

EXPLORE || DIGITAL SKILLS

AWS Compute Services

Train Overview

In this train we will cover the following:

We will touch on the **basics of compute services** in the cloud. We will then take a deep dive into concepts around **virtual machines, containers and serverless compute**. Finally, we consider when to use each respective compute form, highlighting AWS compute offerings which fulfil these scenario requirements.

Introduction to Compute Services

Amazon EC2 Instances

Container Services

Overview of AWS Lambda

Overview of AWS Elastic Beanstalk

Conclusion



What are Compute Services?

Compute services refer to the provision of computer processing power over the internet.



Scalable

You can spin up computing clusters of any size to handle your computation



Secure

Your data is encrypted and your system safe from malicious attacks






Flexible

The ability to create/destroy instances on demand in a planned or unplanned manner.





AWS Compute Offerings

AWS offers mainly three types of compute services: 1) Virtual machines, 2) Containers, and 3) Serverless

Compute Service Offering	Service Description	AWS Service
Virtual Machines (VMs)	A VM is a fully-fledged computer system (with networking, compute and storage resources) which is simulated within a physical computer. VMs allow a single physical server to be partitioned into multiple compute services.	 Amazon EC2
Containers	Containers provide a way of packaging software and all of its dependencies in a single, stand-alone 'image' which can be run on widely-varying hardware/operating systems.	 Amazon ECS
Serverless	In serverless computing, a cloud vendor provides access to scalable compute resources upon which code or applications can be run without needing to manage the underlying infrastructure.	 Amazon Lambda

Considerations when Choosing a Computing Service in the Cloud

The best computing service is the one that adheres to your specific design, financial, regulatory and support requirements

Requirement Category	Main Considerations
 Design Requirements	<ul style="list-style-type: none">• Infrastructure• Storage• Client base size/region• Compute • Cost per user• Total Cost of Ownership• Operational cost• Maintenance • Data protection• Data access• Data backup • Implementation• Maintenance• Customer support
 Financial Requirements	
 Regulatory Requirements	
 Support Requirements	

Introduction to Compute Services

Amazon EC2 Instances

Container Services

Overview of AWS Lambda

Overview of AWS Elastic Beanstalk

Conclusion



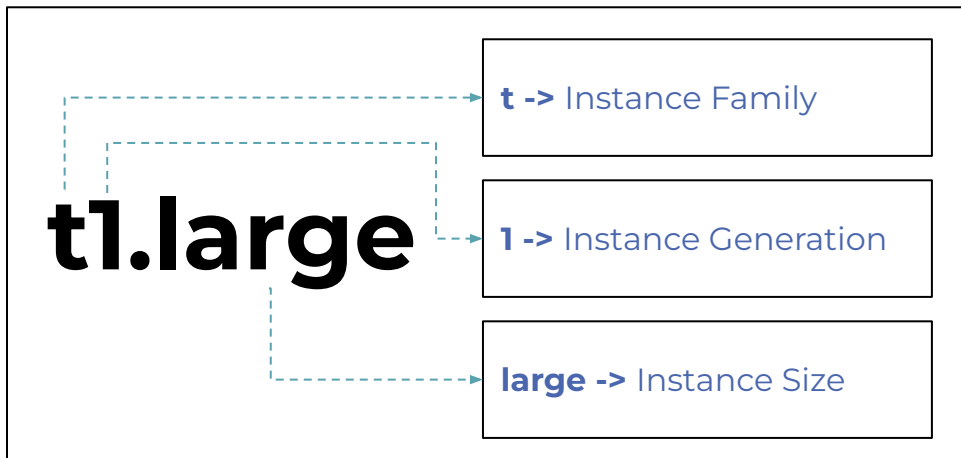
Overview of Amazon EC2 Service

Amazon EC2 (Elastic Cloud Compute) is a web service that provides compute capacity in the cloud



EC2 Instances Naming Conventions

The EC2 naming convention follows a simple structure: first the family, then the generation and finally the size of the instance is defined



General-Purpose: M1, M3, T2, M4



Compute-Optimised: C1, CC2, C3, C4



Memory-Optimised: M2, CR1, R3 {X1}



Dense-Storage: HS1, D2



I/O-Optimised: HI1, I2



GPU: CG1, G2



Micro: T1, T2

EC2 Purchasing Options

You can optimise your EC2 cost by choosing the best combination of billing models for your specific case

	1	2	3	4
	On-Demand	Reserved	Spot	Dedicated
Definition	Pay for the compute capacity you use by the hour	Reserve compute capacity by paying up-front. This reduces the hourly charge significantly	An unused EC2 instance that is available for less than the On-Demand price	Spin up EC2 instances in an Amazon VPC for a specific customer/client
Use Case	Volatile workloads where we cannot forecast the actual demand	Predictable or constant workload demands	For cost sensitive workloads without availability guarantees	Sensitive or compliance related workloads

Introduction to Compute Services

Amazon EC2 Instances

Container Services

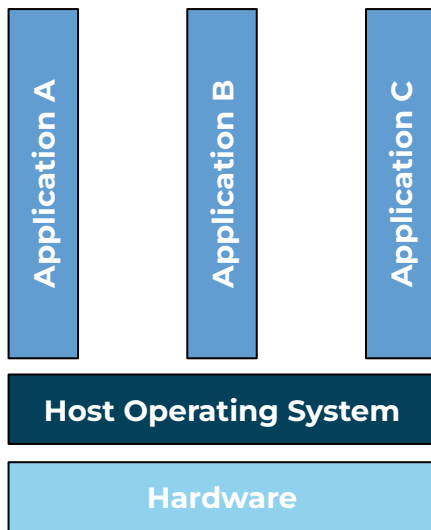
Overview of AWS Lambda

Overview of AWS Elastic Beanstalk

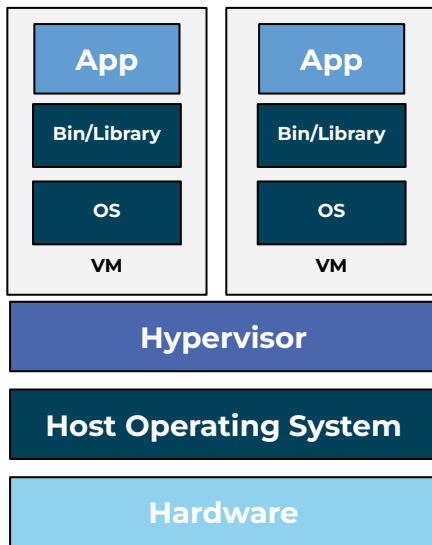
Conclusion

Containers vs. VMs

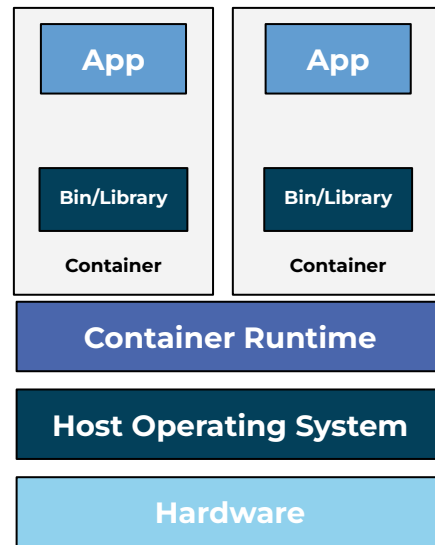
While a virtual machine virtualises an entire computer (OS, hardware, networking, etc), a container virtualises an entire operating system. The significant difference between containers and VMs is that containers share the host's operating system with other containers.



Traditional Deployment



**Virtual Machine (VM)
Deployment**



Container Deployment

But What are Containers?

Containers Overview

Containers are used to package a piece of software or application along with its dependencies in a dense, stand-alone unit.

Containers solve the problem of reliably running software or applications when moving from one computing environment (Operating System and Hardware) to another

Advantages of Using Containers

- Version controlled
- Smaller in size compared to virtual machines
- Almost instantaneous start-up
- A single container image can run on multiple computing environments
- Ease of replication
- Improved modularity (divide complex application into various containerised modules)

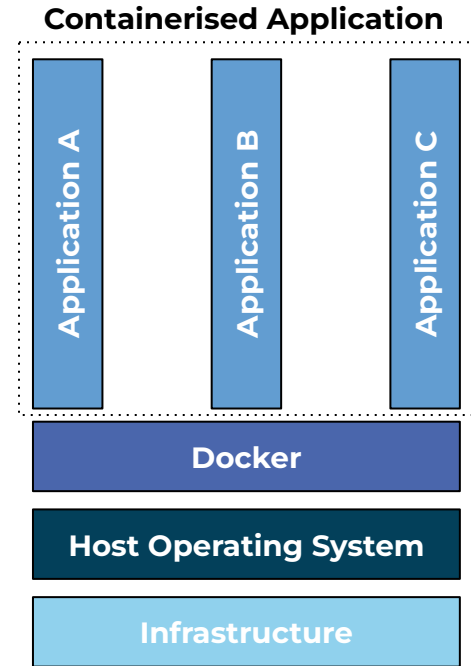
Introduction to Docker

Docker is an open source application that can be used to containerise an application or piece of software. Docker does this by creating a container image of your application - known as a Docker container image.

When you use Docker for containerisation, images become containers when they run on the docker engine.

When Docker Containers are running on the Docker Engine:

1. The container is portable and can run on any Operating System (OS).
2. The containers are lightweight and multiple container applications, can be run on a single host operating system.
3. Applications are safe, secure and isolated.



Container Orchestration Overview

Containers are lightweight and typically only run for a short time. To run a full application in production may translate to running thousands of individual containers - which can become a complex exercise if done manually. To simplify this process we can make use of a container orchestration service.

Container Orchestration Overview

Container orchestration is the automation of the work required to run containerised workloads, applications and services.

Container orchestration typically involves:

- Load balancing
- Networking
- Scaling
- Deployment

Container Orchestration Tools



Amazon ECS



Docker Swarm



Azure Container Service



Kubernetes



Google Container Engine



Amazon ECS

Amazon Web Services' managed container orchestration service



Amazon EKS

AWS-managed Kubernetes service that simplifies the task of deploying a Kubernetes control-plane

Overview of Amazon ECS

Amazon ECS is a highly scalable, high performant container management and orchestration system which supports Docker images



What is Kubernetes?

An open-source system for automating deployment, scaling and managing containerised applications or software

Scaleable

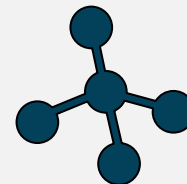
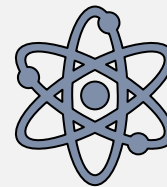
Kubernetes, or K8s, can scale to your requirements. Whether it is running hundreds or billions of containers a week, K8s can scale to handle the load.

Flexible

K8s can adapt to your solution complexity. From single container applications to solutions requiring elaborate orchestration, K8s is built to support any use case.

Multimodal

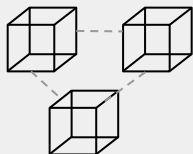
You can use K8s to serve your application over the cloud, through on-premise infrastructure or via a hybrid model.



Overview of Amazon EKS

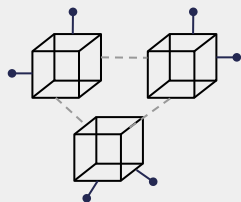
When it comes to containers, it matters where you run your container management service

How to **Run** AWS EKS



Provision EKS Cluster

Create an Amazon EKS cluster.



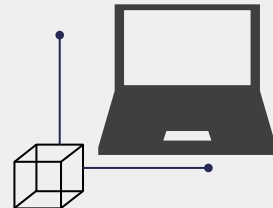
Deploy Compute Instances

Launch managed or self-managed Amazon EC2 nodes, or deploy your workloads to AWS Fargate.



Connect to Amazon EKS

When your cluster is ready, you can configure your favorite Kubernetes tools, such as kubectl, to communicate with your cluster.



Run K8s Application

Deploy and manage workloads on your Amazon EKS cluster the same way that you would with any other Kubernetes environment.

Amazon Elastic Kubernetes Service (EKS) is an Amazon **managed service** which can be used to **run Kubernetes**. Amazon EKS takes care of **updating, managing and maintaining your Kubernetes plane and nodes**.

Introduction to Compute Services

Amazon EC2 Instances

Container Services

Overview of AWS Lambda

Overview of AWS Elastic Beanstalk

Conclusion



What is Serverless Compute?

Serverless compute simplifies the deployment process, enabling developers to focus on developing the solution

Serverless Compute Overview

Serverless compute providers allow a user to write and deploy code without them needing to think of the underlying infrastructure. With serverless compute you only take your code to the serverless service, and they take care of the rest i.e. computing, networking, security, scaling, etc.

Advantages of Serverless Compute

- **Lower Costs:** The user does not pay for unused or idle CPU
- **Easily Scale:** The serverless vendor handles scaling with demand
- **Quick Turnaround:** Simple deployment means reduced time to market



Amazon Elastic Beanstalk

Ideal for deploying and managing fully functional applications



Amazon Lambda

Deployment of small applications or pieces of larger applications at a low cost

Overview of AWS Lambda

AWS Lambda is a fully managed Amazon service for serverless compute

AWS Lambda allows you to run code without provisioning or managing servers.

With AWS Lambda:

1. You only pay for the compute time you consume. If there is no code running, you do not incur any charges.
2. You only need to provide the code. AWS Lambda takes care of all the system and infrastructure requirements for efficient delivery.



Bring Your Own Code: Node.js, Java, Python, Scala, etc.



Flexible Use: Call or send events, Build serverless ecosystems



Stateless: Persistent data using Amazon S3, dynamoDB, etc.



Programming Model: AWS SDK built in for Python and Node

Introduction to Compute Services

Amazon EC2 Instances

Container Services

Overview of AWS Lambda

Overview of AWS Elastic Beanstalk

Conclusion



Overview of AWS Elastic Beanstalk

AWS Elastic Beanstalk is an easy to use service for deploying, scaling, and managing web applications and services

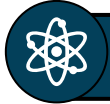
AWS Elastic Beanstalk Requirements



Your Code



Solution Region



Platform Type



**Single Instance or
Load Balance with Auto Scaling**



(Optional) RDS Database

Problems Solved with AWS Elastic Beanstalk

- Reduces complexity of deploying code, provisioning, and managing infrastructure
- Simplifies and automates application scaling
- Enhances the consistency across teams by providing a simple, standardised PaaS solution for deploying, scaling and managing web apps
- Compresses the time from ideation and development to deployment

How to Choose Between AWS EC2, ECS and Lambda?

The choice between EC2, ECS and Lambda boils down to your specific use case and application requirements

Compute Service Offering	Service Description	Example Use Cases
Virtual Machines (VMs)	Amazon EC2: Use when you want control over the machines, storage, networking, and OS	<ul style="list-style-type: none">• Big data• Database software• Enterprise applications• On-premise migration
Containers	Amazon ECS: Use when you want control over the server and application configurations and scaling	<ul style="list-style-type: none">• Web-applications• Docker workloads• Microservices• Batch jobs
Serverless	Amazon Lambda: Use when you want control over running code as and when needed	<ul style="list-style-type: none">• Web-applications• Mobile backends• IoT backends• File processing workloads

Introduction to Compute Services

Amazon EC2 Instances

Container Services

Overview of AWS Lambda

Overview of AWS Elastic Beanstalk

Conclusion



Conclusion

In This Train We've Covered the Following:



What are compute services and what compute services do AWS offer



What is the use case of each compute service in the AWS offering and how to choose which one to use



What are containers, Kubernetes and virtual machines and how these systems differ from one another

Next Steps

- Read the [AWS Documentation](#) on compute services
- Use free-sources such as YouTube to learn more about [Kubernetes and containers](#)
- Study relevant AWS [whitepapers](#)