server OS are supported, including recent versions of Windows, Red Hat, SUSE, Ubuntu, Amazon Linux

With Amazon EC2 , you can launch any number of instances of any size into Availability Zone, pretty much anywhere in the entire world in a matter of minutes. Instances launch from Amazon Machine Images, or AMIs, which are effectively virtual machine templates. You can control traffic to and from these instances by using security groups.

## Launching an Amazon EC2 instance

The first time you launch an Amazon EC2 instance you will likely to use the AWS Management Console Launch Instance Wizard.

## Nine key decisions that you must make when launching an EC2 instance

### 1. AMI ( Amazon Machine Image)

Is a template that is used to create an EC2 instance ( which is a virtual machine, or VM, that runs in the AWS Cloud)

AMI Choices:

- Quick Start : linux and windows AMIs provided by AWS
- My AMIs : Any AMIs that you created
- AWS Marketplace : Pre-configured templates from third parties
- Community AMIs : AMIs shared by others; use at your own risk.

### 2. Instance Type

AWS EC2 provides a selection of instance types optimized for different use cases. The instance types reflect combinations of CPU, memory, storage, and networking capacity. The different instance types give you flexibility to choose the most appropriate mix of resources, such as memory, processing power, disk type, and network performance capabilities as they're needed for your applications.

Instance type categories -

- General purpose
- Compute optimized
- Memory optimized
- Storage optimized
- Accelerated computing

Instance types offer family, generation and size

### **EC2 instance type naming and sizes**

Instance type naming consist of several parts.

Example: t3.large

- T is the family name
- 3 is the generation number
- Large is the size

a t3 instance is the third generation of the T family. In general instance types that are of higher generation are more powerful and provide better value for the price.

Instance types vary in several ways, including CPU type, CPU or Core count, storage type, storage amount, memory amount, and network performance

3. Network settings
4. IAM role
5. User data
6. Storage options
7. Tags
8. Security group
9. Key pair

## **Network features**

In addition to considering the CPU, RAM and storage needs of your workloads. It's also important to consider your network bandwidth requirements. Each instance type provides a documented network performance level. For example, in a1.medium instance, will provide up to 10 gigabits per second, but a p3dn.24xlarge instance, provides up to 100 gigabits per second. It is important to choose an instance type that supports your networking requirements. When you launch multiple EC2 instances, Amazon attempts to place them so that they are spread out across the underlying hardware, and not all located on the same hardware. However, you can use placement groups to influence the placement of interdependent instances to meet the needs of your workload.

For example, you might specify that three instances should all be deployed in the same availability zone to ensure lower network latency and higher network throughput between the instances. Many instance types also enable you to configure enhanced networking to get significantly higher packet per second performance, lower network jitter, and lower latencies.

## **Section 3: Amazon EC2 part 2**

### **3. Specify network settings**

You must specify the network location where the EC2 instance will be deployed. The choice of region must be made before you start the Launch instance wizard. Verify that you are in the correct region within the Amazon EC2 console before you choose Launch instance. Within the region, you can specify to place the instance into any existing subnet into any existing VPC. The wizard also provides a link to create a new VPC or to create a new subnet if wanted.

### **4. Attach IAM role (optional)**

It is common to use EC2 instances to run an application that must make API calls to other AWS services. To support these use cases, AWS allows you to create and attach an IAM role to the EC2 instance. Without this feature, you might be tempted to place AWS credentials on EC2 instances so an application can use them to call another AWS service. However, you should never store AWS credentials on an EC2 instance. It's not secure. Instead, attach an IAM role to the EC2 instance. An instance profile is a container for an IAM role. If you use the AWS Management

Console to create a role for Amazon EC2 , the console automatically creates an instance profile and gives it the same name as the role.

## 5. User data script (optional)

When you create an EC2 instance you have the option of passing user data to the instance. User data can automate the completion of installations and configurations at instance launch. For example, a user data script might patch and update the instance operating system, fetch and install software license keys or install additional software.

When the EC2 instance is created, the user data script will run during the final phase of the boot process. By default, user data only runs the first time that an instance starts up. User data scripts can be used strategically to reduce the number of custom AMIs that you need to build and maintain.

For example, instead of maintaining software installations and configurations and custom AMIs, consider maintaining separate user data scripts to handle those actions at instance launch.

## 6. Specify storage

When you launch an EC2 instance, you can configure storage options. For example, you can configure the size of the root volume where the guest operating system such as Windows or Linux is installed. You can also attach additional storage volumes when you launch the instance. Some AMI are configured to launch more than one storage volume by default to provide storage options that are separate from the root volume. For each volume that your instance will have, you can specify the size of the volumes and the volume type, and whether the storage will be retained if the instance is terminated. You can also specify if encryption should be used or not.

Amazon EC2 storage options

- Amazon Elastic Block store ( Amazon EBS) -
  - Durable, easy to use high performance block storage service designed to be used with Amazon EC2
  - You can stop the instance and start it again, and the data will still be there. (temporary block level storage for your instance). This storage is located

on hard disks physically attached to the host computer

- Amazon EC2 instance store -
  - Storage is provided on disks that are attached to the host computer where the EC2 instance is running.
  - If the instance stops data stored here is deleted.
- Other options for storage ( not for the root volume) -
  - Mount an Amazon Elastic File System ( Amazon EFS) file system.
  - Connect to Amazon Simple Storage Service ( Amazon S3).( object storage service that offers scalability, data availability, security, and performance.)

## Section 4 : Amazon EC2 Part 3

### 7. Add tags

A tag is a label that you can assign to an AWS resource. Consists of a key and an optional value.

Tags enable you to categorize your AWS resources in different ways, for example, by purpose, owner, or environment. This is useful when you have many resources of the same type — you can quickly identify a specific resource based on the tags you have assigned to it. Each tag consists of a Key and a Value, both of which you define. You can define multiple tags to associate with the instance if you want to.

Tagging is how you can attach metadata to an EC2 instance. Potential benefits of tagging — Filtering, automation, cost allocation, and access control. For example, a commonly used tag for EC2 instances is a tag key called Name and a tag value that describes the instance such as WebServer1. The name tag is exposed by default in the Amazon Console instances page. It's best practice to develop tagging strategies.

### 8. Security group settings

A security group acts as a virtual firewall rules that control traffic to the instance. It exists outside of the instance's guest OS. When you launch instance you can specify one or more security groups. Otherwise the default security group is used. You can create rules that specify the source and which ports that network



communications can use. The source can be an IP address, or an IP address range, another security group, a VPC endpoint, or anywhere which means that all sources will be allowed. By default a security group includes an outbound rule that allows all outbound traffic.

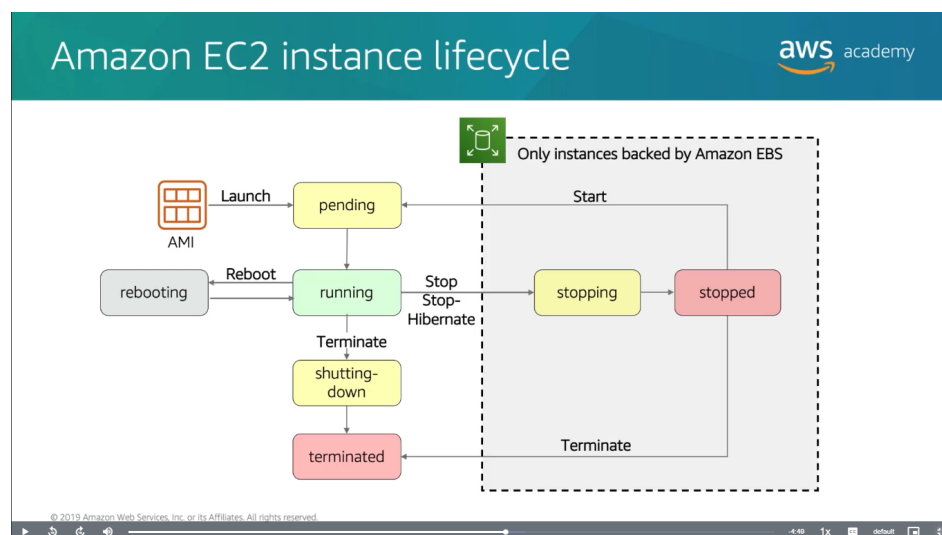
## 9. Identify or create the Key pair

A key pair consists of a public key (that AWS stores) and a private key (that you store). It enables secure connections to an EC2 instance. When creating an EC2 instance, you can choose an existing key pair, proceed without a key pair, or create a new key pair. If you create a new key pair download it and save it in a safe location. The only opportunity you have to download and save the new private key is at the time of creation. To connect to a Windows instance, you use the private key to obtain the administrator password and then log into the EC2 instance's Windows desktop by using Remote Desktop Protocol or RDP. To establish an SSH connection from a Windows machine to an Amazon EC2 instance. You can use a tool such as PuTTY which will require the same private key.

## Another option to Launch an EC2 instance

EC2 instances can also be created programmatically through AWS command line interface or through one of the AWS SDKs.

## Amazon EC2 instance lifecycle



- Rebooting an instance will not change any IP addresses or DNS hostnames.
- When an instance is stopped and then started again-
  - The public IPv4 address and external DNS hostname will change.
  - The private IPv4 address and internal DNS hostname do not change.
- If you require a persistent public IP address -
  - Associate an Elastic IP address with the instance.
- Elastic IP address characteristics-
  - Can be associated with instances in the Region as needed.
  - Remains allocated to your account until you choose to release it.
  - By default all AWS accounts are limited to five Elastic IP addresses per Region because public internet addresses are scarce. You can request a limit increase.

## EC2 instance metadata

Instance metadata is data about your instance. You can view it while you're connected to the instance.

- In a browser : `http://169.254.169.254/latest/meta-data`
- In a terminal window: `curl http://169.254.169.254/latest/meta-data/`

The IP address 169.254.169.254 is a link-local address and it is valid only from the instance.

## Amazon CloudWatch for monitoring

### Use Amazon CloudWatch to monitor EC2 instances


- Provides near-real-time metrics
- Provides charts in the Amazon EC2 console Monitoring tab that you can view.
- Maintains 15 months of historical data.

Basic Monitoring

Detailed Monitoring

- Default, no additional cost
- Metric data sent to CloudWatch every 5 minutes
- Fixed monthly rate for seven pre-selected metrics.
- Metric data is delivered every 1 minute

## Section 5 : Amazon EC2 pricing models



# Amazon EC2 pricing models

### On-Demand Instances

- Pay by the hour
- No long-term commitments.
- Eligible for the [AWS Free Tier](#).

### Reserved Instances

- Full, partial, or no upfront payment for instance you reserve.
- Discount on hourly charge for that instance.
- 1-year or 3-year term.

### Spot Instances

- Instances run as long as they are available and your bid is above the Spot Instance price.
- They can be interrupted by AWS with a 2-minute notification.
- Interruption options include terminated, stopped or hibernated.
- Prices can be significantly less expensive compared to On-Demand Instances
- Good choice when you have flexibility in when your applications can run.

### Dedicated Hosts

- A physical server with EC2 instance capacity fully dedicated to your use.

### Scheduled Reserved Instances

- Purchase a capacity reservation that is always available on a recurring schedule you specify.
- 1-year term.

### Dedicated Instances

- Instances that run in a VPC on hardware that is dedicated to a single customer.

Per second billing available for On-Demand Instances, Reserved Instances, and Spot Instances that run Amazon Linux or Ubuntu.

© 2019 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

## Amazon EC2 pricing models : Benefits

On-demand Instances:

1. Low cost and flexibility

Spot instances :

1. Large scale, dynamic workload

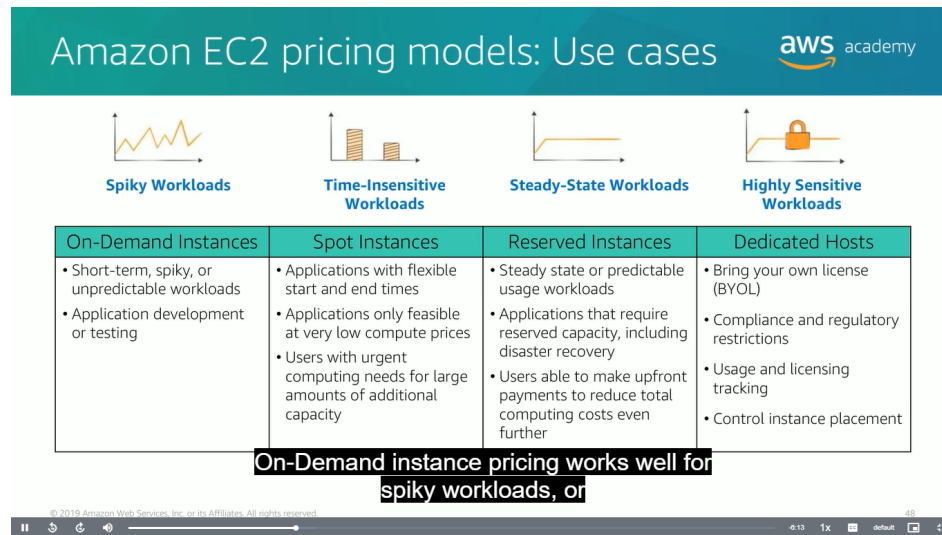
Reserved Instances:

1. Predictability ensures that compute capacity is available when needed

Dedicated Hosts:

1. Save money on licensing costs.
2. Help meet compliance and regulatory requirements

## Amazon EC2 pricing models : Use cases



## Four pillars of cost optimization

|            |                     |                       |                          |
|------------|---------------------|-----------------------|--------------------------|
| Right Size | Increase elasticity | Optimal pricing model | Optimize storage choices |
|------------|---------------------|-----------------------|--------------------------|

### 1. Pillar 1 : Right size

AWS offers more than 60 instance types and sizes. The broad selection enables customers to choose the instance type that best fits their workload. Provision instances to match the need

- CPU, memory, storage, and network throughput
- Select appropriate instance types for your use

Review your deployed resources look for opportunities to downsize when possible.

Use Amazon CloudWatch metrics to review CPU, RAM, storage, and network utilization and identify the instances that could be downsized.

Select the most cost-effective instance type available that still meets your performance requirements.

And if your usage pattern will be consistent, purchase Reserved instances.

### 2. Pillar 2 : Increase elasticity

One form of elasticity is to stop or hibernate Amazon EBS-backed instances that are not actively in use.

Common candidates for stopping or hibernating are non-production development environments, development workloads, or test workloads.

For production workloads, configuring more precise and granular automatic scaling policies can help you take advantage of horizontal scaling to meet peak capacity needs and to not pay for peak capacity all the time.

### 3. Pillar 3 : Optimal Pricing model

Leverage the right pricing model for your use case. You can combine multiple purchase types to optimize pricing based on your current and forecast capacity needs.

Examples:

- Use On-Demand Instance and Spot Instances for variable workloads.
- Use Reserved Instances for predictable workloads.

Consider serverless solutions (AWS Lambda)

### 4. Pillar 4 : Optimize storage choices

It is a best practice to try to reduce costs while maintaining storage performance and availability. One way you can accomplish this is by resizing EBS volumes.

Customers often use EBS snapshots to create data backups. However, some customers forget to delete snapshots that are no longer needed. Delete these unneeded snapshots to save on storage costs.

Finally, identify the most appropriate destination for specific types of data.

- Does the application need the data that it uses to reside on Amazon EBS?
- Would the application run equally as well if it used Amazon S3 for storage instead ?

Configuring data life cycle policies can also reduce costs. For example, you might automate the migration of older infrequently used data to more cost effective storage locations, such as Amazon Simple Storage Service Glacier.

## Measure, Monitor, and Improve

Cost optimization is ongoing process.

Recommendations -

- Define and enforce cost allocation tagging.
- Define metrics, set targets, and review regularly.
- Encourage teams to architect for cost.
- Assign the responsibility of optimization to an individual or to a team.
- AWS Cost Explorer is a free tool that you can use to view graphs of your costs.

## Section 6:

### Container basics

- Containers are a method of operating system virtualization. Containers are smaller than virtual machines and do not contain an entire operating system. Instead, containers share a virtualized operating system and run as resource isolated processes.
- Benefits:
  - They support repeatability.
  - Deliver environmental consistency because the application's code, configurations, and dependencies are packaged into self-contained environments.
  - They can help ensure that applications deploy quickly, reliably, and consistently, regardless of the deployment environment.
  - Lastly in terms of space container images are usually an order of magnitude smaller than virtual machines.
  - Spinning up a container happens in hundreds of milliseconds resulting in faster launch times than traditional virtual machines

## What is Docker ?

- Docker is a software platform that packages software such as applications into containers.
- Docker is installed on each server that will host containers and it provides simple commands that you can use to build, start, or stop containers.
- Containers are created from templates called Docker images.
- Containers have everything that an application you want to run needs, including libraries, system tools, and the application code, as well as any runtime libraries.

## Containers versus virtual machines

Virtual machines run directly on a hypervisor while containers run on any operating system, if they have the appropriate kernel features to support Docker host software, and the Docker daemon is present.

In an actual container-based deployment, a large EC2 instance could run hundreds of containers.

## Amazon Elastic Container Service ( Amazon ECS)

You could launch one or more Amazon EC2 instances, install Docker on each instance, and manage and run the Docker containers on those Amazon EC2 instances yourself.

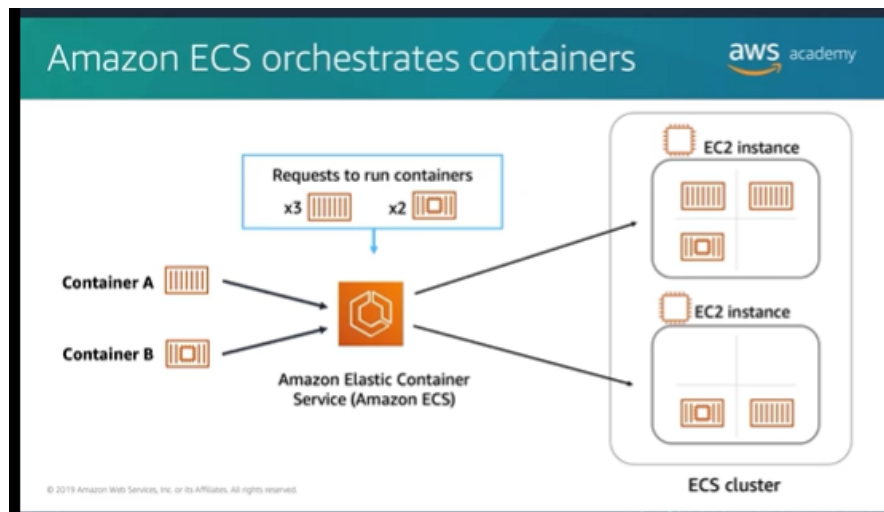
While that's an option. AWS provides a service called Amazon Elastic Container Service, or Amazon ECS, that simplifies container management.

Amazon ECS is highly scalable, fast, container management service that orchestrates the running of Docker containers.

It manages the fleet of nodes, or EC2 instances, that run your containers, removing the complexity of managing the infrastructure yourself.

Amazon ECS provides a familiar set of features to anyone who uses the EC2 service, including elastic load balancing, security groups, EBS volumes, and IAM roles.

## Amazon ECS orchestrates containers



To prepare your application, you create a task definition which is considered a blueprint of your application. Task definitions specify details, such as which containers should be deployed to run the task. You specify the number of tasks that will run on your cluster and Amazon ECS task scheduler is responsible for placing tasks within your cluster. When Amazon ECS runs the container based on your task, it places them on an ECS cluster. When you choose the EC2 Launch type, the cluster consists of a group of EC2 instances ,each of which is running an Amazon ECS Container Agent.

### Amazon ECS cluster options

Key question : Do you want to manage the Amazon ECS cluster that runs the containers?

- If yes, create an Amazon ECS cluster backed by Amazon EC2 ( provides more granular control over infrastrucuture)
- If no, create an Amazon ECS cluster backed by AWS Fargate ( easier to maintain, focus on you applications)

### What is Kubernetes?

Kubernetes is open source software for container orchestration,

- Deploy and manage containerized applications at scale.



- The same toolset can be used on premises and in the cloud.

## **Complements Docker**

- Docker enables you to run multiple containers on a single OS host.
- Kubernetes orchestrates multiple Docker hosts (nodes)

## **Automates**

- Container provisioning
- Networking
- Load distribution
- Scaling.

## **Amazon Elastic Kubernetes Service (Amazon EKS)**

Amazon Elastic Kubernetes Service

- Enables you to run Kubernetes on AWS
- Certified Kubernetes conformant ( supports easy migration)
- Supports Linux and Windows containers
- Compatible with Kubernetes community tools and supports popular Kubernetes add-ons.

Use Amazon EKS to -

- Manage clusters of Amazon EC2 compute instances
- Run containers that are orchestrated by Kubernetes on those instances.

## **Amazon Elastic Container Registry ( Amazon ECR)**

Amazon ECR is a fully managed Docker container registry that makes it easy for developers to store, manage, and deploy Docker container images.

- Amazon ECS integration
- Docker support
- Team collaboration

- Access control
- Third-party integrations

## Section 7 : Introduction to AWS Lambda

AWS Lambda is an event-driven serverless compute service. Lambda enables you to run code without provisioning or managing servers. You create a Lambda function which is the AWS resource that contains the code that you want to run. You then set the Lambda function to be triggered either on a schedule or in response to an event. Your code only runs when it's triggered. You pay only for the compute time you consume. You are not charged when your code is not running. With Lambda, there are no new languages, tools, or frameworks to learn. Lambda supports multiple programming languages including Java, Go, PowerShell, Node.js, C#, Python and Ruby..

Lambda completely automates the administration. It manages all the infrastructure to run your code on highly-available fault tolerant infrastructure which enables you to focus on building differentiated backend services. Lambda provides built-in fault tolerance. It maintains compute capacity across multiple Availability Zones in each Region to help protect your code against individual machine failures or data center failures. You can orchestrate multiple Lambda functions for complex or long-running tasks by building workflows with AWS step functions. With Step Functions and Lambda, you can build stateful, long-running processes for applications and backends. Finally, with Lambda you pay only for the requests that are served and the compute time that is required to run your code. Billing is metered in increments of 100 milliseconds which makes it cost effective to run code on AWS Lambda.

### AWS Lambda event sources

Lambda functions are triggered by event sources. An event source can be an AWS service or developer-created application that produces events that trigger the AWS Lambda function to run.

Some services such as Amazon S3, Amazon Cloudwatch events

AWS Lambda has specific limits on how long a function can run:

- **Maximum Execution Time:** A Lambda function can run for a maximum of **15 minutes** (900 seconds) per invocation. If your function exceeds this time limit, it will be automatically terminated.

This limit means that Lambda is suitable for short-lived tasks, but it's not ideal for processes that require long-running computations or operations. If you need to run tasks that exceed 15 minutes, you might consider breaking them into smaller tasks or using other AWS services like AWS Step Functions or AWS EC2.

## Section 8 : Introduction to AWS Elastic Beanstalk

Elastic Beanstalk is another AWS compute service option. It provides an easy way to get web applications up and running on the AWS Cloud. AWS Elastic Beanstalk automatically handles infrastructure provisioning and configurations, deployment, load balancing, automatic scaling, health monitoring, analysis and debugging, and logging.

You upload your code and Elastic Beanstalk automatically handles the deployment, from capacity provisioning to load balancing, to automatic scaling and monitoring and application health.

When you use AWS Elastic Beanstalk, you retain full control over the AWS resources that power your application, you can choose to access the underlying resources at any time. There is no additional charge for AWS Elastic Beanstalk. You only pay for the AWS resources such as EC2 or S3 Buckets, that you create and store and run your application on.

You only pay for what you use, as you use it.

### AWS Elastic Beanstalk deployments

AWS Elastic Beanstalk enables you to deploy your code through the AWS Management Console, the AWS Command Line Interface, Visual Studio, or Eclipse. It supports a broad range of platforms, including Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker

Elastic Beanstalk provides all the application services that you need for your application. The only thing you must create is your code.

AWS Elastic Beanstalk deploys your code on several popular application web servers.

## Benefits of Elastic Beanstalk

- Fast and simple to start using : You use the AWS Management Console, a Git repository, or an IDE such as Eclipse or Visual Studio, to upload your application.
- You can improve developer productivity by focusing on writing code instead of configuring servers, databases, load balancers, firewalls, and networks.
- Difficult to outgrow. Your application can handle peaks in workload or traffic. It automatically scales your application up and down based on your application's specific needs, by using easily adjustable automatic scaling settings. You can even use CPU utilization metrics to trigger automatic scaling actions.
- Complete resource control

## The End

Thanks

2081/05/02