

AWS Cloud Security Foundations– AWS-CSF

Dr. Ibrahim Gomaa

Lecture 3 – AWS Foundational Services - Compute



مجموعة جامعة
الجميع الذكية
للتعليم القابضة

Lecture 3: AWS Foundational Services - Compute

Topics

- Compute services overview
- Amazon EC2
- Amazon EC2 cost optimization
- Container services
- Introduction to AWS Lambda
- Introduction to AWS Elastic Beanstalk

Activities

- Amazon EC2 versus Managed Service

Demo

- Recorded demonstration of Amazon EC2

Lab

- Introduction to Amazon EC2



Knowledge check



Compute services overview



Services	Key Concepts	Characteristics	Ease of Use
Amazon EC2	- Infrastructure as a service (IaaS) - Instance-based - Virtual machines	Provision virtual machines that you can manage as you choose	A familiar concept to many IT professionals.
AWS Lambda	- Serverless computing - Function-based - Low-cost	- Write and deploy code that runs on a schedule or that can be triggered by events - Use when possible (architect for the cloud)	A relatively new concept for many IT staff members, but easy to use after you learn how.
- Amazon ECS - Amazon EKS - AWS Fargate - Amazon ECR	- Container-based computing - Instance-based	Spin up and run jobs more quickly	AWS Fargate reduces administrative overhead, but you can use options that give you more control.
AWS Elastic Beanstalk	- Platform as a service (PaaS) - For web applications	- Focus on your code (building your application) - Can easily tie into other services—databases, Domain Name System (DNS), etc.	Fast and easy to get started.

Choosing the optimal compute service

- The optimal compute service or services that you use will depend on your use case
- Some aspects to consider –
 - What is your application design?
 - What are your usage patterns?
 - Which configuration settings will you 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

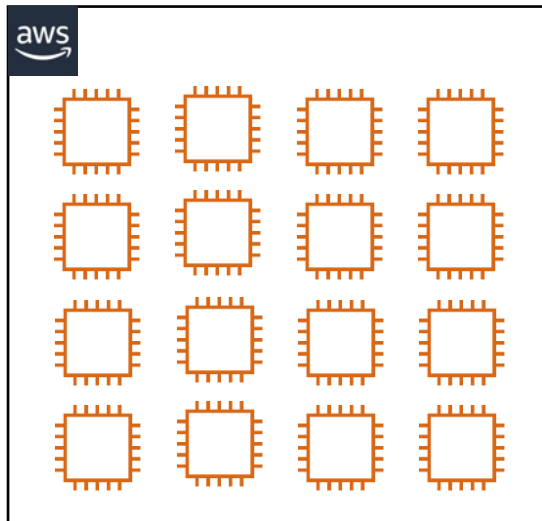
Amazon EC2



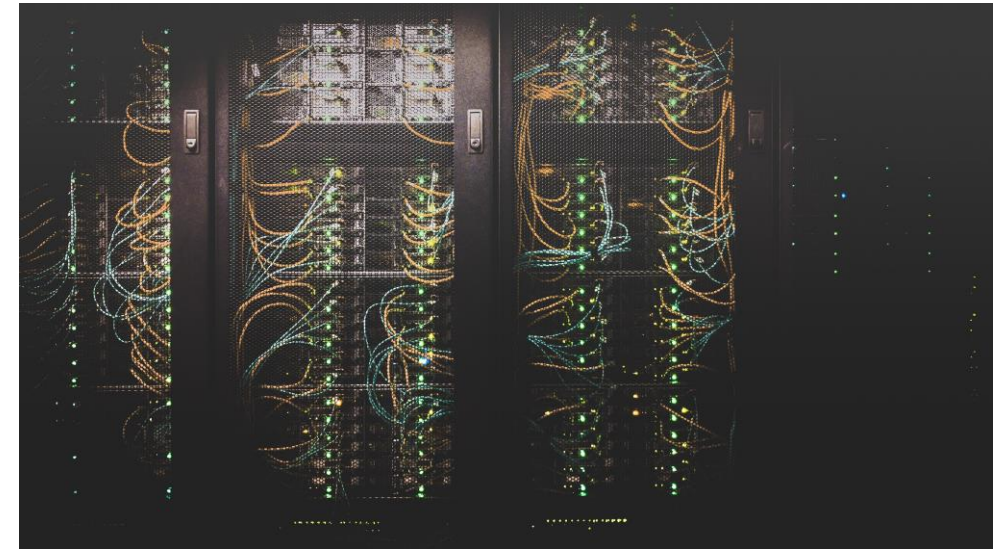
Amazon Elastic Compute Cloud (Amazon EC2)

Amazon Elastic Compute Cloud (Amazon EC2) provides virtual machines where you can host the same kinds of applications that you might run on a traditional on-premises server

Example uses of Amazon EC2 instances



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

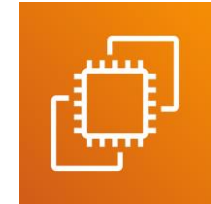


On-premises servers

AWS Cloud Security Foundations– AWS-CSF

Amazon EC2 overview

- **Amazon Elastic Compute Cloud (Amazon EC2)**
 - Provides **virtual machines**—referred to as **EC2 instances**—in the cloud.
 - Gives you *full control* over the guest operating system (Windows or Linux) on each instance.
- You can launch instances of any size into an Availability Zone anywhere in the world.
 - Launch instances from **Amazon Machine Images (AMIs)**.
 - Launch instances with a few clicks or a line of code, and they are ready in minutes.
- You can control traffic to and from instances.

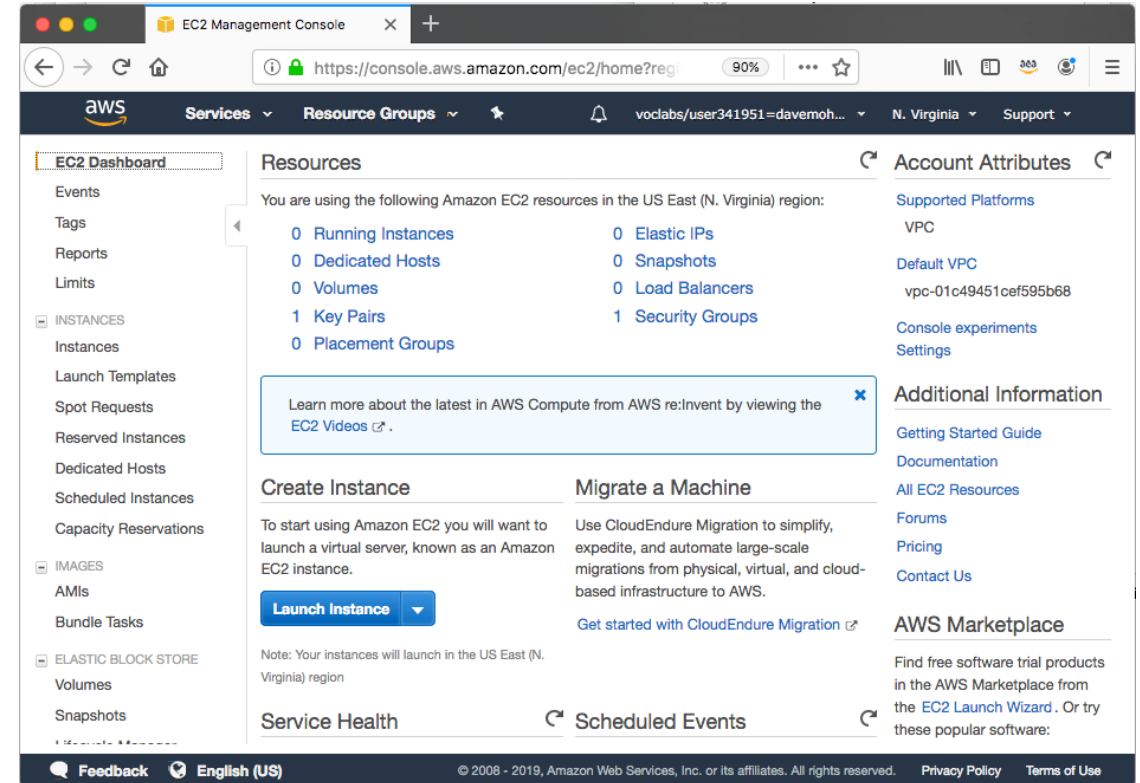


Amazon
EC2

Launching an Amazon EC2 instance

This section of the module walks through **nine key decisions** to make when you create an EC2 instance by using the AWS Management Console **Launch Instance Wizard**.

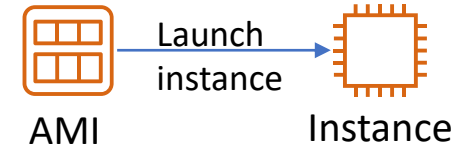
- Along the way, essential Amazon EC2 concepts will be explored.



1. Select an AMI

Choices made using the Launch Instance Wizard:

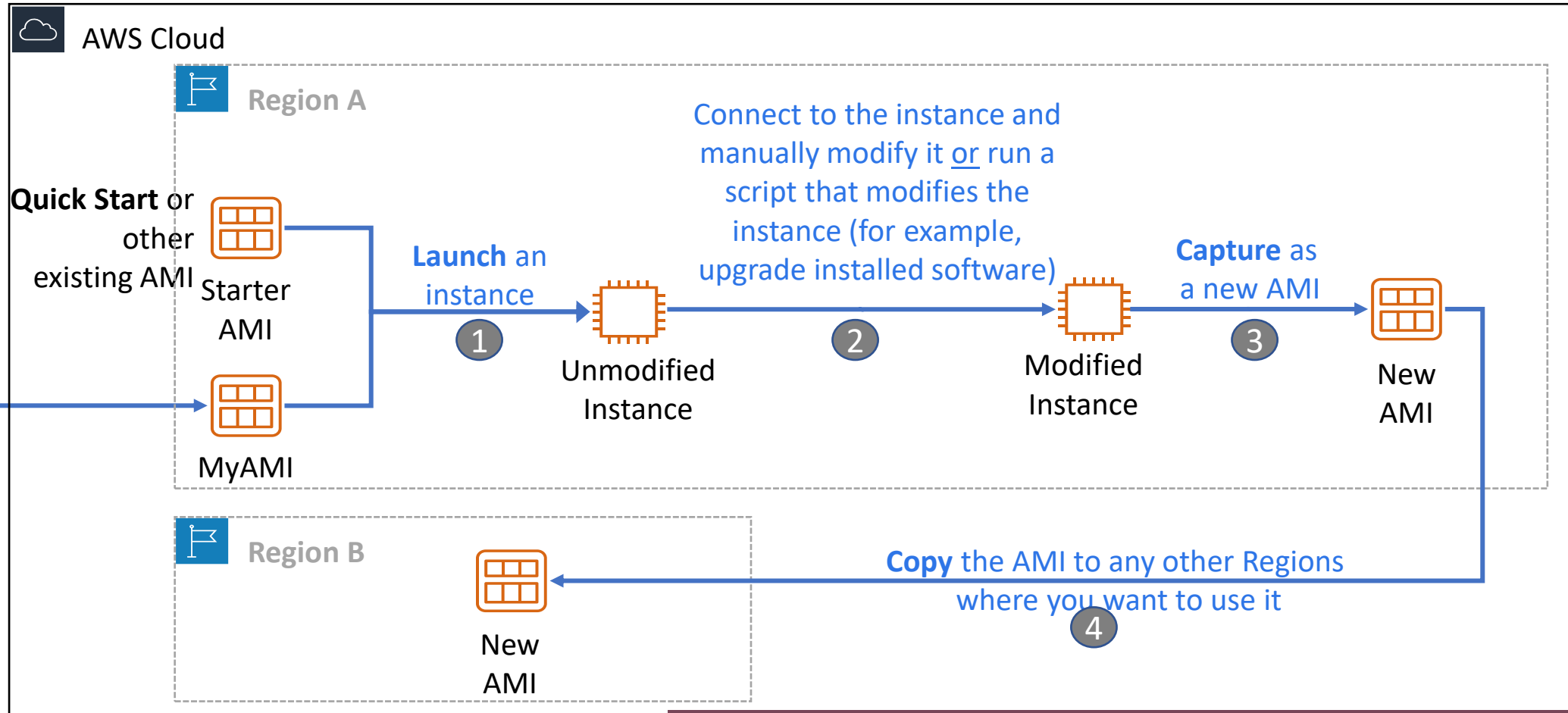
1. **AMI**
2. **Instance Type**
3. **Network settings**
4. **IAM role**
5. **User data**
6. **Storage options**
7. **Tags**
8. **Security group**
9. **Key pair**



- Amazon Machine Image (AMI)
 - Is a template that is used to create an EC2 instance (which is a **virtual machine, or VM**, that runs in the AWS Cloud)
 - Contains a **Windows** or **Linux** operating system
 - Often also has some **software** pre-installed
- AMI choices:
 - Quick Start – *Linux and Windows AMIs that are 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*

Creating a new AMI: Example

AMI details

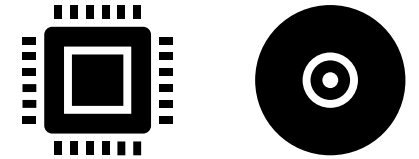
(Optional) Import
a virtual machine

2. Select an instance type

Choices made using the Launch Instance Wizard:

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

- Consider your use case
 - How will the EC2 instance you create be used?
- The **instance type** that you choose determines –
 - Memory (RAM)
 - Processing power (CPU)
 - Disk space and disk type (Storage)
 - Network performance
- 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

Example instance sizes

Instance Name	vCPU	Memory (GB)	Storage
t3.nano	2	0.5	EBS-Only
t3.micro	2	1	EBS-Only
t3.small	2	2	EBS-Only
t3.medium	2	4	EBS-Only
t3.large	2	8	EBS-Only
t3.xlarge	4	16	EBS-Only
t3.2xlarge	8	32	EBS-Only

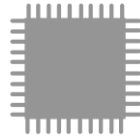
Instance type details

Instance type naming

- Example: **t3.large**
 - **T** is the family name
 - **3** is the generation number
 - **Large** is the size

Select instance type: Based on use case

Instance type details

General
PurposeCompute
OptimizedMemory
OptimizedAccelerated
ComputingStorage
Optimized

Instance Types	a1, m4, m5, t2, t3	c4, c5	r4, r5, x1, z1	f1, g3, g4, p2, p3	d2, h1, i3
Use Case	Broad	High performance	In-memory databases	Machine learning	Distributed file systems

Instance types: Networking features

- The network bandwidth (Gbps) varies by instance type.
 - See [Amazon EC2 Instance Types](#) to compare.
- To maximize networking and bandwidth performance of your instance type:
 - If you have interdependent instances, launch them into a **cluster placement group**.
 - Enable enhanced networking.
- Enhanced networking types are supported on most instance types.
 - See the [Networking and Storage Features](#) documentation for details.
- Enhanced networking types –
 - **Elastic Network Adapter (ENA)**: Supports network speeds of up to 100 Gbps.
 - **Intel 82599 Virtual Function interface**: Supports network speeds of up to 10 Gbps.

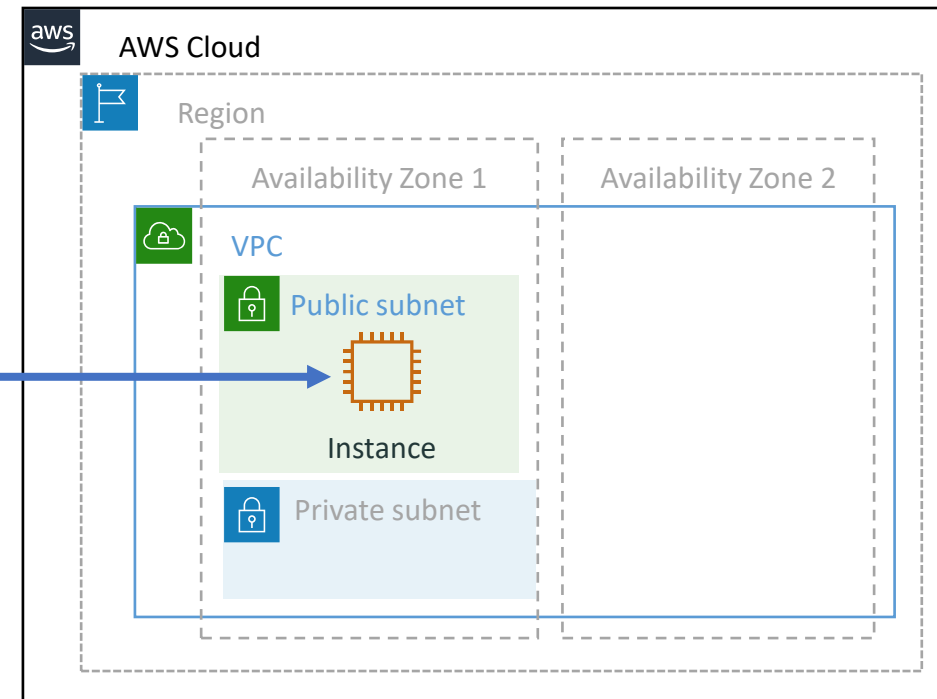
3. Specify network settings

Choices made by using the Launch Instance Wizard:

1. AMI
2. Instance Type
3. **Network settings**
4. IAM role
5. User data
6. Storage options
7. Tags
8. Security group
9. Key pair

- Where should the instance be deployed?
 - Identify the **VPC** and optionally the **subnet**
- Should a **public IP address** be automatically assigned?
 - To make it internet-accessible

*Example: specify
to deploy the
instance here*



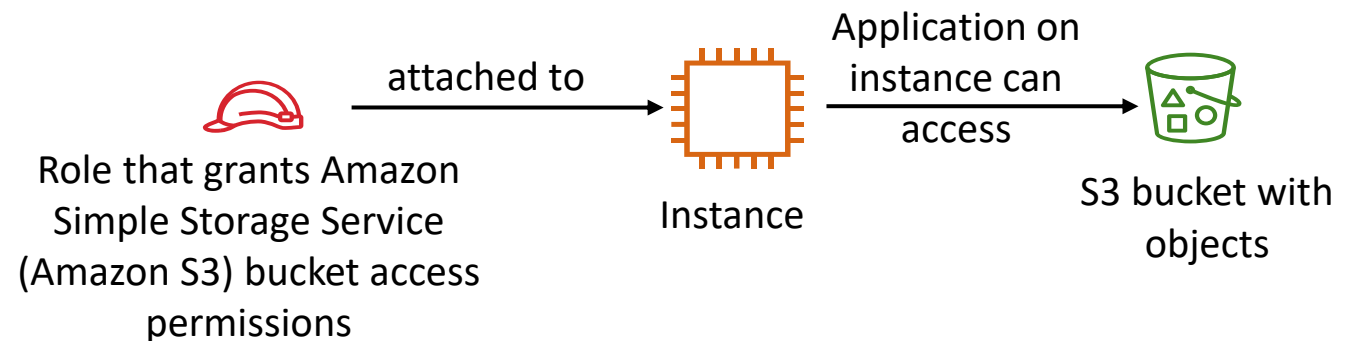
AWS Cloud Security Foundations– AWS-CSF

4. Attach IAM role (optional)

Choices made by using the Launch Instance Wizard:

1. AMI
2. Instance Type
3. Network settings
4. **IAM role**
5. User data
6. Storage options
7. Tags
8. Security group
9. Key pair

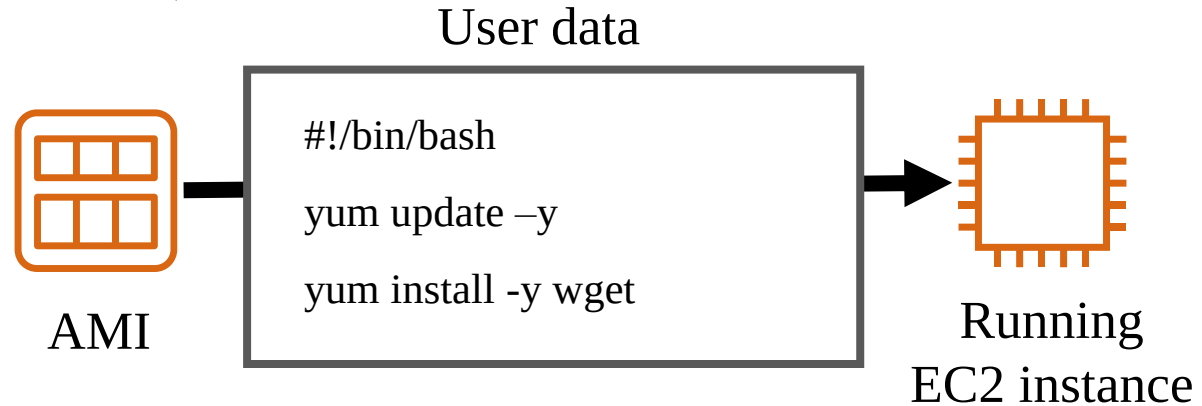
- Will software on the EC2 instance need to interact with other AWS services?
 - If yes, attach an appropriate **IAM Role**.
- An AWS Identity and Access Management (IAM) role that is attached to an EC2 instance is kept in an **instance profile**.
- You are *not* restricted to attaching a role only at instance launch.
 - You can also attach a role to an instance that already exists.



4. Attach IAM role (optional)

Choices made by using the Launch Instance Wizard:

1. AMI
2. Instance Type
3. Network settings
4. IAM role
5. **User data**
6. Storage options
7. Tags
8. Security group
9. Key pair



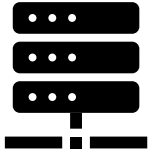
- Optionally specify a user data script at instance launch
- Use **user data** scripts to customize the runtime environment of your instance
 - **Script runs the first time the instance starts**
- Can be used strategically
 - For example, reduce the number of custom AMIs that you build and maintain

6. Specify storage

Choices made by using the Launch Instance Wizard:

1. AMI
2. Instance Type
3. Network settings
4. IAM role
5. User data
6. **Storage options**
7. Tags
8. Security group
9. Key pair

- Configure the **root volume**
 - Where the guest operating system is installed
- Attach **additional storage volumes** (optional)
 - AMI might already include more than one volume
- For each volume, specify:
 - The **size** of the disk (in GB)
 - The **volume type**
 - Different types of solid state drives (SSDs) and hard disk drives (HDDs) are available
 - If the volume will be deleted when the instance is terminated
 - If **encryption** should be used



Amazon EC2 storage options

- **Amazon Elastic Block Store (Amazon EBS)** –
 - **Durable**, block-level storage volumes.
 - You can stop the instance and start it again, and the data will still be there.
- **Amazon EC2 Instance Store** –
 - **Ephemeral** 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)**.



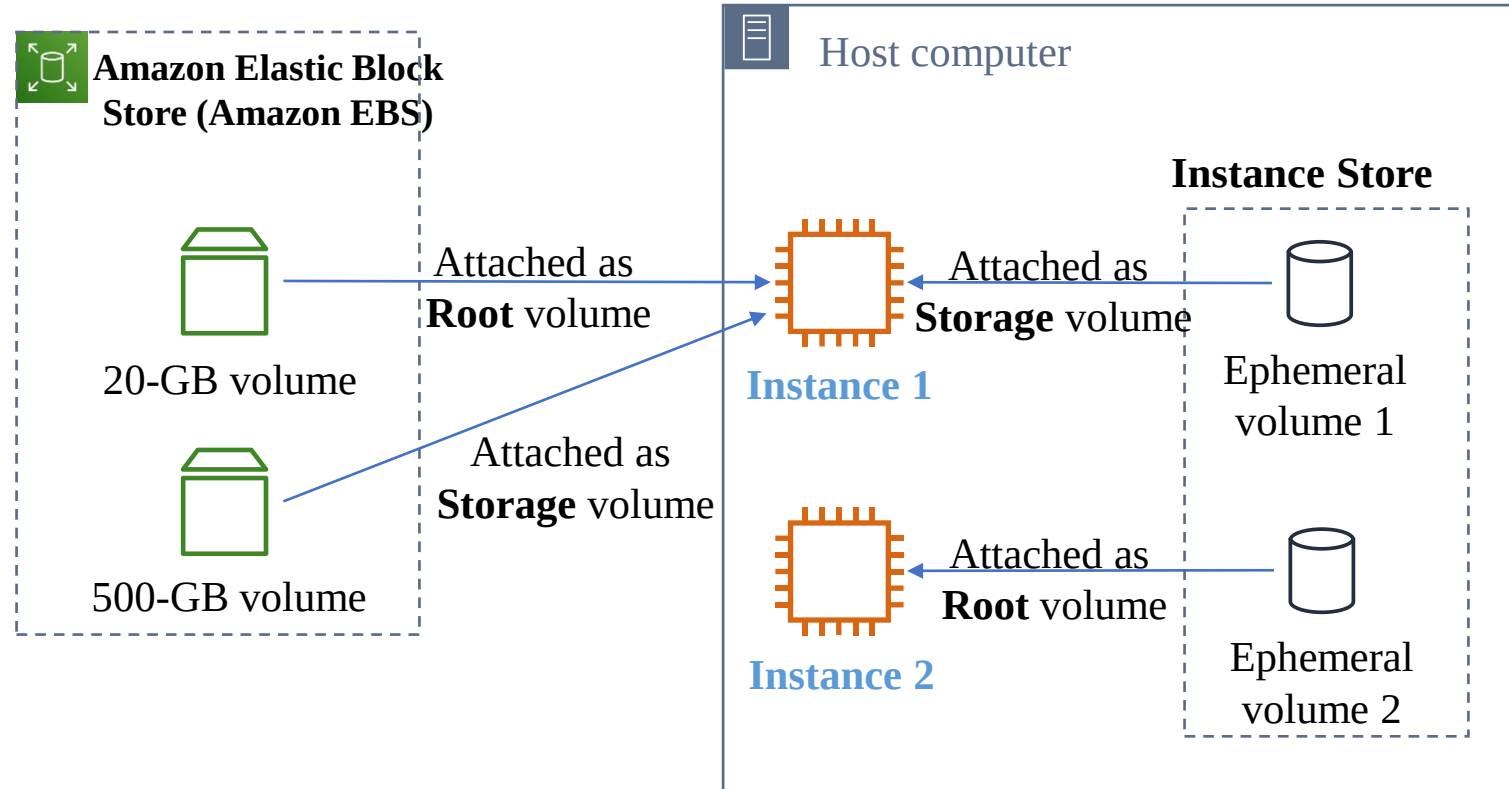
Example storage options

Instance 1 characteristics –

- It has an **Amazon EBS** *root volume* type for the operating system.
- What will happen if the instance is stopped and then started again?

Instance 2 characteristics –

- It has an **Instance Store** *root volume* type for the operating system.
- What will happen if the instance stops (because of user error or a system malfunction)?



7. Add tags

Choices made by using the Launch Instance Wizard:

1. AMI
2. Instance Type
3. Network settings
4. IAM role
5. User data
6. Storage options
7. **Tags**
8. Security group
9. Key pair

- A **tag** is a label that you can assign to an AWS resource.
 - Consists of a *key* and an optional *value*.
- Tagging is how you can attach **metadata** to an EC2 instance.
- Potential benefits of tagging—Filtering, automation, cost allocation, and access control.

Example:

Key (128 characters maximum)	Value (256 characters maximum)
Name	WebServer1
Add another tag (Up to 50 tags maximum)	

8. Security group settings

Choices made by using the Launch Instance Wizard:

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

- A **security group** is a **set of firewall rules** that control traffic to the instance.
 - It exists *outside* of the instance's guest OS.
- Create **rules** that specify the **source** and which **ports** that network communications can use.
 - Specify the **port** number and the **protocol**, such as Transmission Control Protocol (TCP), User Datagram Protocol (UDP), or Internet Control Message Protocol (ICMP).
 - Specify the **source** (for example, an IP address or another security group) that is allowed to use the rule.

Example rule:

Type ⓘ	Protocol ⓘ	Port Range ⓘ	Source ⓘ
SSH ⚙	TCP	22	My IP ⚙ 72.21.198.67/32

9. Identify or create the key pair

Choices made by using the Launch Instance Wizard:

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

- At instance launch, you specify an existing key pair *or* create a new key pair.
- A **key pair** consists of –
 - A **public key** that AWS stores.
 - A **private key** file that you store.
- It enables secure connections to the instance.
- For **Windows AMIs** –
 - Use the private key to obtain the administrator password that you need to log in to your instance.
- For **Linux AMIs** –
 - Use the private key to use SSH to securely connect to your instance.



Activity: Amazon EC2

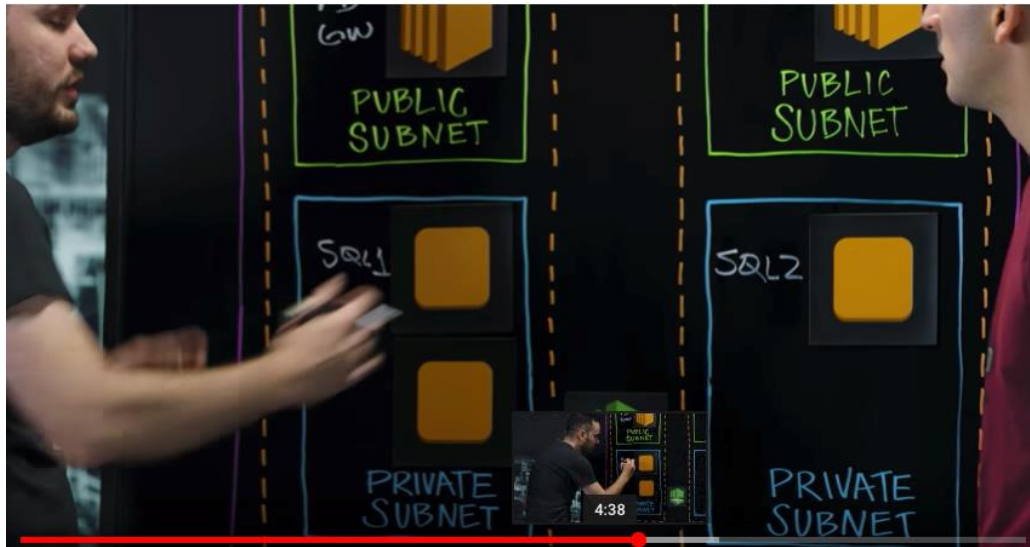
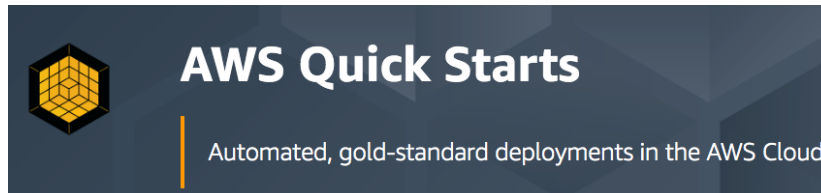


Lecture
3

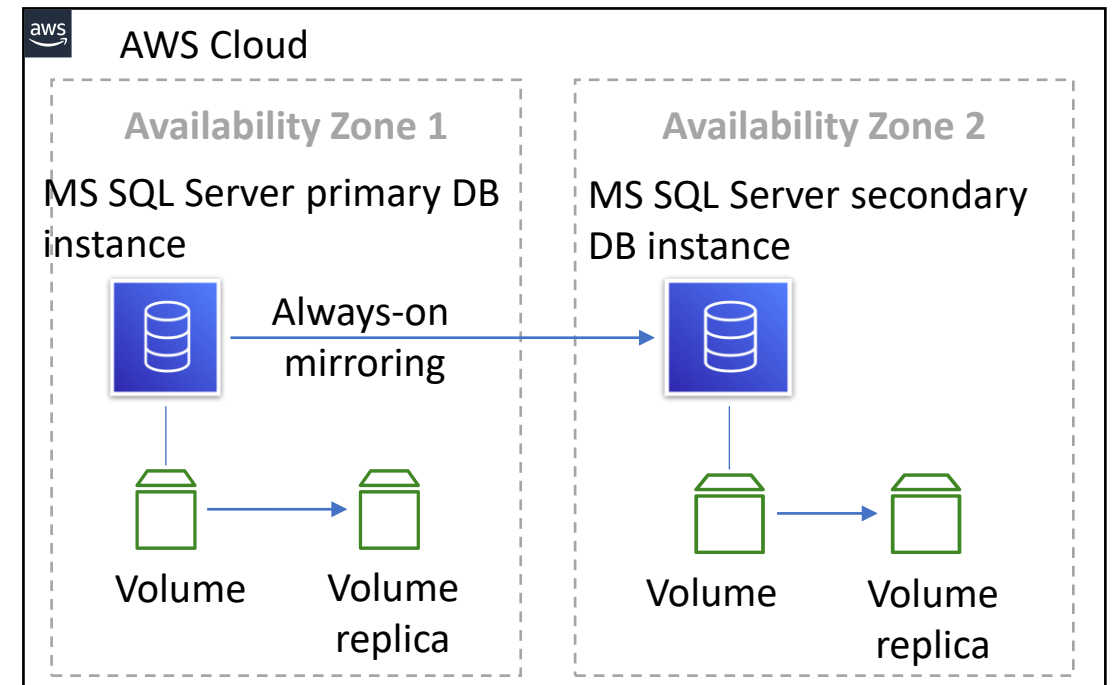
Activity: Amazon EC2



Amazon EC2



Amazon RDS



1. Between Amazon EC2 or Amazon RDS, which provides a managed service? What does *managed service* mean?
 - **ANSWER: ???**
2. Name at least one advantage of deploying Microsoft SQL Server on Amazon EC2 instead of Amazon RDS.
 - **ANSWER: ???**
3. What advantage does the Quick Start provide over a manual installation on Amazon EC2?
 - **ANSWER: ???**
4. Which deployment option offers the best approach for all use cases?
 - **ANSWER: ???**
5. Which approach costs more: using Amazon EC2 or using Amazon RDS?
 - **ANSWER: ???**

Amazon EC2 Cost Optimization



Amazon EC2 pricing models

On-Demand Instances

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

Dedicated Hosts

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

Dedicated Instances

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

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.

Scheduled Reserved Instances

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



Amazon EC2 pricing models: Benefits



On-Demand Instances	Spot Instances	Reserved Instances	Dedicated Hosts
Low cost and flexibility	Large scale, dynamic workload	Predictability ensures compute capacity is available when needed	- Save money on licensing costs - Help meet compliance and regulatory requirements

Amazon EC2 pricing models: Use cases



Spiky Workloads

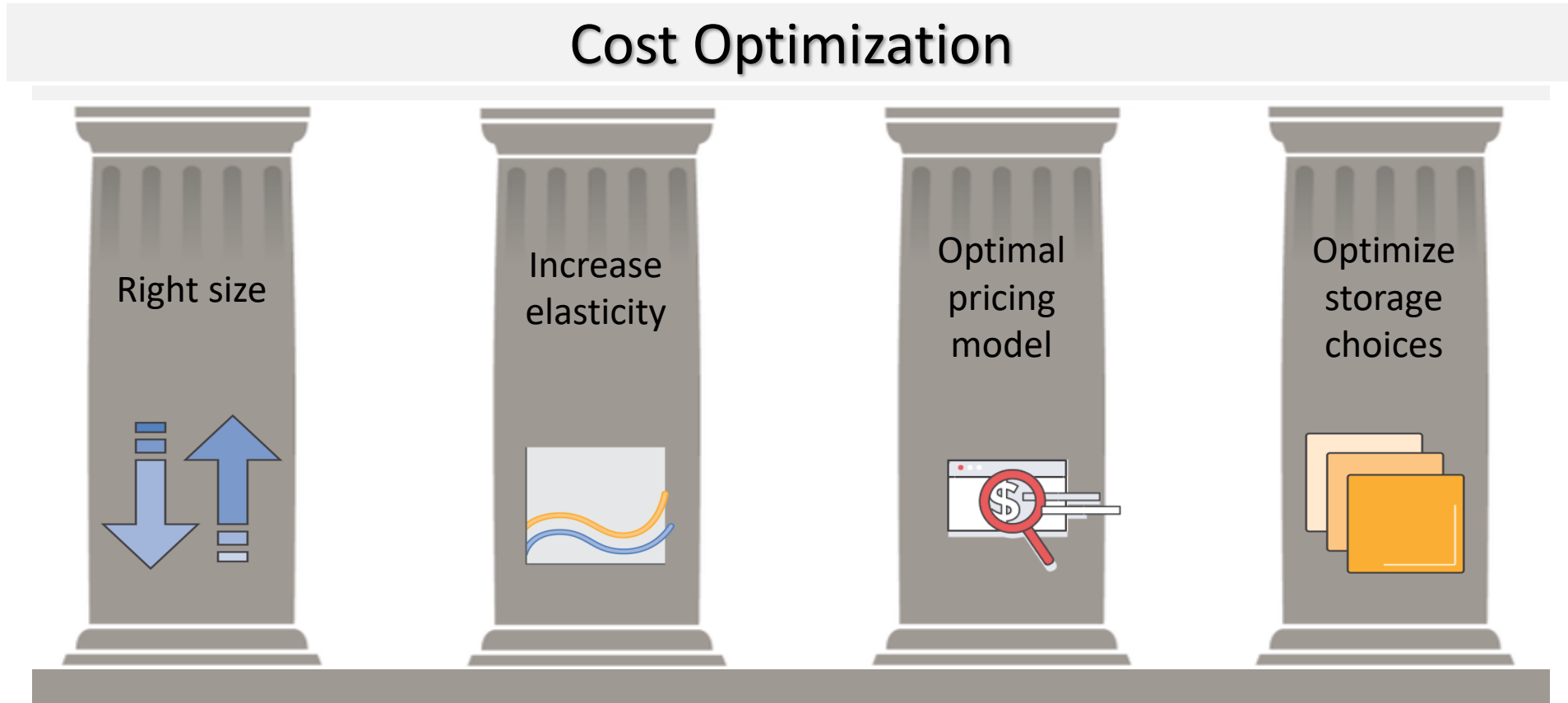
Time-Insensitive
Workloads

Steady-State Workloads

Highly Sensitive
Workloads

On-Demand Instances	Spot Instances	Reserved Instances	Dedicated Hosts
• Short-term, spiky, or unpredictable workloads • Application development or testing	• Applications with flexible start and end times • Applications only feasible at very low compute prices • Users with urgent computing needs for large amounts of additional capacity	• Steady state or predictable usage workloads • Applications that require reserved capacity, including disaster recovery • Users able to make upfront payments to reduce total computing costs even further	• Bring your own license (BYOL) • Compliance and regulatory restrictions • Usage and licensing tracking • Control instance placement

The four pillars of cost optimization



Pillar 1: Right size

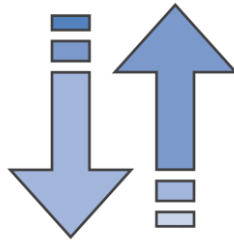
Pillars:

1. Right size

2. Increase elasticity

3. Optimal pricing model

4. Optimize storage choices



✓ Provision instances to match the need

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

✓ Use Amazon CloudWatch metrics

- How idle are instances? When?
- Downsize instances

✓ Best practice: Right size, then reserve

Pillar 2: Increase elasticity

Pillars:

1. Right-Size
2. Increase Elasticity
3. Optimal pricing model
4. Optimize storage choices



- ✓ **Stop** or **hibernate** Amazon EBS-backed instances that are not actively in use
 - Example: non-production development or test instances
- ✓ Use **automatic scaling** to match needs based on usage
 - Automated and time-based elasticity

Pillar 3: Optimal pricing model

Pillars:

1. Right-Size
2. Increase Elasticity
3. Optimal pricing model
4. Optimize storage choices



✓ Leverage the right pricing model for your use case

- Consider your usage patterns

✓ Optimize and *combine* purchase types

✓ Examples:

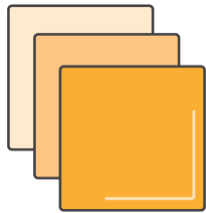
- Use **On-Demand Instance** and **Spot Instances** for variable workloads
- Use **Reserved Instances** for predictable workloads

✓ Consider serverless solutions (AWS Lambda)

Pillar 4: Optimize storage choices

Pillars:

1. Right-Size
2. Increase Elasticity
3. Optimal pricing model
4. Optimize storage choices



- ✓ Reduce costs while maintaining storage performance and availability
- ✓ Resize EBS volumes
- ✓ Change EBS volume types
 - ✓ Can you meet performance requirements with less expensive storage?
 - ✓ Example: **Amazon EBS Throughput Optimized HDD (st1)** storage typically costs half as much as the default **General Purpose SSD (gp2)** storage option.
- ✓ Delete EBS snapshots that are no longer needed
- ✓ Identify the most appropriate destination for specific types of data
 - ✓ Does the application need the instance to reside on Amazon EBS?
 - ✓ Amazon S3 storage options with lifecycle policies can reduce costs

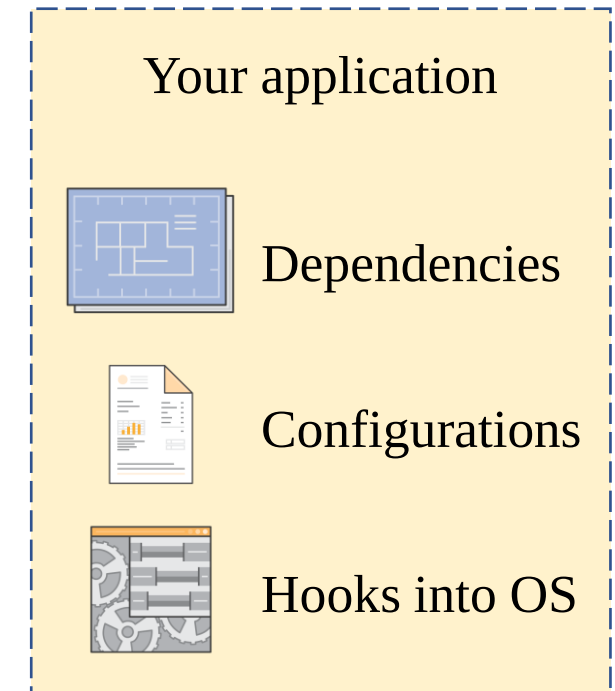
Container services



Container basics

- **Containers** are a method of **operating system virtualization**.
- Benefits –
 - Repeatable.
 - Self-contained environments.
 - Software runs the same in different environments.
 - Developer's laptop, test, production.
 - Faster to launch and stop or terminate than virtual machines

Your Container

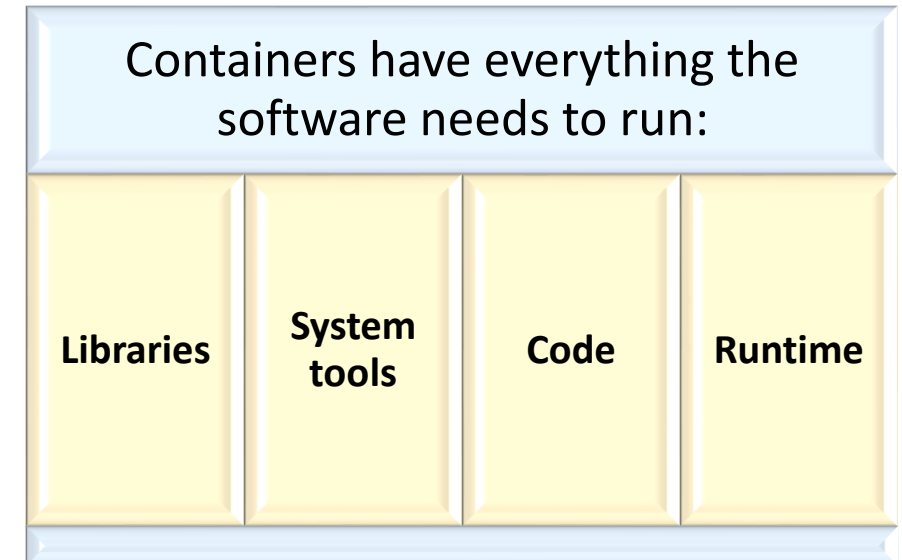


What is Docker?

- **Docker** is a software platform that enables you to build, test, and deploy applications quickly.
- You run containers on Docker.
 - Containers are created from a template called an *image*.
- A **container** has everything a software application needs to run.



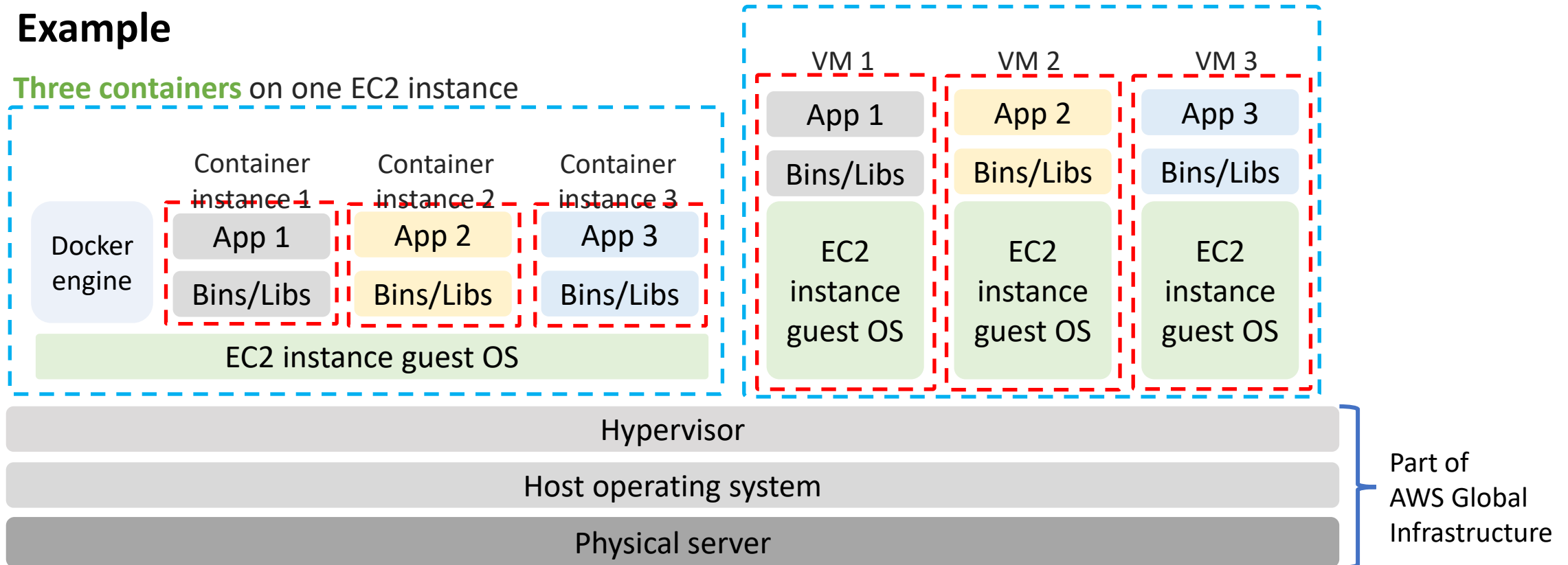
Container



Containers versus virtual machines

Example

Three containers on one EC2 instance



Amazon Elastic Container Service (Amazon ECS)

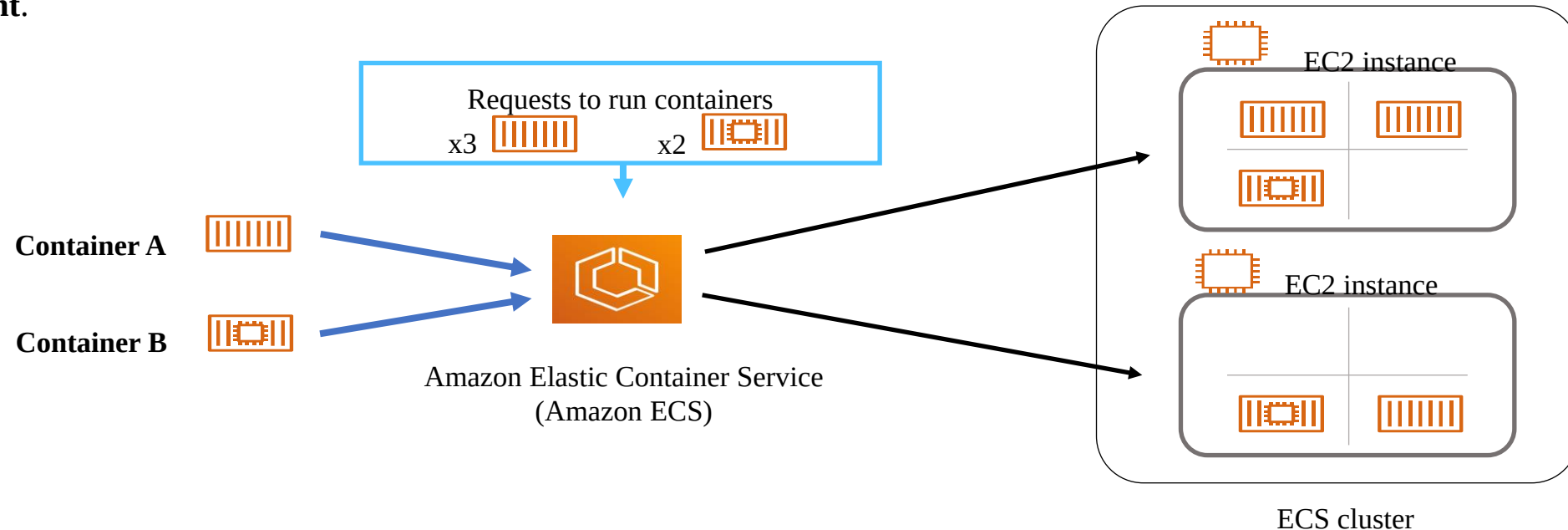
- Amazon Elastic Container Service (**Amazon ECS**) –
 - A highly scalable, fast, **container management service**
- Key benefits –
 - Orchestrates the running of Docker containers
 - Maintains and scales the fleet of nodes that run your containers
 - Removes the complexity of standing up the infrastructure
- Integrated with features that are familiar to Amazon EC2 service users –
 - Elastic Load Balancing
 - Amazon EC2 security groups
 - Amazon EBS volumes
 - IAM roles



**Amazon Elastic
Container Service**

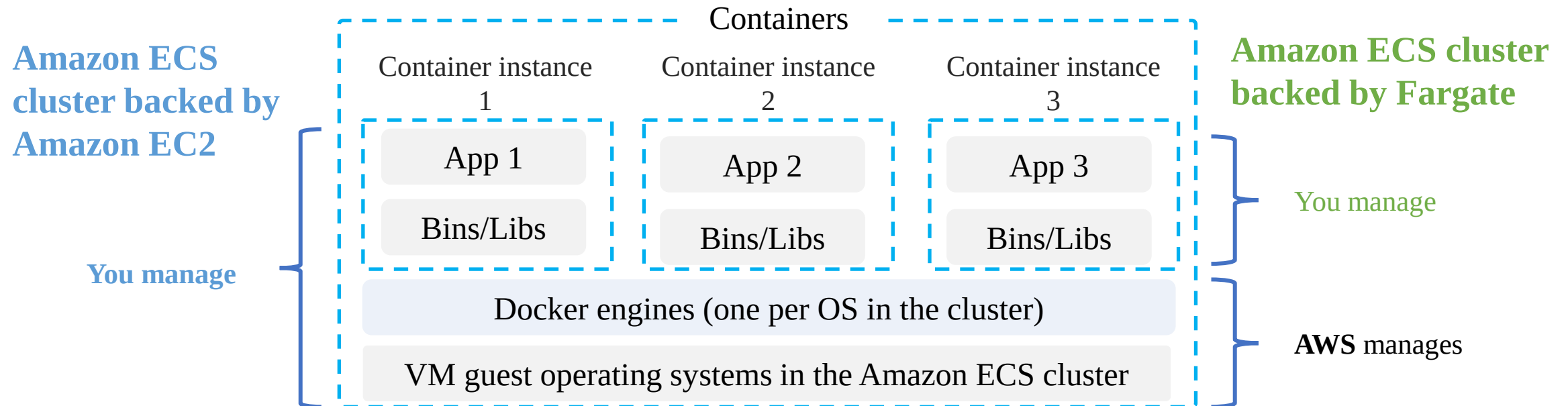
Amazon Elastic Container Service (Amazon ECS)

To prepare your application to run on **Amazon ECS**, you create a **task definition** which is a text file that describes **one or more containers**, up to a maximum of ten, that form your application. The **Amazon ECS task scheduler** is responsible for placing tasks within your cluster. A task will run anywhere from one to ten containers, depending on the task definition you defined. When Amazon ECS runs the containers that make up your task, it places them on an **ECS cluster**. 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 infrastructure)
 - If **no**, create an **Amazon ECS cluster backed by AWS Fargate** (easier to maintain, focus on your 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 (**Amazon EKS**)
 - 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
Kubernetes Service**

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.



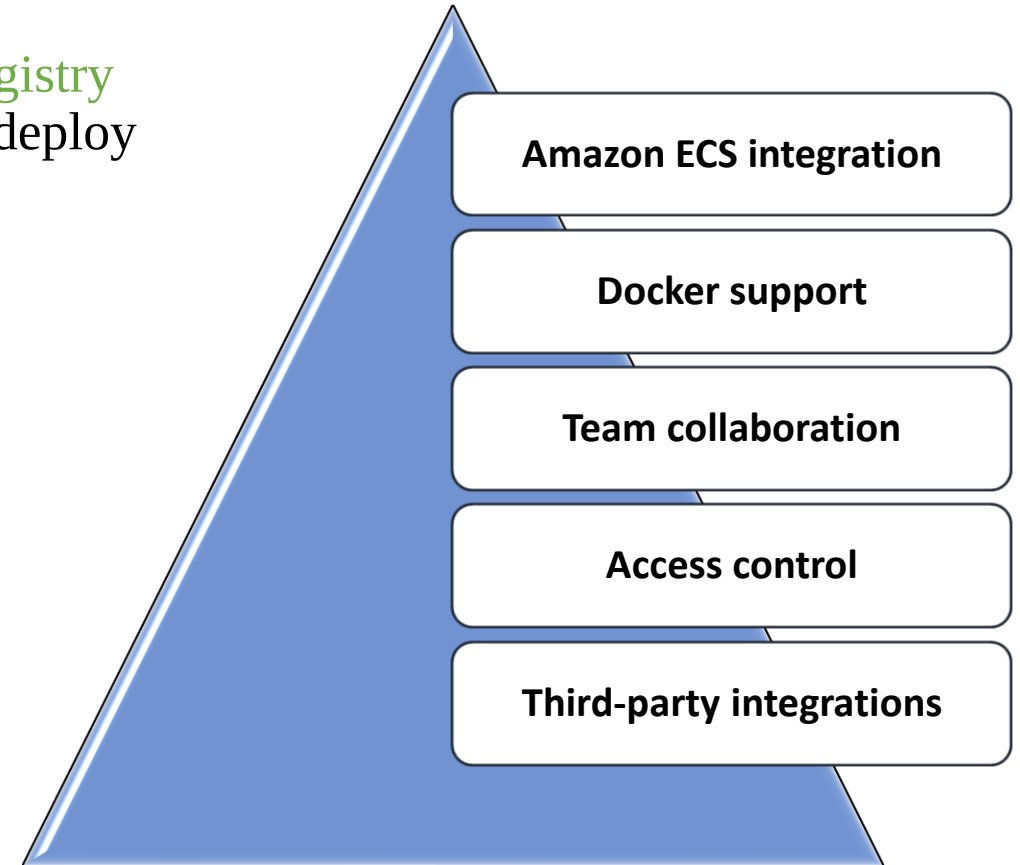
Amazon Elastic
Container Registry



Image



Registry

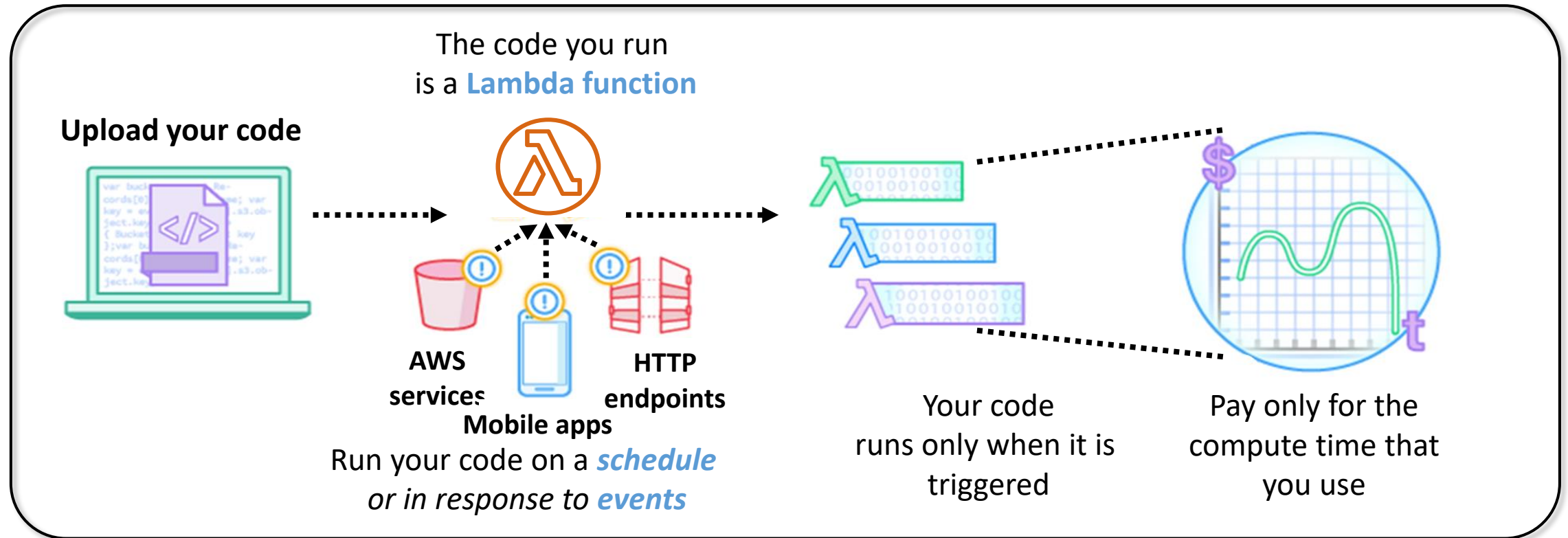


Introduction to AWS Lambda



AWS Lambda: Run code without servers

AWS Lambda is a **serverless** compute service.



Benefits of Lambda



**AWS
Lambda**



It supports multiple programming languages



Completely automated administration



Built-in fault tolerance



It supports the orchestration of multiple functions



Pay-per-use pricing

AWS Lambda event sources

Event sources

-  Amazon S3
-  Amazon DynamoDB
-  Amazon Simple Notification Service (Amazon SNS)
-  Amazon Simple Queue Service (Amazon SQS)
-  Amazon API Gateway
-  Application Load Balancer

Many more...

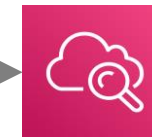
Lambda
function

AWS Lambda

Configure other AWS services as **event sources** to invoke your function as shown here.

Alternatively, invoke a Lambda function from the Lambda console, AWS SDK, or AWS CLI.

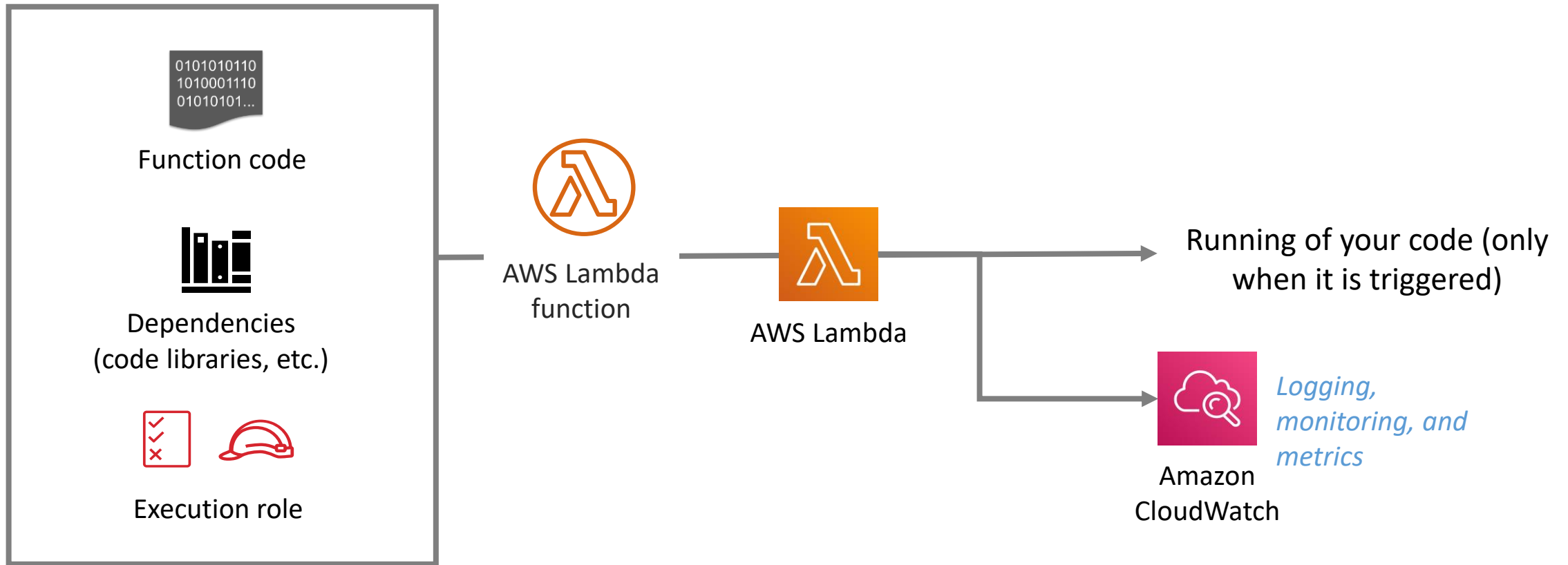
Running of your code
(only when triggered)

Amazon
CloudWatch

*Logging, monitoring,
and metrics*

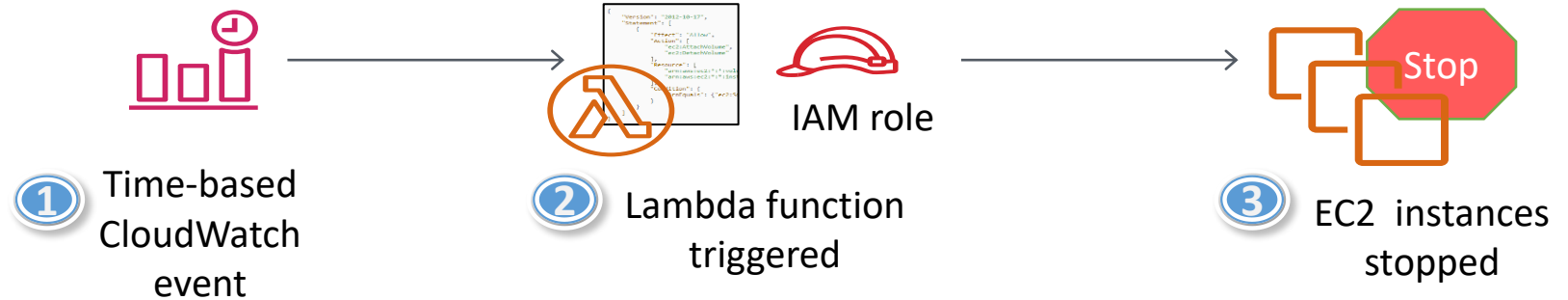
AWS Lambda function configuration

Lambda function configuration

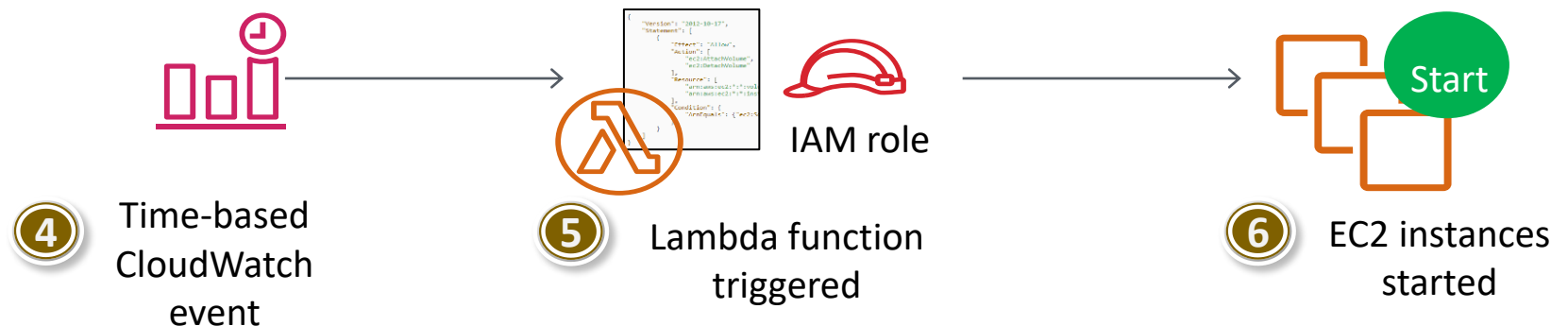


Schedule-based Lambda function example: Start and stop EC2 instances

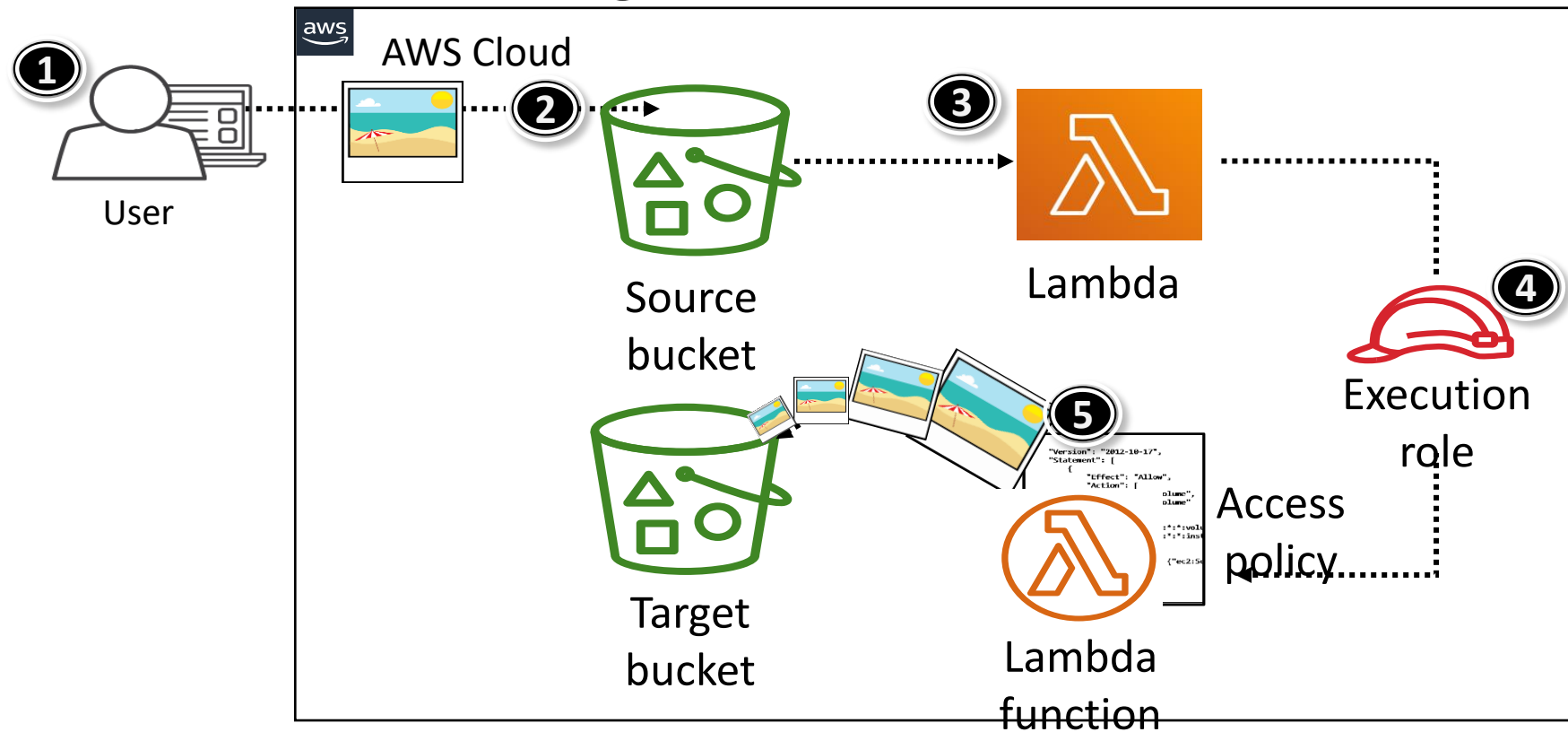
Stop instances example



Start instances example



Event-based Lambda function example: Create thumbnail images



AWS Lambda quotas

Soft limits per Region:

- Concurrent executions = 1,000
- Function and layer storage = 75 GB

Hard limits for individual functions:

- Maximum function memory allocation = 3,008 MB
- Function timeout = 15 minutes
- Deployment package size = 250 MB unzipped, including layers

Additional limits also exist. Details are in the [AWS Lambda quotas](#) documentation.

Introduction to AWS Elastic Beanstalk



AWS Elastic Beanstalk



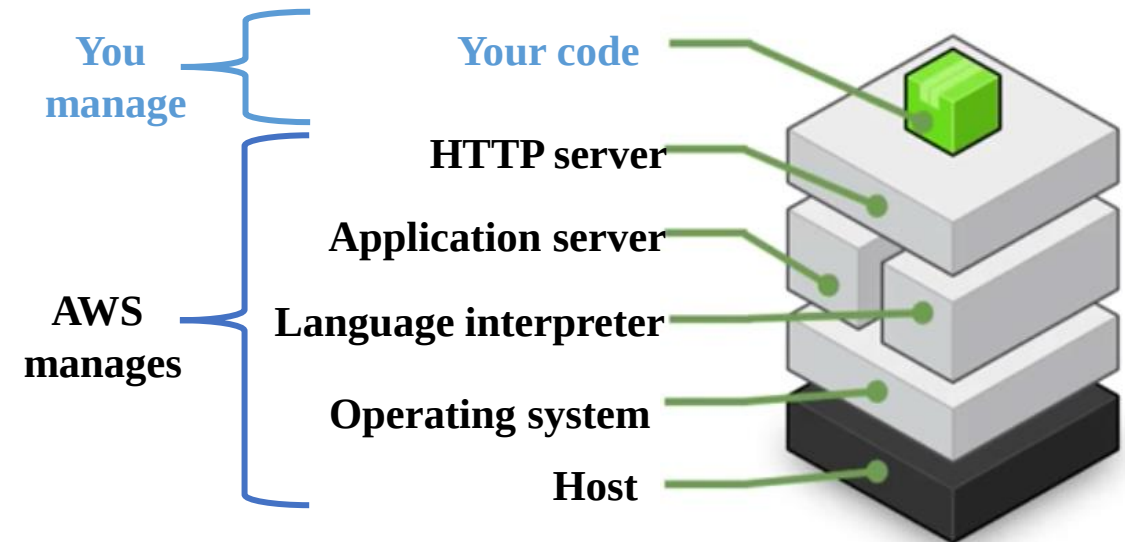
AWS Elastic Beanstalk

- An easy way to get **web applications** up and running
- A **managed service** that automatically handles –
 - Infrastructure provisioning and configuration
 - Deployment
 - Load balancing
 - Automatic scaling
 - Health monitoring
 - Analysis and debugging
 - Logging
- No additional charge for Elastic Beanstalk
 - Pay only for the underlying resources that are used



AWS Elastic Beanstalk deployments

- It supports web applications written for common platforms
 - **Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker**
- You upload your code
 - Elastic Beanstalk automatically handles the deployment
 - Deploys on servers such as Apache, NGINX, Passenger, Puma, and Microsoft Internet Information Services (IIS)



Benefits of Elastic Beanstalk



Fast and simple to start using

Use the AWS Management Console, a Git repository, or an integrated development environment (IDE) such as Eclipse or Visual Studio to upload your application



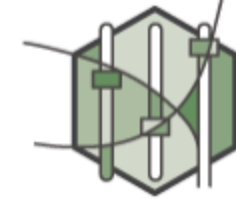
Developer productivity

by focusing on writing code instead of managing and configuring servers, databases, load balancers, firewalls, and networks



Difficult to outgrow

With Elastic Beanstalk, your application can handle peaks in workload or traffic while minimizing your costs



Complete resource control

—such as Amazon EC2 instance type—that are optimal for your application.

Thanks