Compute

Module overview

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
- Hands-on with AWS Lambda
- Hands-on with AWS Elastic Beanstalk

Demo

- Recorded demonstration of Amazon EC2

Lab

- Introduction to Amazon EC2

✓ **Knowledge check**

Module objectives

After completing this module, you should be able to:

- Provide an overview of different AWS compute services in the cloud
- Demonstrate why to use Amazon Elastic Compute Cloud (Amazon EC2)
- Identify the functionality in the EC2 console
- Perform basic functions in Amazon EC2 to build a virtual computing environment
- Identify Amazon EC2 cost optimization elements
- Demonstrate when to use AWS Elastic Beanstalk
- Demonstrate when to use AWS Lambda
- Identify how to run containerized applications in a cluster of managed servers

Compute

Section 1: Compute services overview

AWS compute services

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

- Amazon EC2
- Amazon EC2 Auto Scaling
- Amazon Elastic Container Registry (Amazon ECR)
- Amazon Elastic Container Service (Amazon ECS)
- VMware Cloud on AWS
- AWS Elastic Beanstalk
- AWS Lambda
- Amazon Elastic Kubernetes Service (Amazon EKS)
- Amazon Lightsail
- AWS Batch
- AWS Fargate
- AWS Outposts
- AWS Serverless Application Repository

Categorizing compute services

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

Compute

Section 2: Amazon EC2

Amazon Elastic Compute Cloud (Amazon EC2)

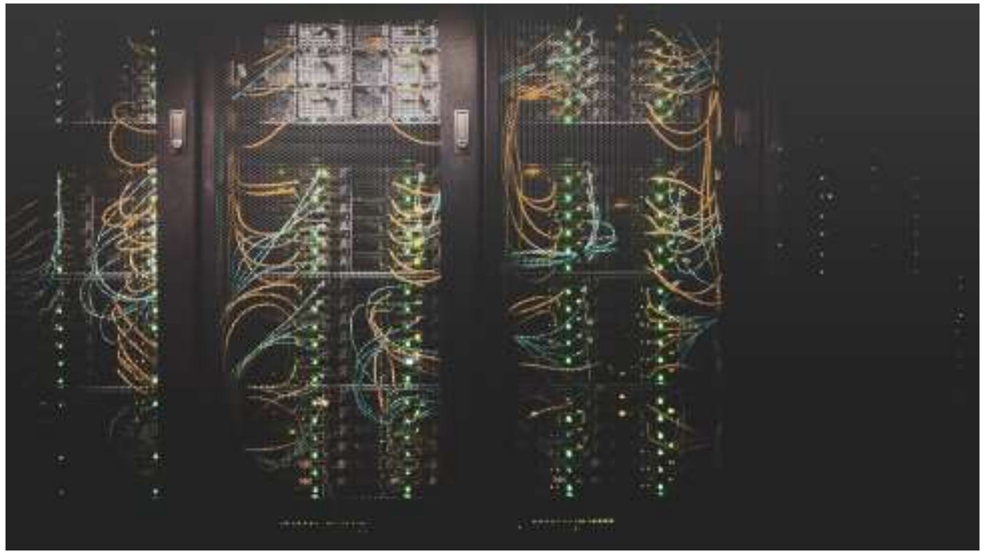
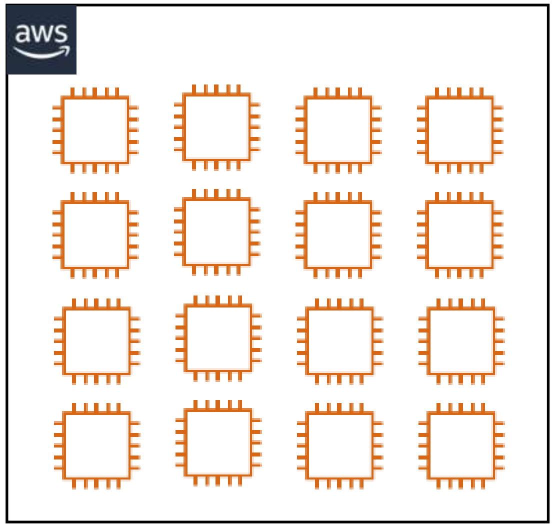


Photo by Taylor Vick on Unsplash

On-premises servers

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**



Amazon EC2 instances

Photo by panumas nikhomkhai from Pexels

Amazon EC2 overview

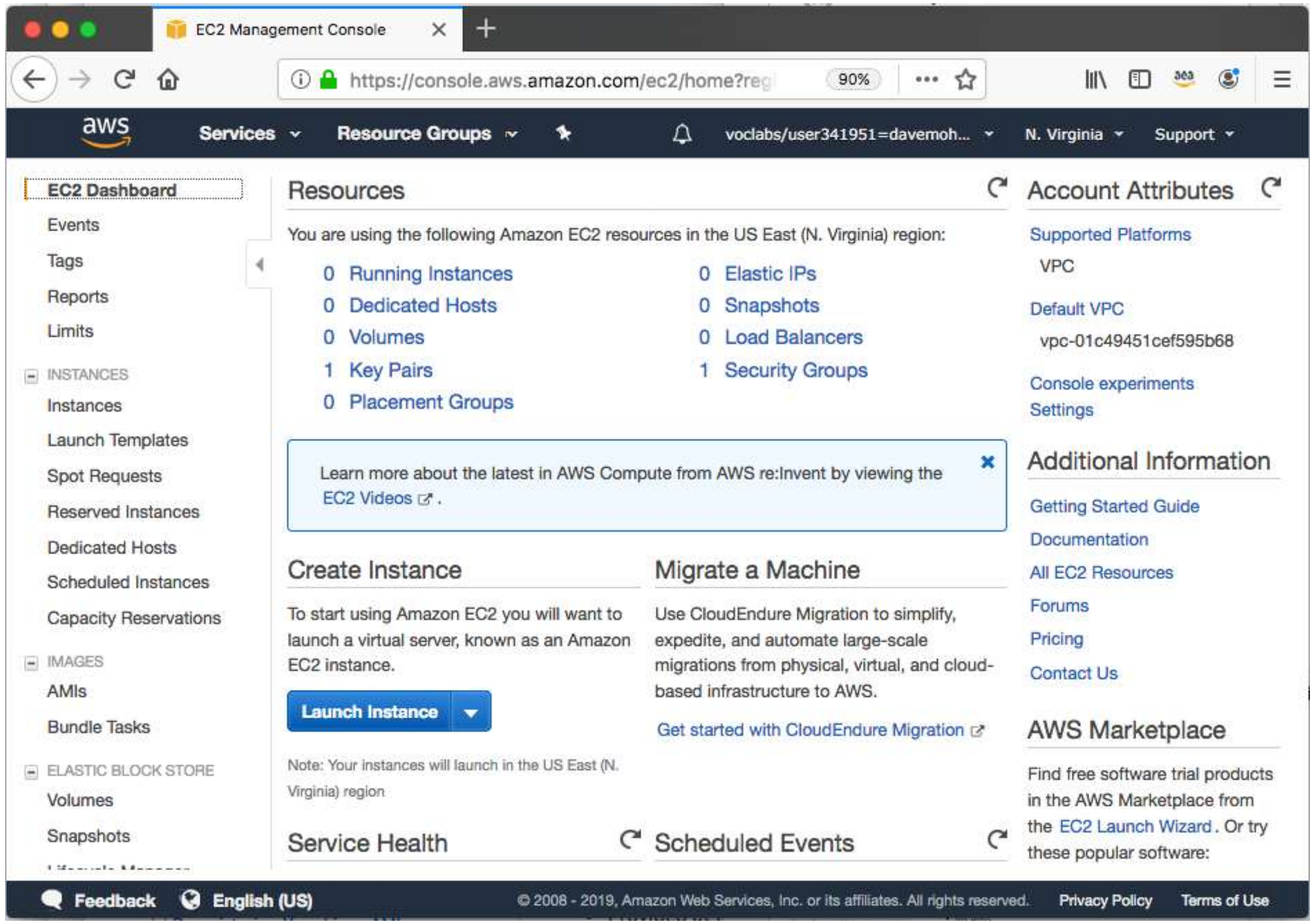
Amazon
EC2

- **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.

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.



Resources

You are using the following Amazon EC2 resources in the US East (N. Virginia) region:

0 Running Instances	0 Elastic IPs
0 Dedicated Hosts	0 Snapshots
0 Volumes	0 Load Balancers
1 Key Pairs	1 Security Groups
0 Placement Groups	

Learn more about the latest in AWS Compute from AWS re:Invent by viewing the EC2 Videos .

Create Instance

To start using Amazon EC2 you will want to launch a virtual server, known as an Amazon EC2 instance.

Launch Instance

Note: Your instances will launch in the US East (N. Virginia) region

Migrate a Machine

Use CloudEndure Migration to simplify, expedite, and automate large-scale migrations from physical, virtual, and cloud-based infrastructure to AWS.

Get started with CloudEndure Migration

Service Health

Scheduled Events

Account Attributes

Supported Platforms
VPC

Default VPC
vpc-01c49451cef595b68

Console experiments
Settings

Additional Information

Getting Started Guide
Documentation
All EC2 Resources
Forums
Pricing
Contact Us

AWS Marketplace

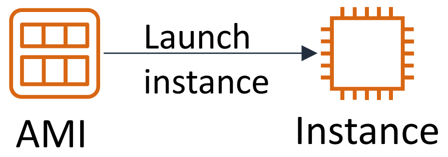
Find free software trial products in the AWS Marketplace from the EC2 Launch Wizard. Or try these popular software:

Feedback English (US) © 2008 - 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

1. Select an AMI

Choices made using the Launch Instance Wizard:

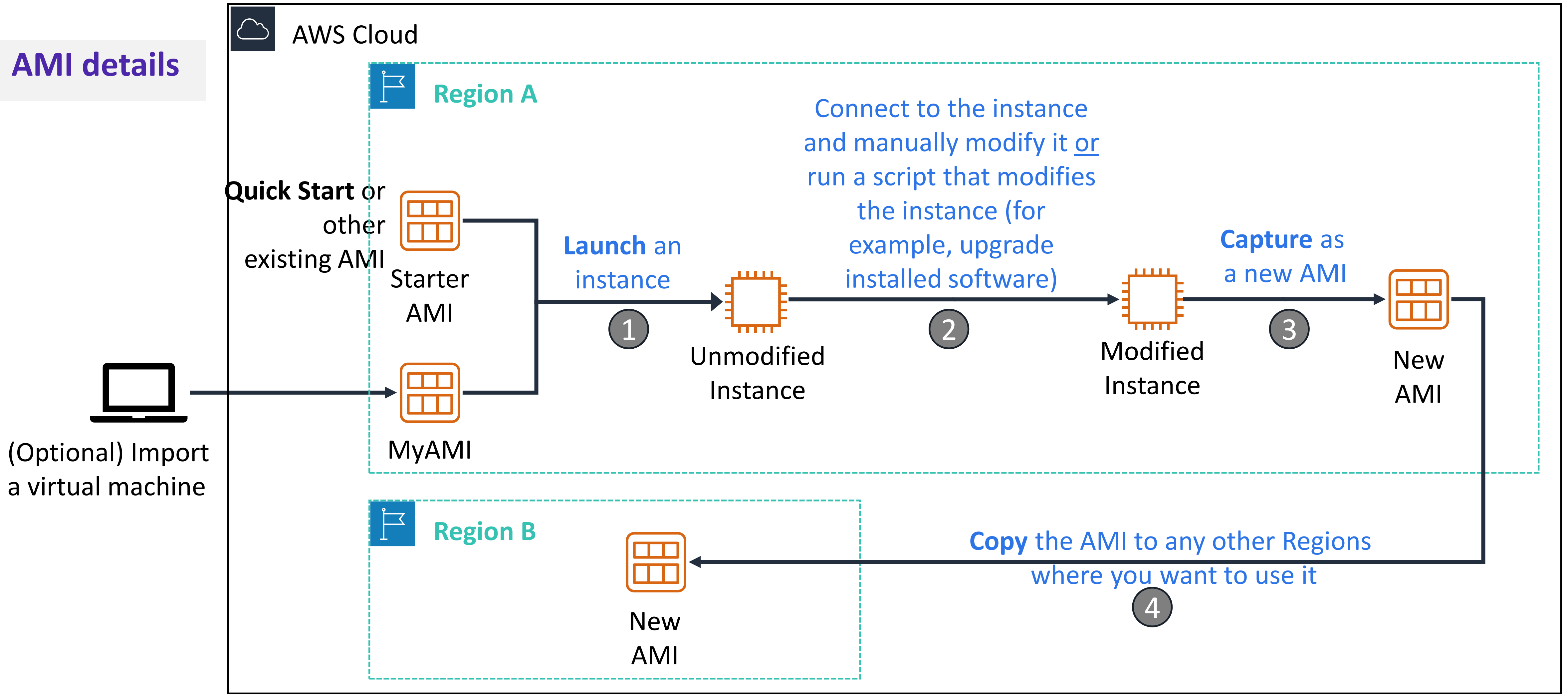
1. AMI
2. **Instance Type**
3. **Network settings**
4. **IAM role**
5. **User data**
6. **Storage options**
7. **Tags**
8. **Security group**
9. **Key pair**



- 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



AWS Cloud
Region A
Quick Start or other existing AMI
Starter AMI
MyAMI
(Optional) Import a virtual machine
Launch an instance
1
Unmodified Instance
Connect to the instance and manually modify it or run a script that modifies the instance (for example, upgrade installed software)
2
Modified Instance
Capture as a new AMI
3
New AMI
Region B
New AMI
Copy the AMI to any other Regions where you want to use it
4

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

Instance type details

Instance type naming

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

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

Select instance type: Based on use case

Instance type details

	General Purpose	Compute Optimized	Memory Optimized	Accelerated Computing	Storage 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.

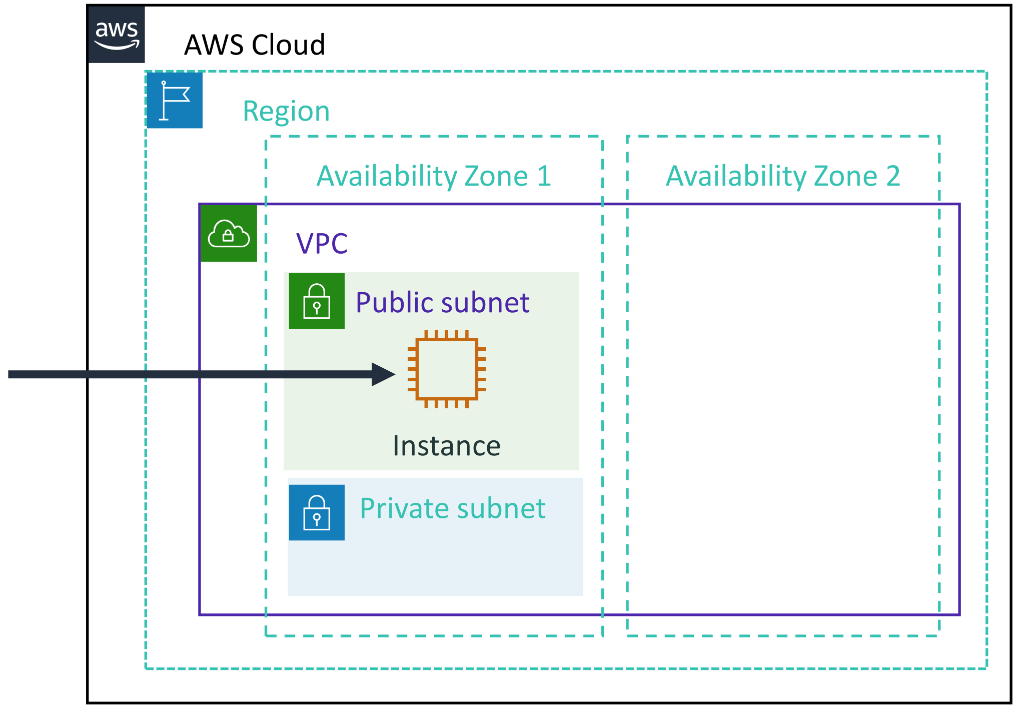
3. Specify network settings

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



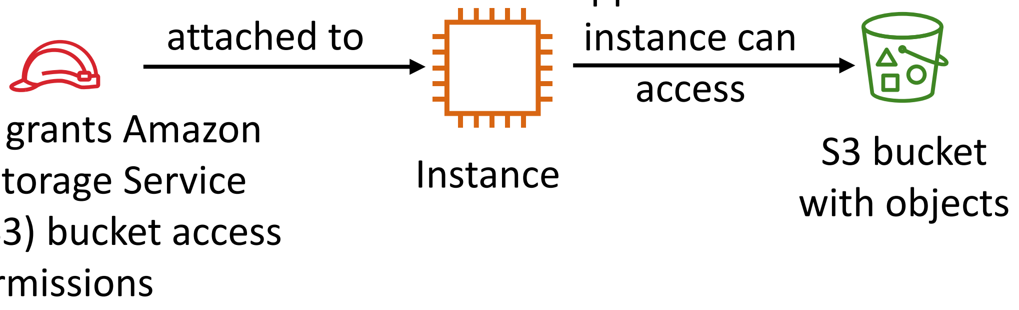
4. Attach IAM role (optional)

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.

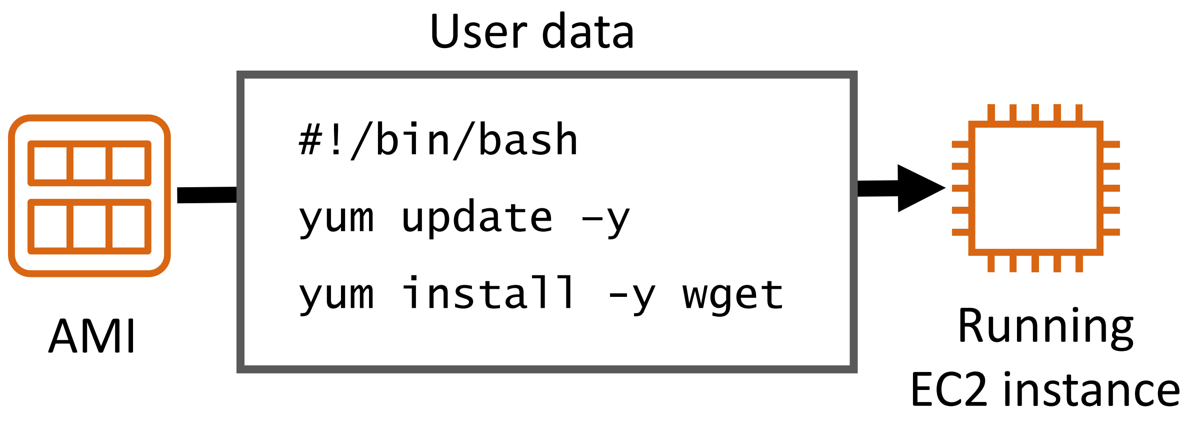
Example:


attached to
Application on instance can access
Role that grants Amazon Simple Storage Service (Amazon S3) bucket access permissions
Instance
S3 bucket with objects

5. User data script (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**



```
#!/bin/bash
yum update -y
yum install -y wget
```

- 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**

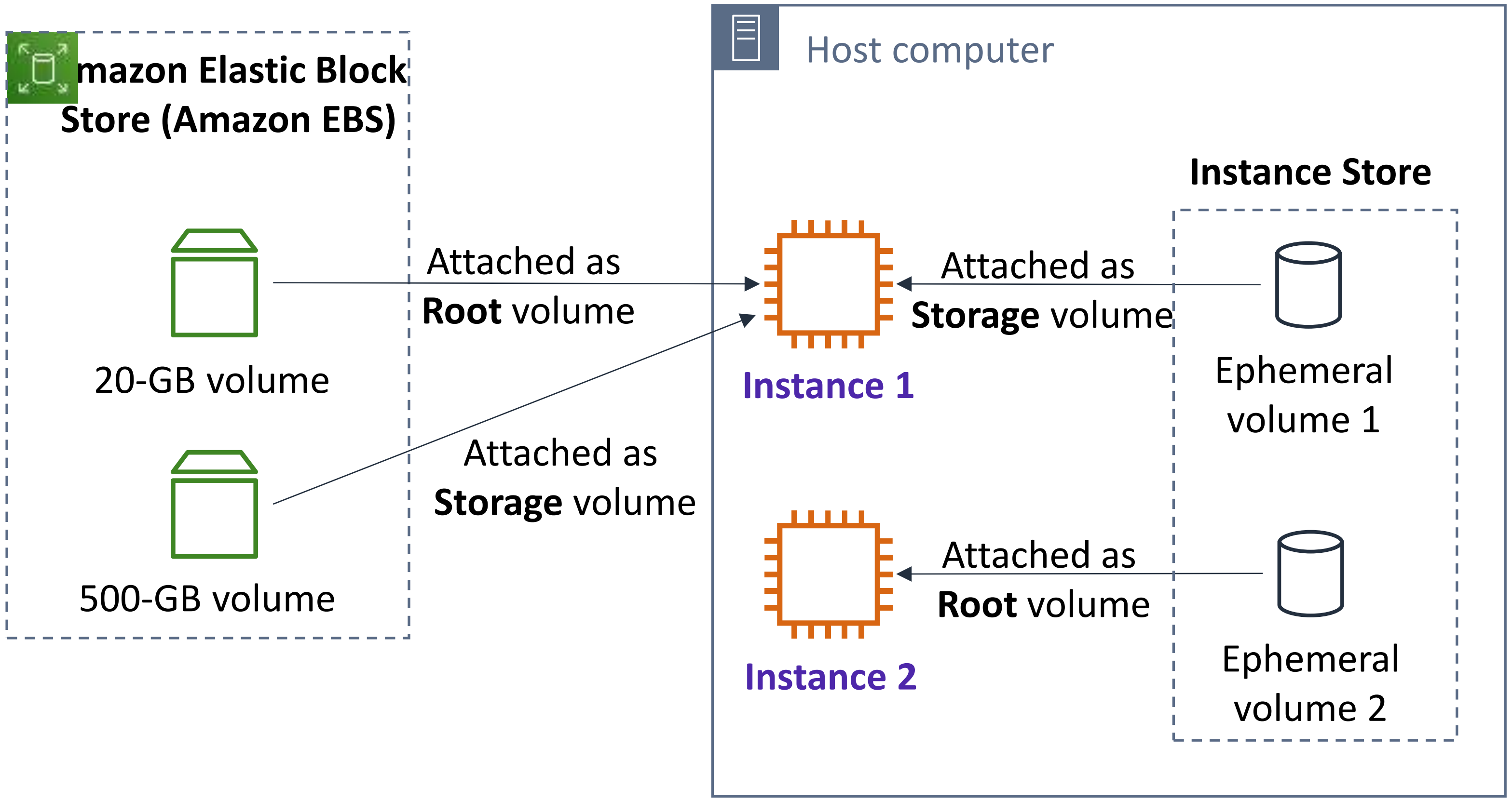
- Configure the root volume
  - Where the guest operating system is installed
- Attach additional storage volumes (optional)
  - AMI might already include more than one volume
- For each volume, specify:
  - The **size** of the disk (in GB)
  - The **volume type**
    - Different types of solid state 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)?



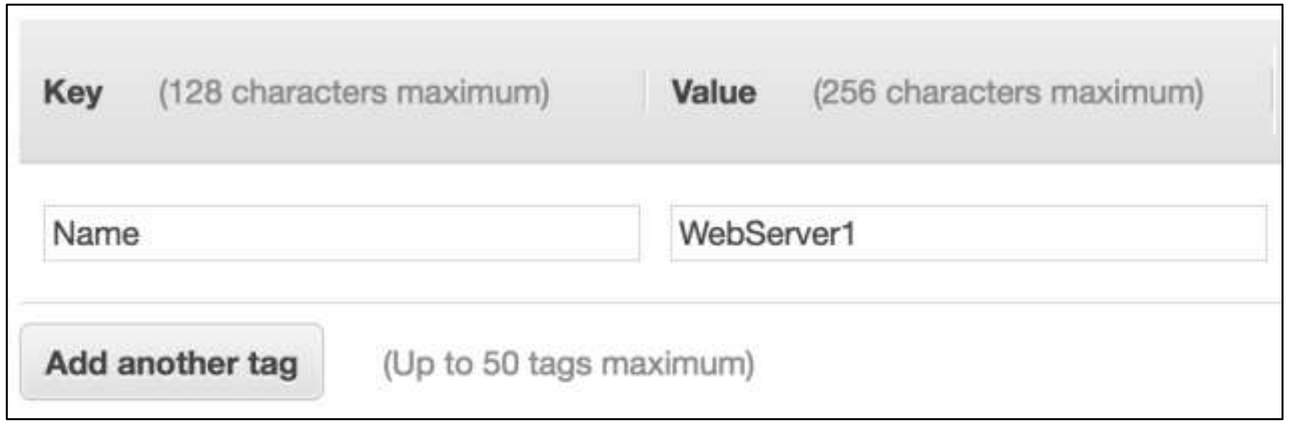
7. Add tags

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:



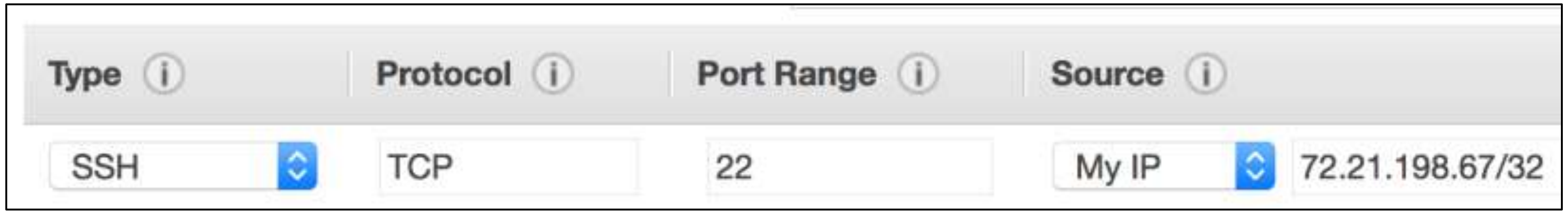
Key (128 characters maximum)	Value (256 characters maximum)
Name	WebServer1

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.



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

9. Identify or create the key pair

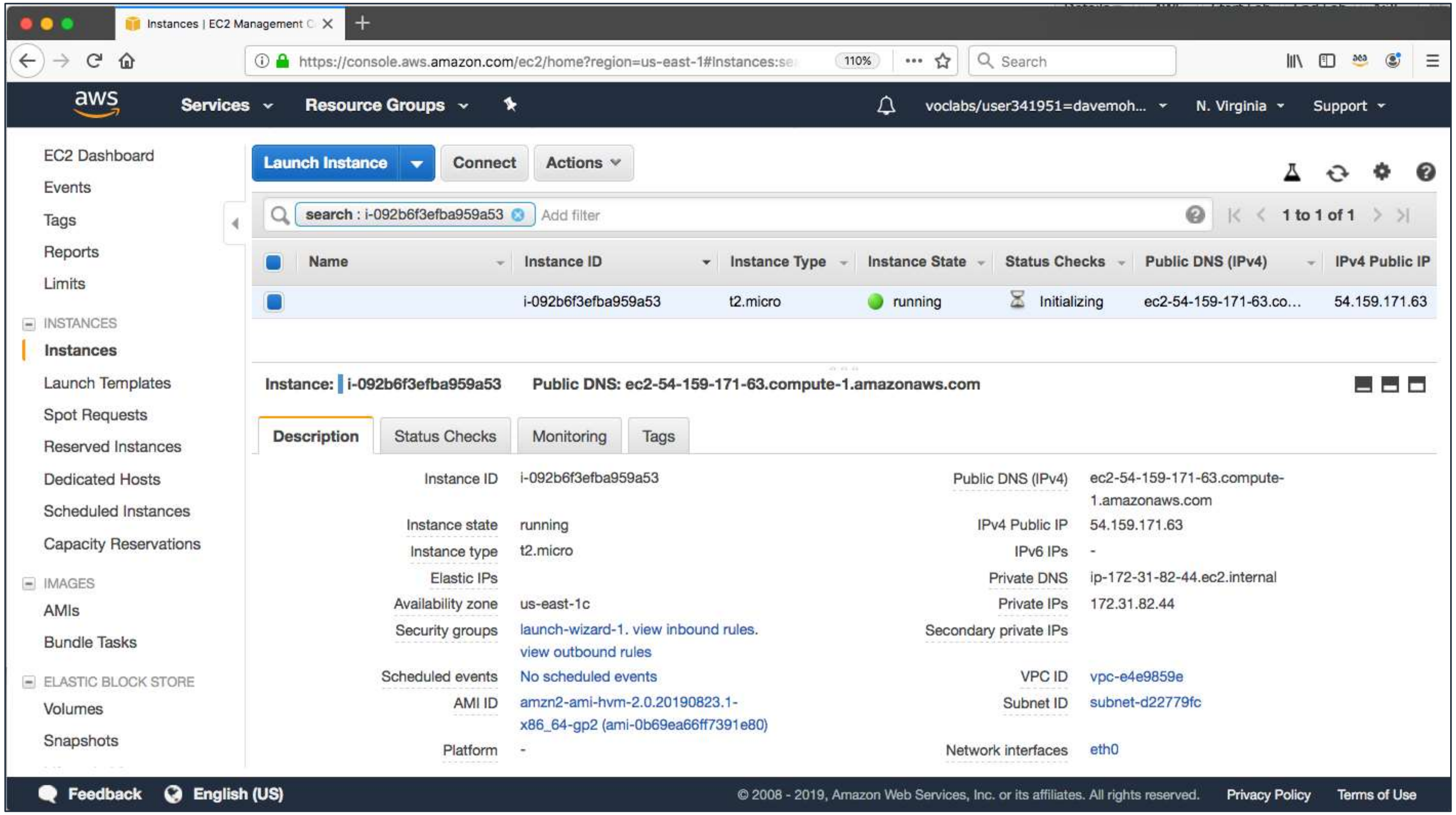
Choices made by using the Launch Instance Wizard:

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.

mykey.pem

Amazon EC2 console view of a running EC2 instance



Name	Instance ID	Instance Type	Instance State	Status Checks	Public DNS (IPv4)	IPv4 Public IP
	i-092b6f3efba959a53	t2.micro	● running	Initializing	ec2-54-159-171-63.co...	54.159.171.63

Instance: i-092b6f3efba959a53   Public DNS: ec2-54-159-171-63.compute-1.amazonaws.com

Description | Status Checks | Monitoring | Tags

Instance ID	i-092b6f3efba959a53	Public DNS (IPv4)	ec2-54-159-171-63.compute-1.amazonaws.com
Instance state	running	IPv4 Public IP	54.159.171.63
Instance type	t2.micro	IPv6 IPs	-
Elastic IPs		Private DNS	ip-172-31-82-44.ec2.internal
Availability zone	us-east-1c	Private IPs	172.31.82.44
Security groups	launch-wizard-1. view inbound rules. view outbound rules	Secondary private IPs	
Scheduled events	No scheduled events	VPC ID	vpc-e4e9859e
AMI ID	amzn2-ami-hvm-2.0.20190823.1-x86_64-gp2 (ami-0b69ea66ff7391e80)	Subnet ID	subnet-d22779fc
Platform	-	Network interfaces	eth0

© 2008 - 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Another option: Launch an EC2 instance with the AWS Command Line Interface

- EC2 instances can also be created programmatically.

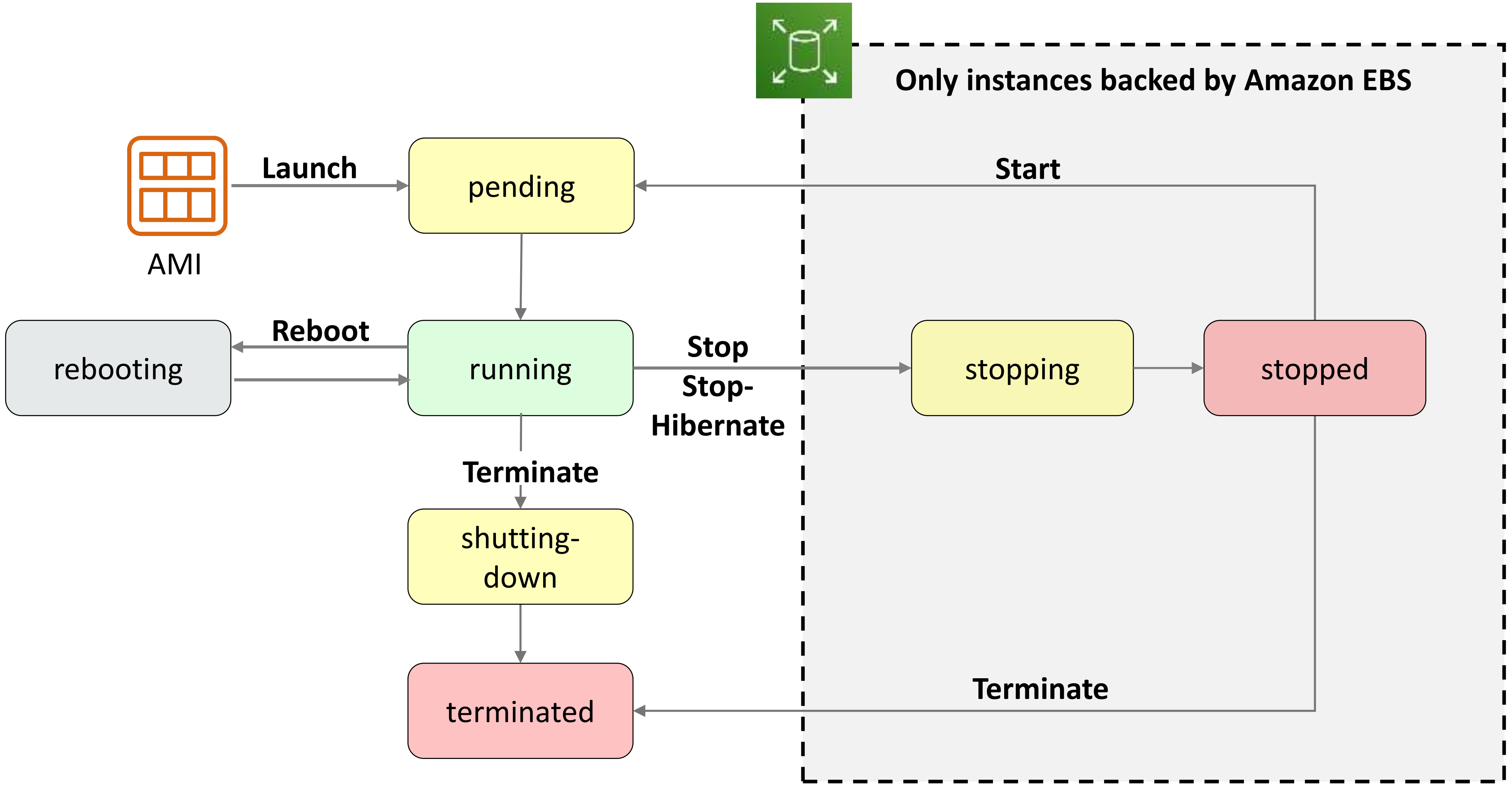
AWS Command Line Interface (AWS CLI)

- This example shows how simple the command can be.
  - This command assumes that the key pair and security group already exist.
  - More options could be specified. See the [AWS CLI Command Reference]() for details.

Example command:

```
aws ec2 run-instances \
--image-id ami-1a2b3c4d \
--count 1 \
--instance-type c3.large \
--key-name MyKeyPair \
--security-groups MySecurityGroup \
--region us-east-1
```

Amazon EC2 instance lifecycle



Consider using an Elastic IP address

- **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.

Elastic IP Address

EC2 instance metadata

- **Instance metadata** is data about your instance.
- While you are connected to the instance, you can view it –
  - In a browser: `http://169.254.169.254/latest/meta-data/`
  - In a terminal window: `curl http://169.254.169.254/latest/meta-data/`
- Example retrievable values –
  - Public IP address, private IP address, public hostname, instance ID, security groups, Region, Availability Zone.
  - Any user data specified at instance launch can also be accessed at: `http://169.254.169.254/latest/user-data/`
- It can be used to configure or manage a running instance.
  - For example, author a configuration script that reads the metadata and uses it to configure applications or OS settings.

Amazon CloudWatch for monitoring

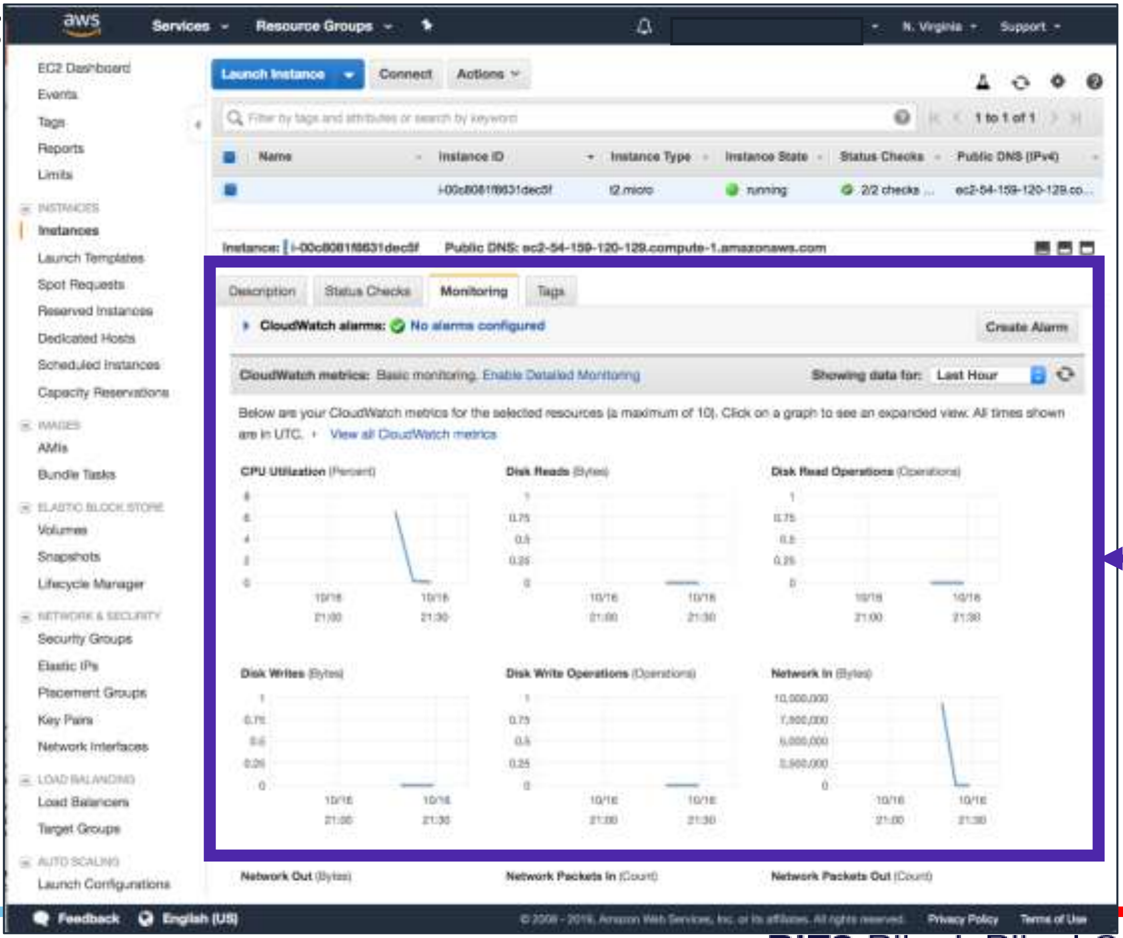
- Use **Amazon CloudWatch** to monitor EC2 instances
  - Provides near-real-time metrics
  - Provides charts in the Amazon EC2 console **Monitoring** tab that you can view
  - Maintains 15 months of historical data

- **Basic monitoring**
  - Default, no additional cost
  - Metric data sent to CloudWatch every 5 minutes

- **Detailed monitoring**
  - Fixed monthly rate for seven pre-selected metrics
  - Metric data delivered every 1 minute

Amazon CloudWatch

Instance with CloudWatch



Name	Instance ID	Instance Type	Instance State	Status Checks	Public DNS (IPv4)
	i-00c8081f8631dec5f	t2.micro	running	2/2 checks ...	ec2-54-159-120-129.co...

Instance: i-00c8081f8631dec5f Public DNS: ec2-54-159-120-129.compute-1.amazonaws.com

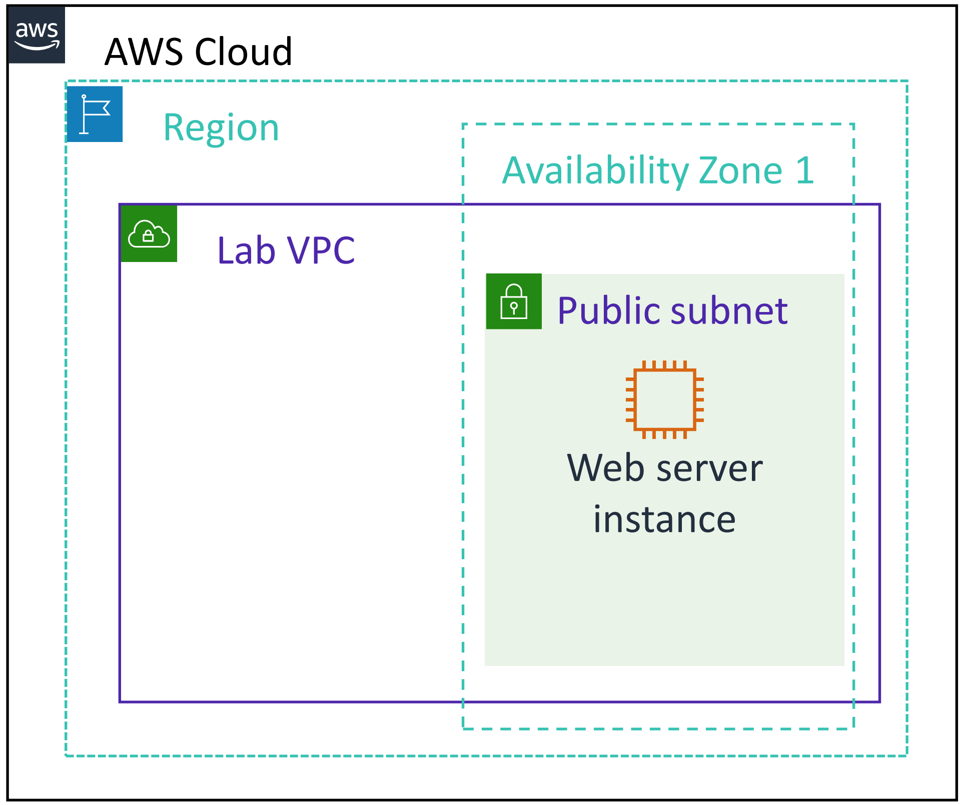
CloudWatch alarms: No alarms configured

CloudWatch metrics: Basic monitoring. Enable Detailed Monitoring Showing data for: Last Hour

Below are your CloudWatch metrics for the selected resources (a maximum of 10). Click on a graph to see an expanded view. All times shown are in UTC. View all CloudWatch metrics

Lab scenario

In this lab, you will launch and configure your first virtual machine that runs on Amazon EC2.



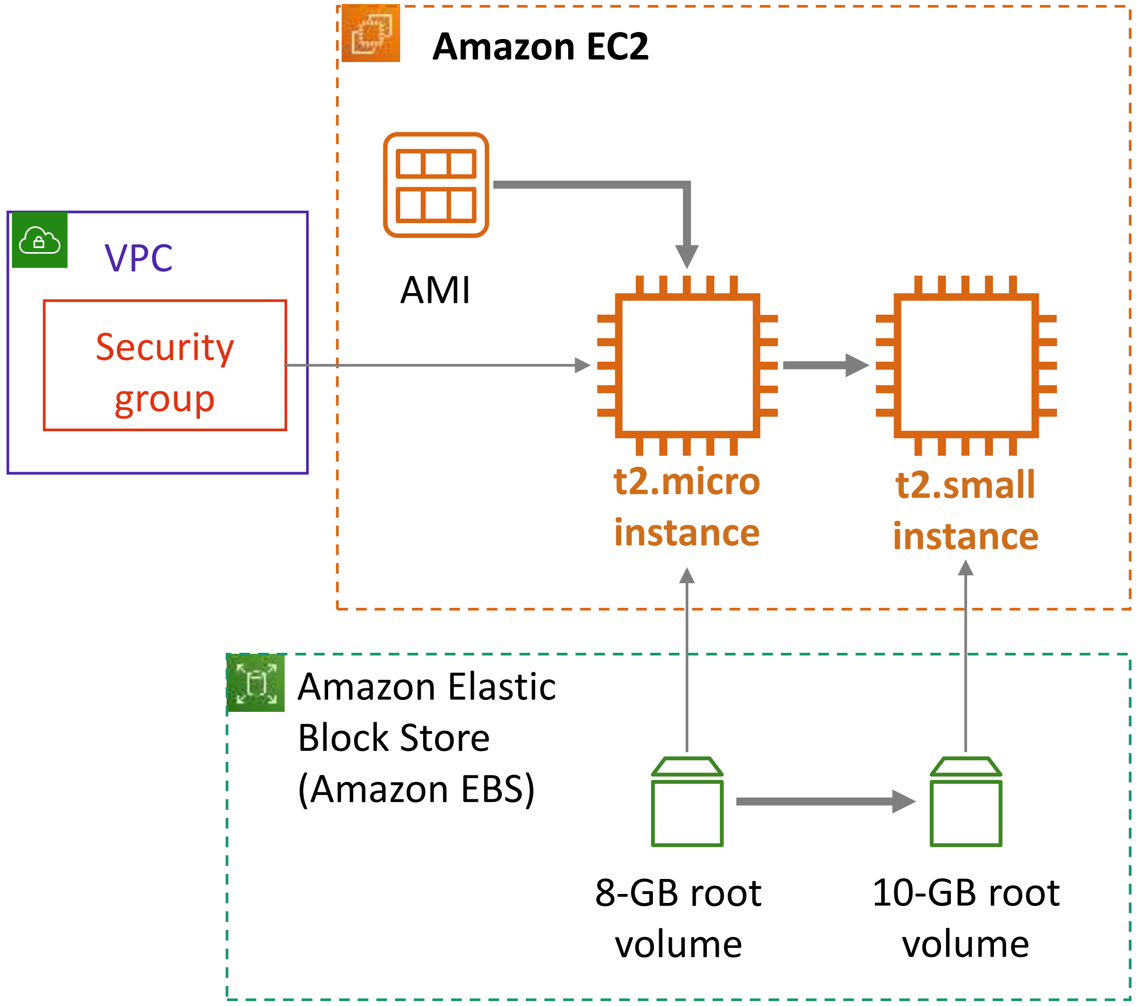
Lab: Tasks

- Task 1 – Launch Your Amazon EC2 Instance
- Task 2 – Monitor Your Instance
- Task 3 – Update Your Security Group and Access the Web Server
- Task 4 – Resize Your Instance: Instance Type and EBS Volume
- Task 5 – Explore EC2 Limits
- Task 6 – Test Termination Protection

Lab: Final product

By the end of the lab, you will have:

1. Launched an instance that is configured as a web server
2. Viewed the instance system log
3. Reconfigured a security group
4. Modified the instance type and root volume size



Activity: Check your understanding

1. Between Amazon EC2 or Amazon RDS, which provides a managed service? What does *managed service* mean?
   - **ANSWER:** Amazon RDS provides a managed service. Amazon RDS handles provisioning, installation and patching, automated backups, restoring snapshots from points in time, high availability, and monitoring.
2. Name at least one advantage of deploying Microsoft SQL Server on Amazon EC2 instead of Amazon RDS.
   - **ANSWER:** Amazon EC2 offers complete control over every configuration, the OS, and the software stack.
3. What advantage does the Quick Start provide over a manual installation on Amazon EC2?
   - **ANSWER:** The Quick Start is a reference architecture with proven best practices built into the design.
4. Which deployment option offers the best approach for all use cases?
   - **ANSWER:** Neither. The correct deployment option depends on your specific needs.
5. Which approach costs more: using Amazon EC2 or using Amazon RDS?
   - **ANSWER:** It depends. Managing the database deployment on Amazon EC2 requires more customer oversight and time. If time is your priority, then Amazon RDS might be less expensive. If you have in-house expertise, Amazon EC2 might be more cost-effective.

Compute

Section 3: 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

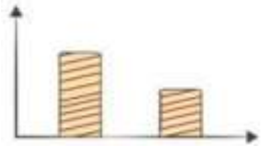
- Purchase a capacity reservation that is always available on a recurring schedule you specify.
- 1-year term.

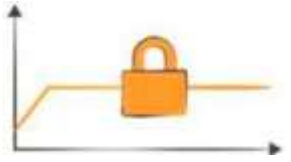
Spot Instances

- 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.

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

Amazon EC2 pricing models: Benefits

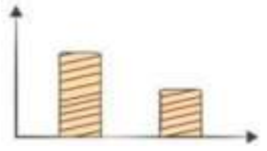




On-Demand Instances	Spot Instances	Reserved Instances	Dedicated Hosts
• Low cost and flexibility	• Large scale, dynamic workload	• Predictability ensures compute capacity is available when needed	• Save money on licensing costs • Help meet compliance and regulatory requirements

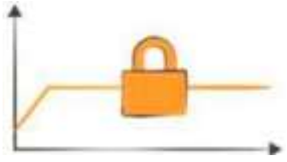
Amazon EC2 pricing models: Use cases

Spiky Workloads



Time-Insensitive Workloads

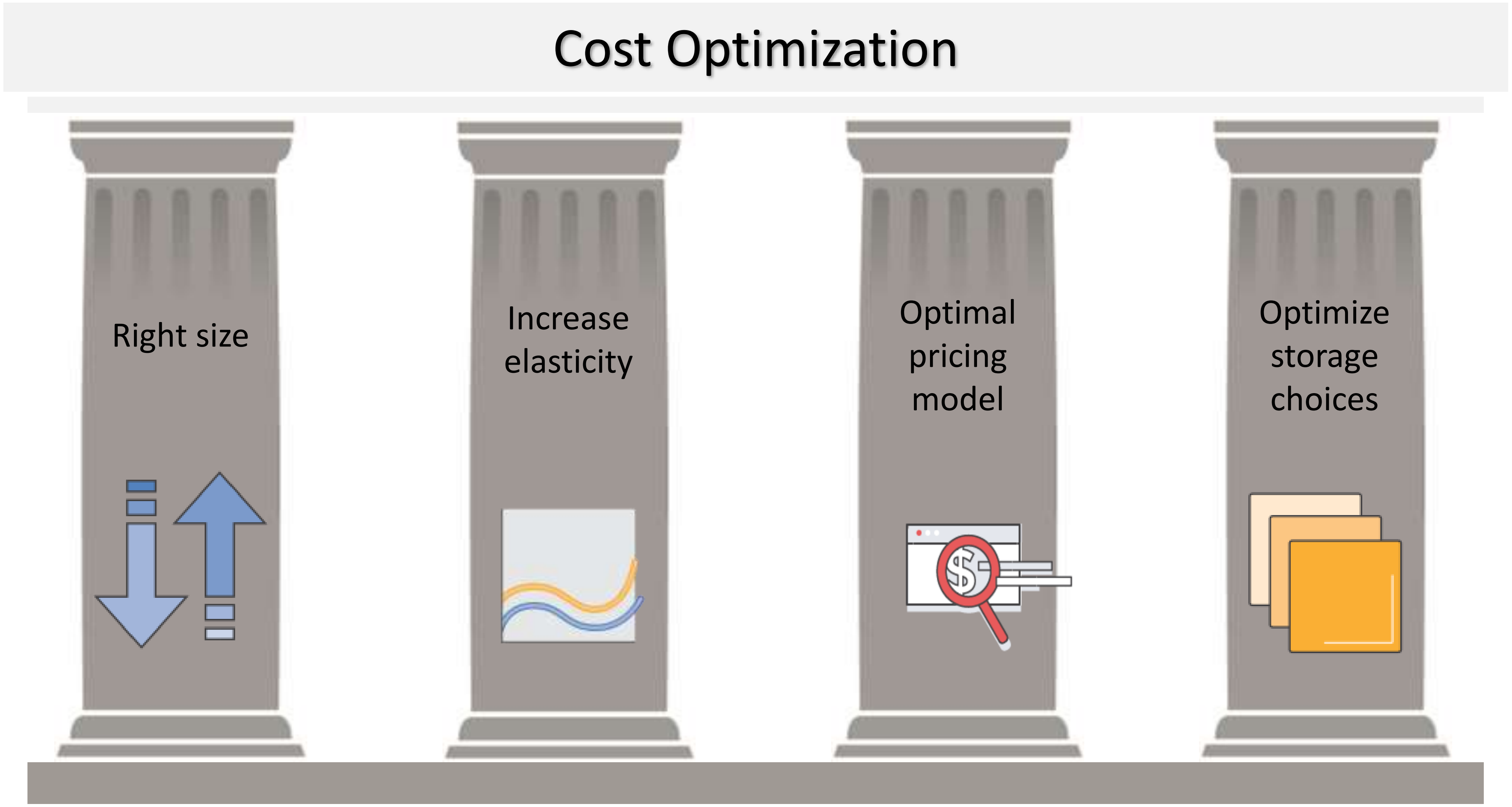
Steady-State Workloads



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

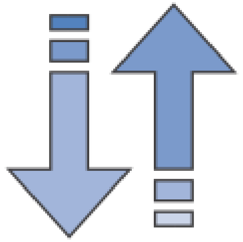
The four pillars of cost optimization



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

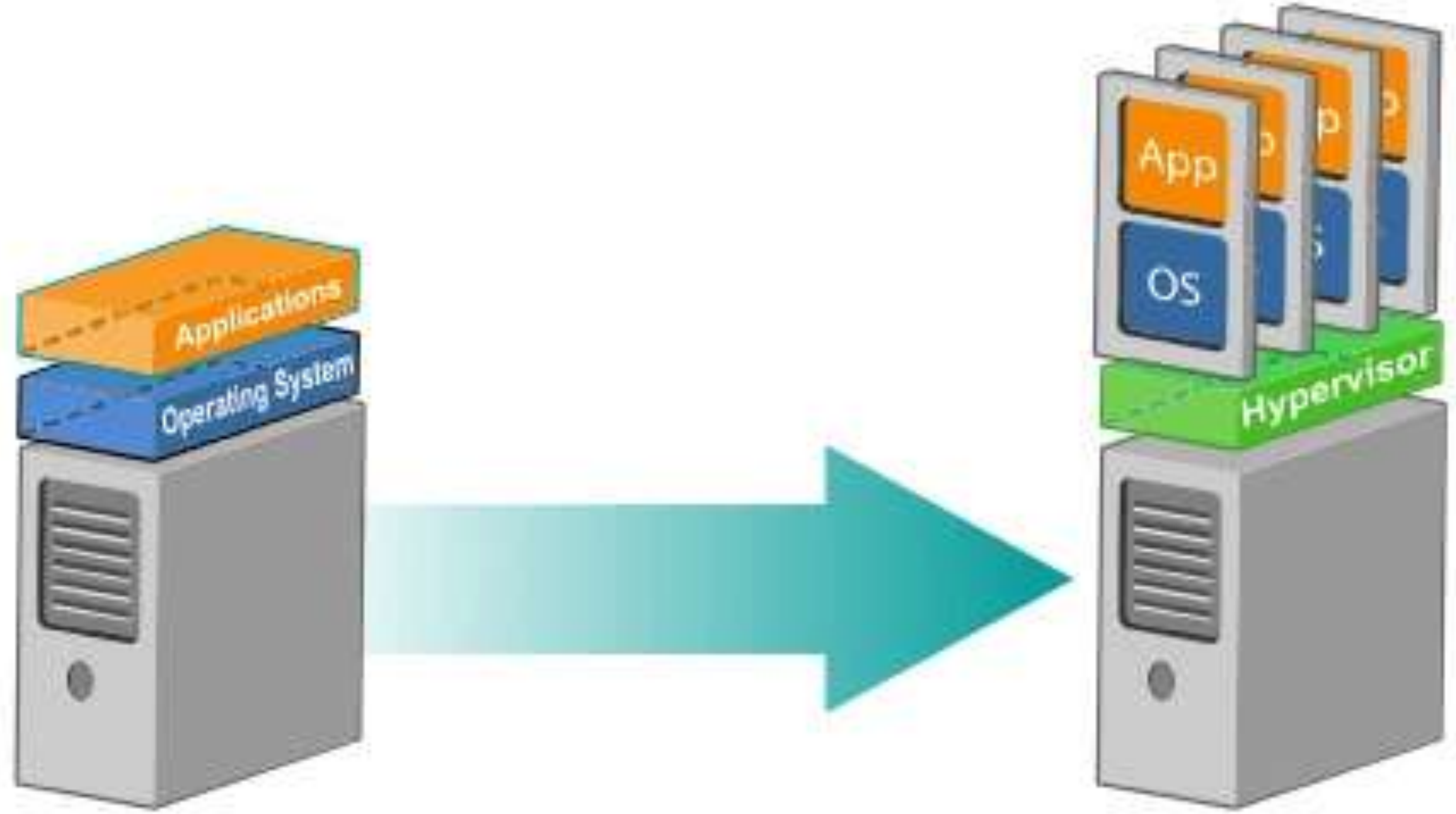
Measure, monitor, and improve

- Cost optimization is an 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.

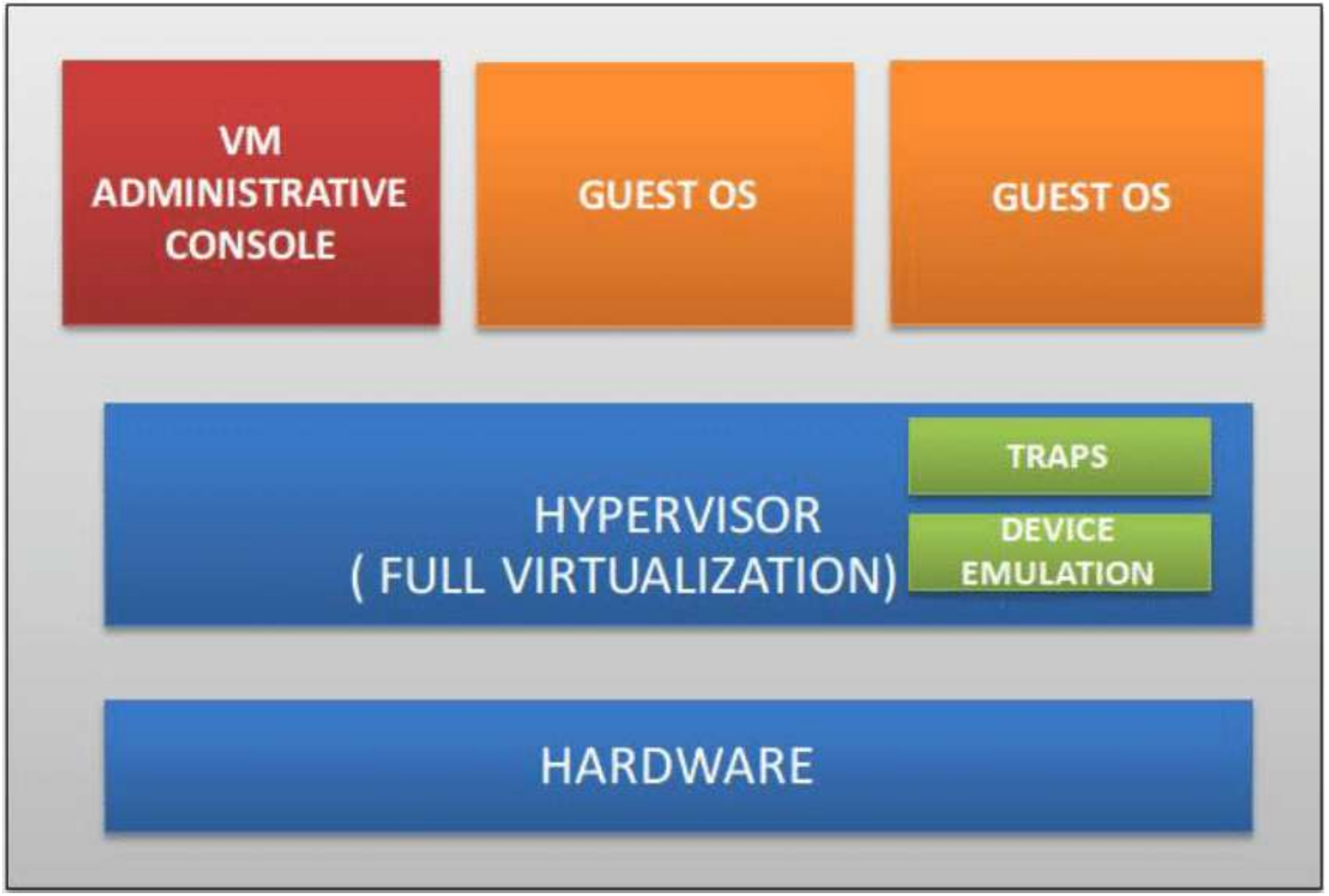
Module 3: Compute

Section 4: Container services

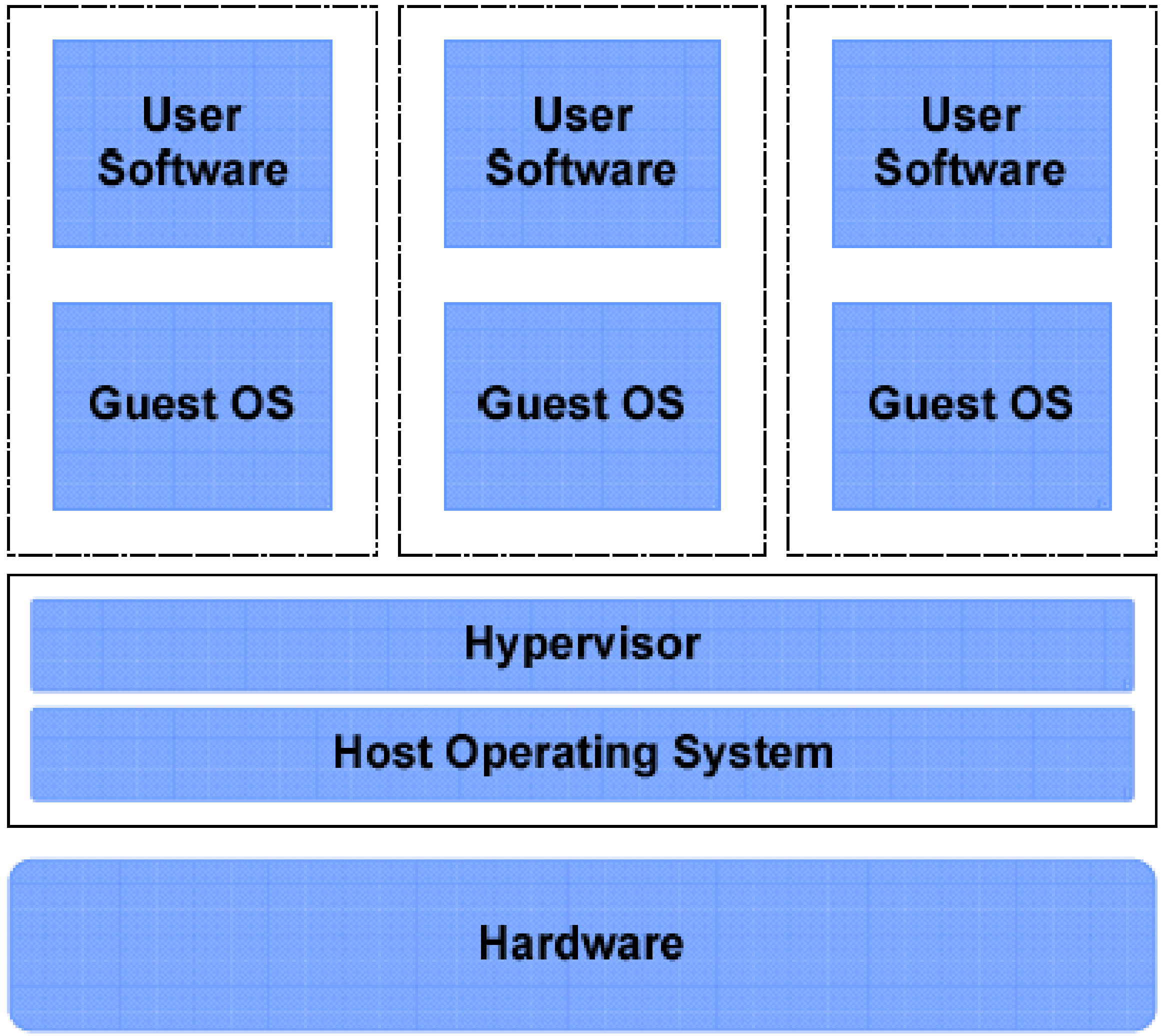
Virtualization



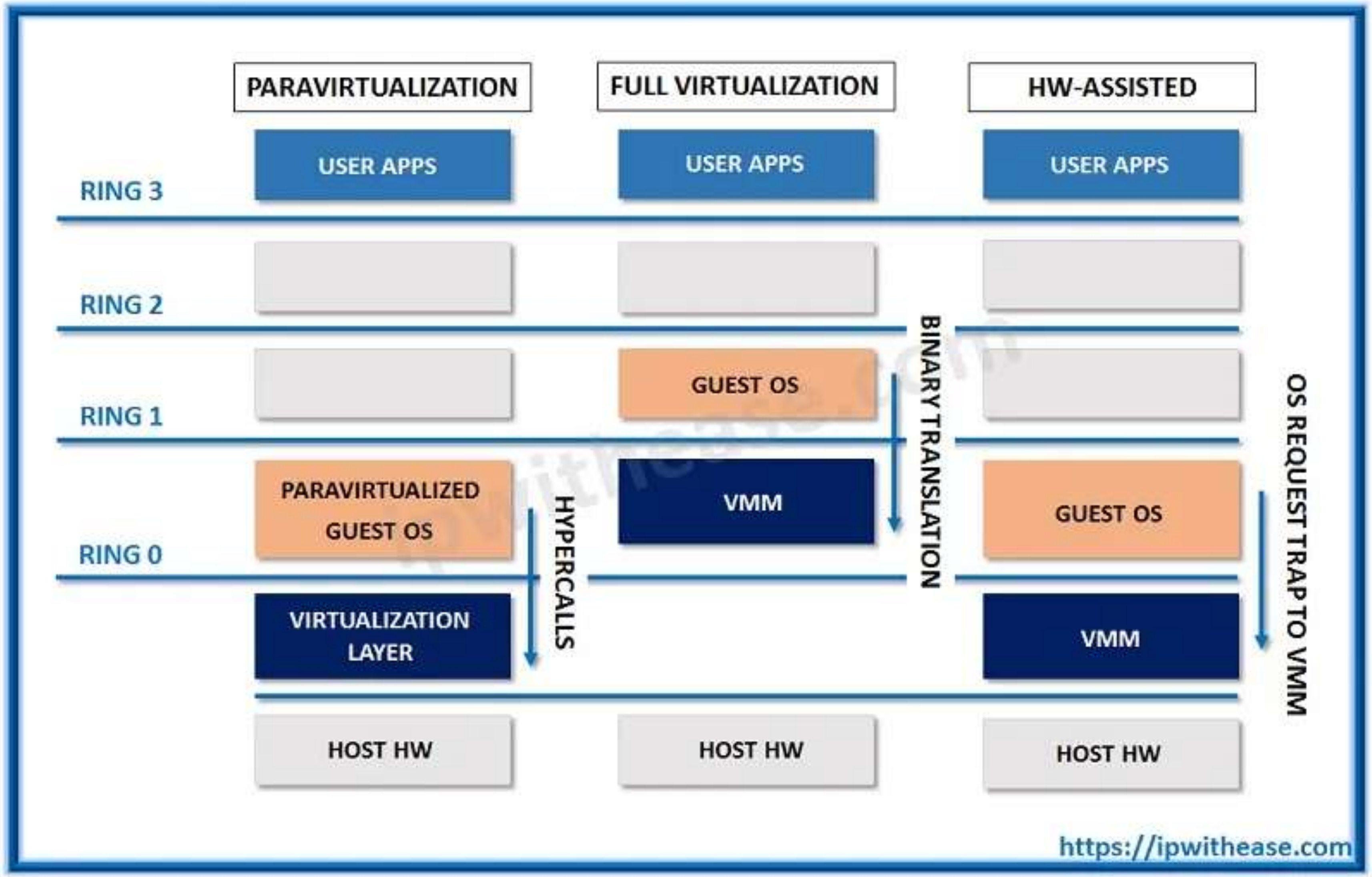
Full Virtualization



Para Virtualization



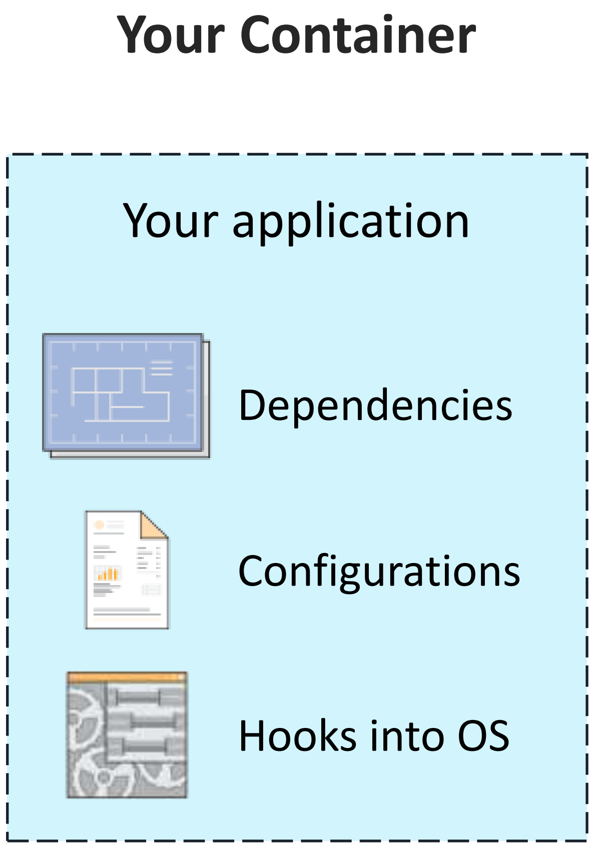
Comparison


PARAVIRTUALIZATION
FULL VIRTUALIZATION
HW-ASSISTED
RING 3
RING 2
RING 1
RING 0
USER APPS
USER APPS
USER APPS
GUEST OS
PARAVIRTUALIZED GUEST OS
VMM
GUEST OS
VIRTUALIZATION LAYER
VMM
HYPERCALLS
BINARY TRANSLATION
OS REQUEST TRAP TO VMM
HOST HW
HOST HW
HOST HW
https://ipwithease.com

Container basics

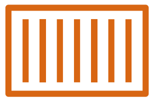
- **Containers** are a method of operating system virtualization.

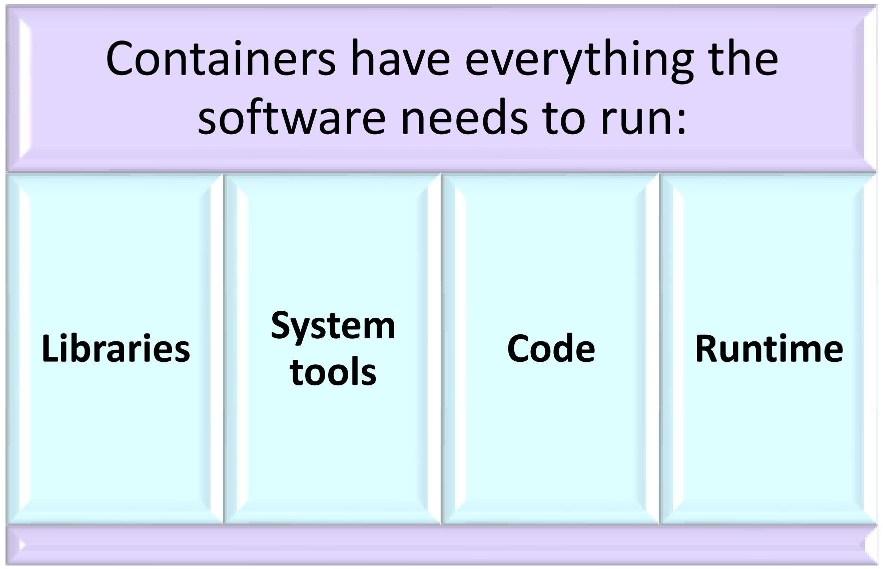
- Benefits –
  - Repeatable.
  - Self-contained environments.
  - Software runs the same in different environments.
    - Developer's laptop, test, production.
  - Faster to launch and stop or terminate than virtual machines



What is Docker?

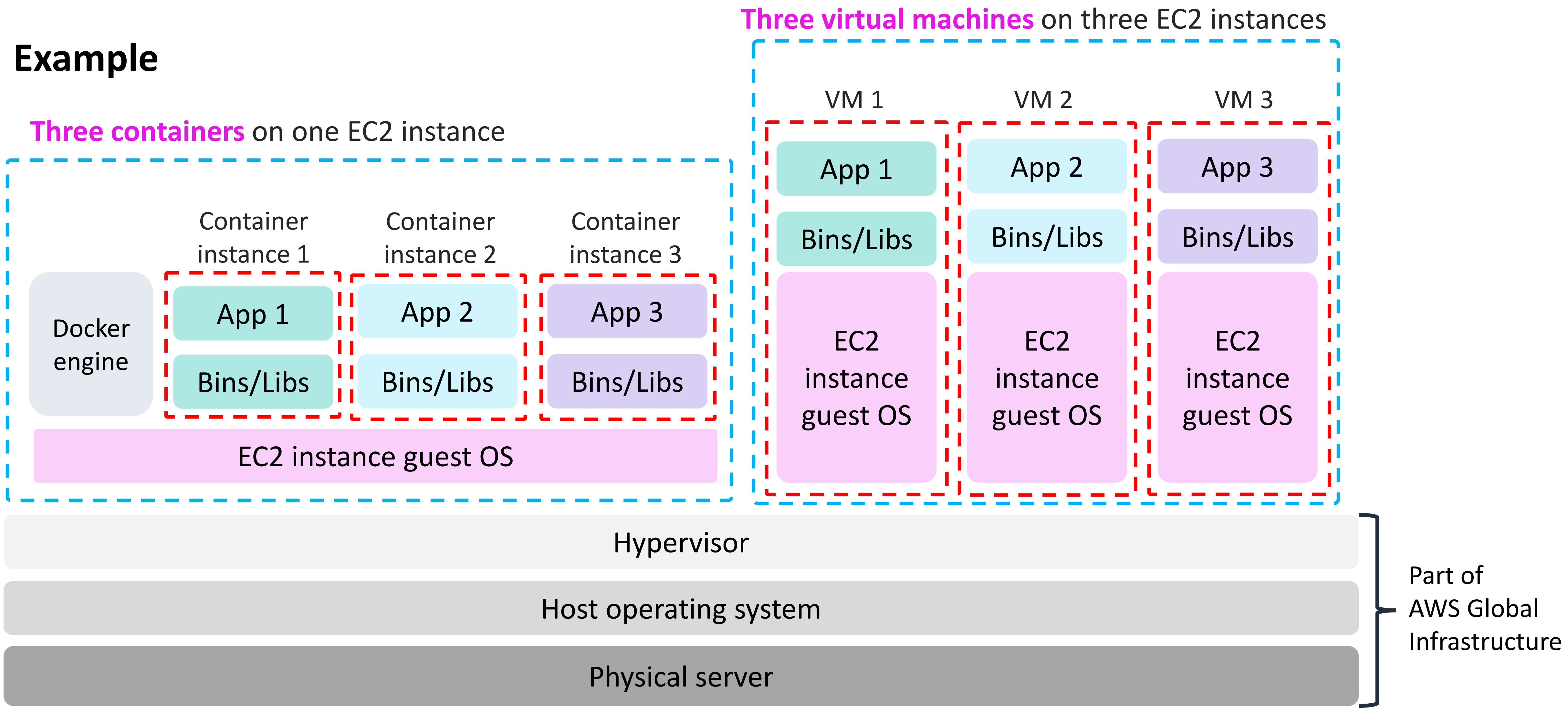
- **Docker** is a software platform that enables you to build, test, and deploy applications quickly.
- You run containers on Docker.
  - Containers are created from a template called an *image*.
- A **container** has everything a software application needs to run.


Container



Containers versus virtual machines

Example



Amazon Elastic Container Service (Amazon ECS)

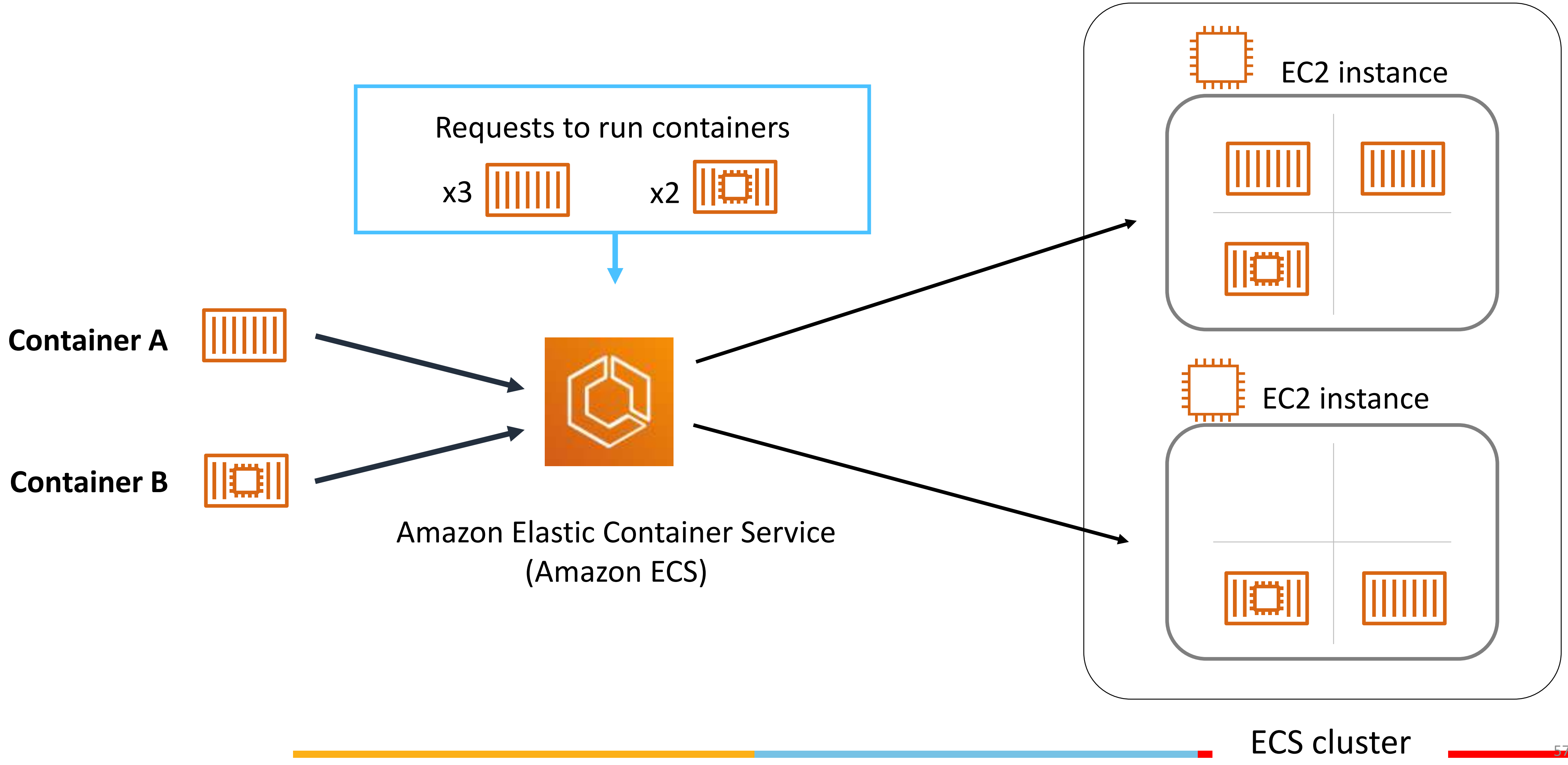
- Amazon Elastic Container Service (**Amazon ECS**) –
  - A highly scalable, fast, container management service

- 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

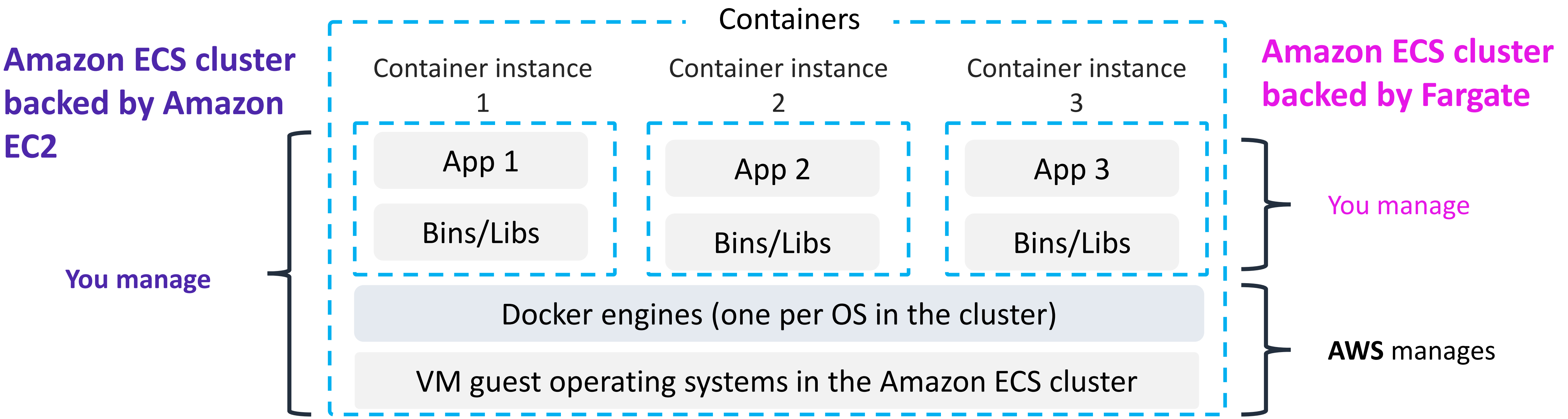
Amazon Elastic Container Service

Amazon ECS orchestrates containers



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)


Amazon ECS cluster backed by Amazon EC2
You manage
Containers
Container instance 1
Container instance 2
Container instance 3
App 1
Bins/Libs
App 2
Bins/Libs
App 3
Bins/Libs
Docker engines (one per OS in the cluster)
VM guest operating systems in the Amazon ECS cluster
Amazon ECS cluster backed by Fargate
You manage
AWS manages

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

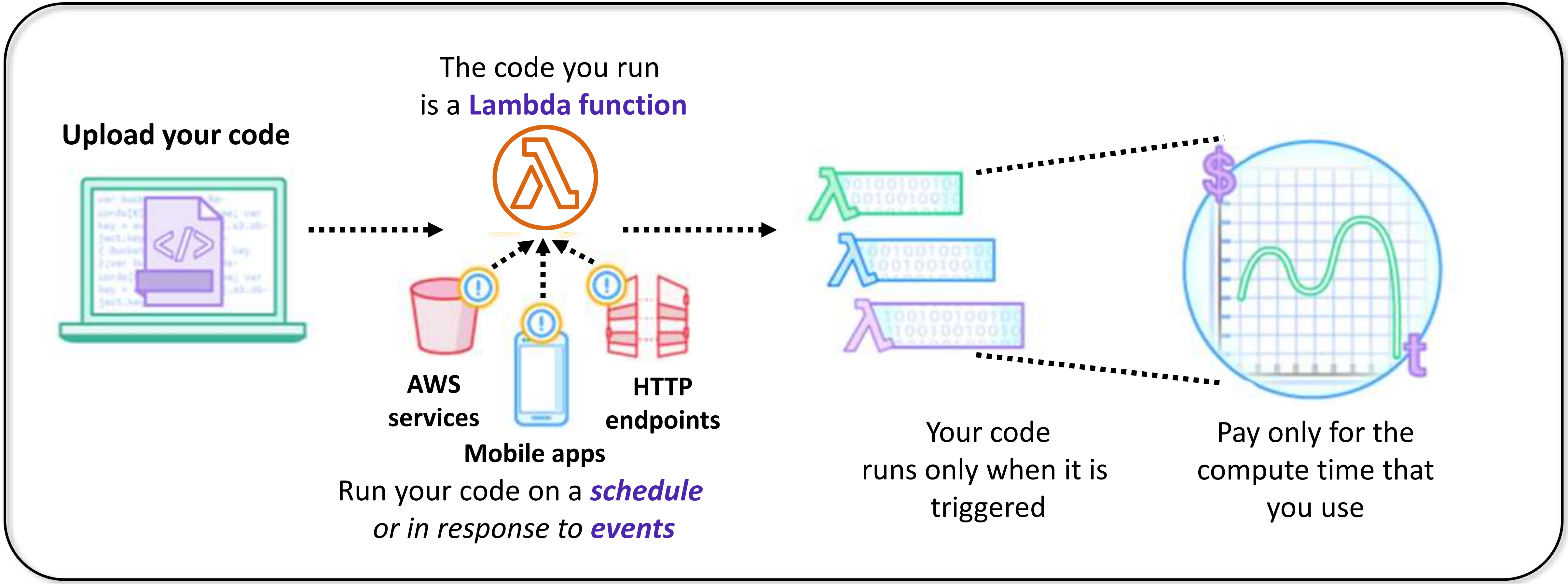
- **Amazon ECS integration**
- **Docker support**
- **Team collaboration**
- **Access control**
- **Third-party integrations**

Compute

Section 5: Introduction to AWS Lambda

AWS Lambda: Run code without servers

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



Benefits of Lambda

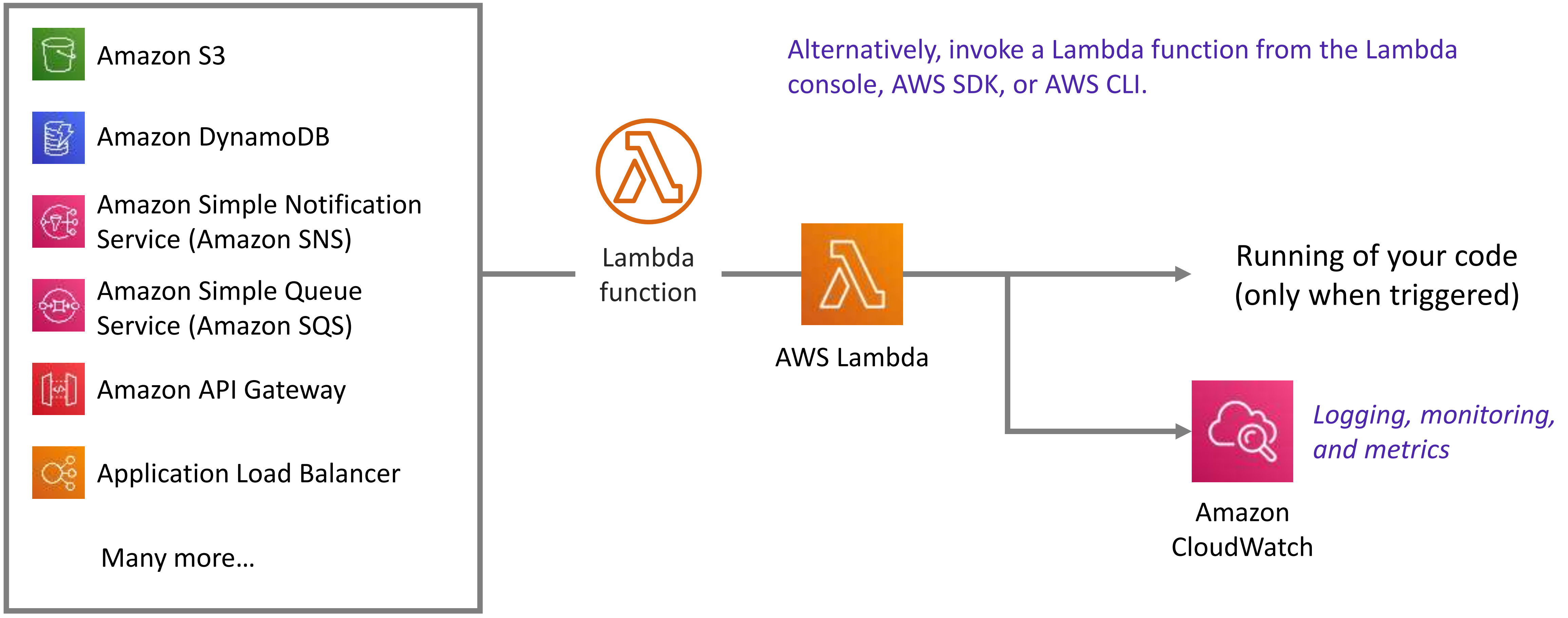
AWS Lambda

- 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

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

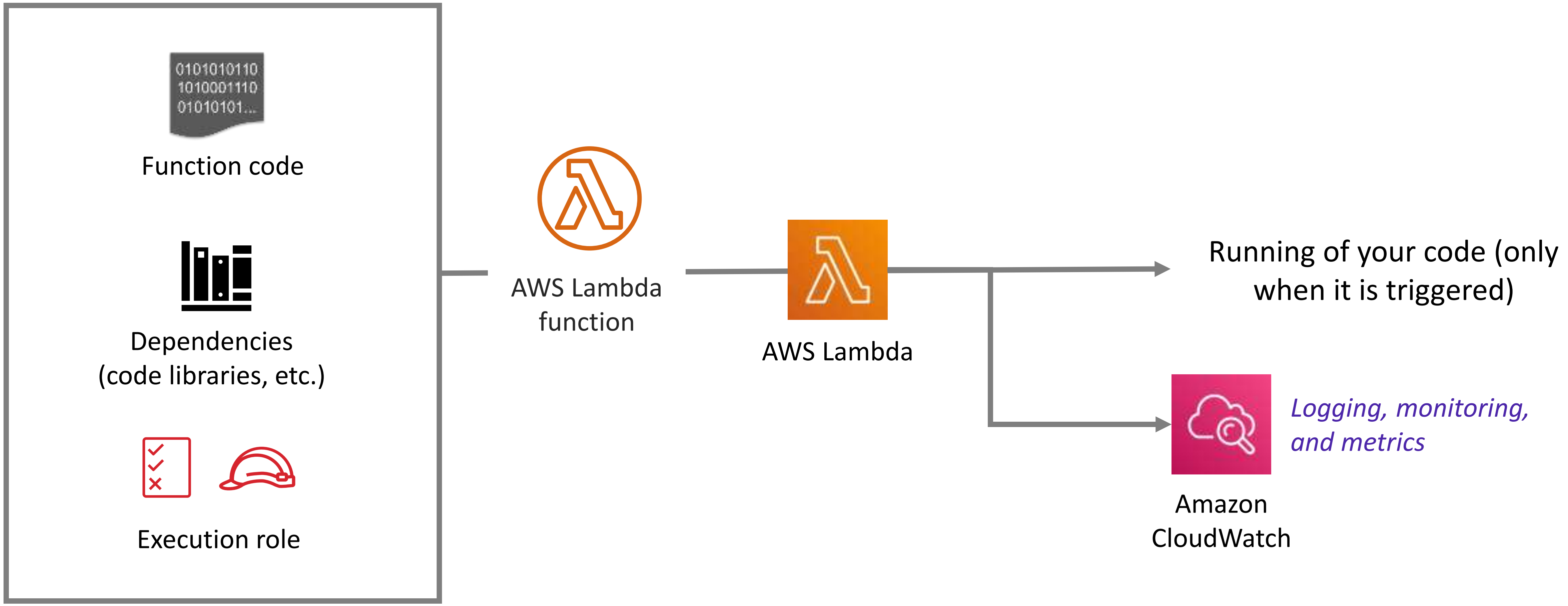
Alternatively, invoke a Lambda function from the Lambda console, AWS SDK, or AWS CLI.



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
Running of your code (only when triggered)
Amazon CloudWatch
Logging, monitoring, and metrics

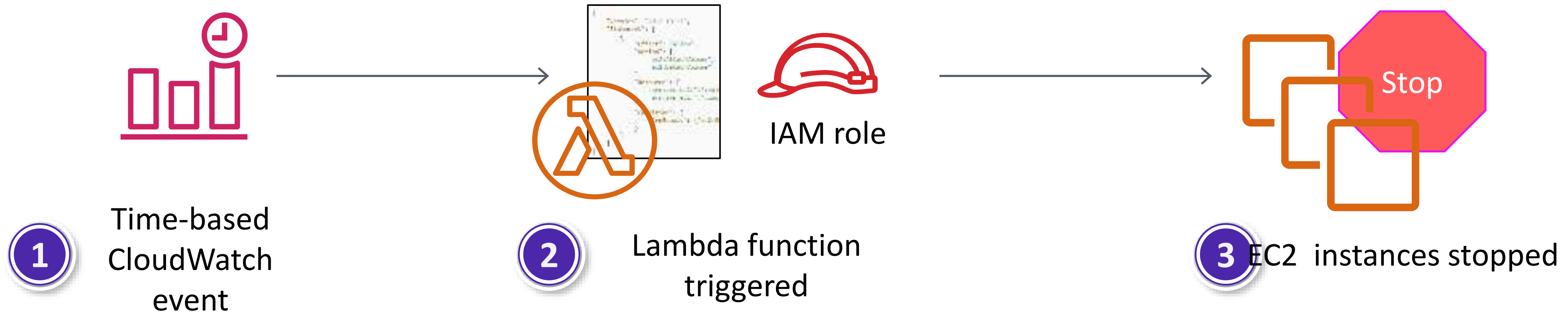
AWS Lambda function configuration

Lambda function configuration

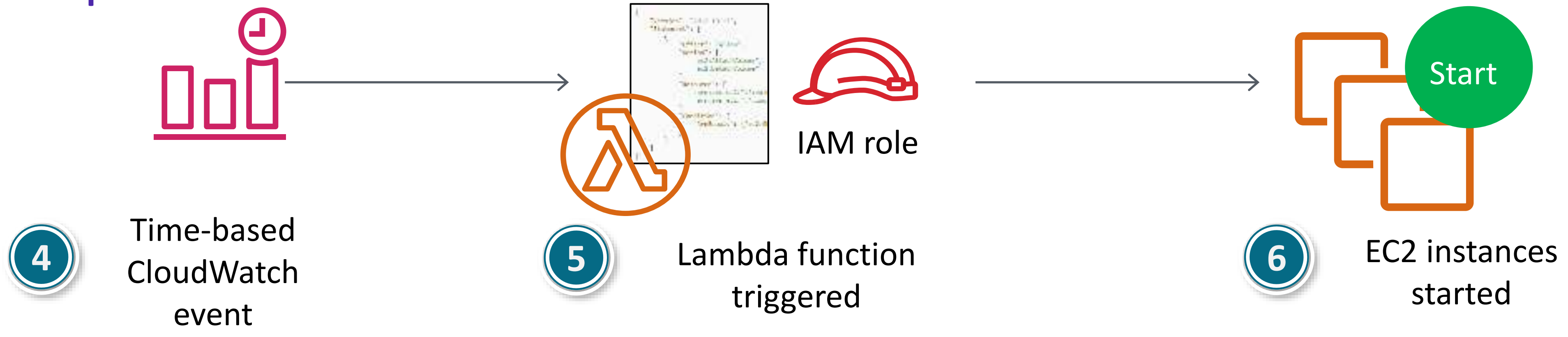


Schedule-based Lambda function example: Start and stop EC2 instances

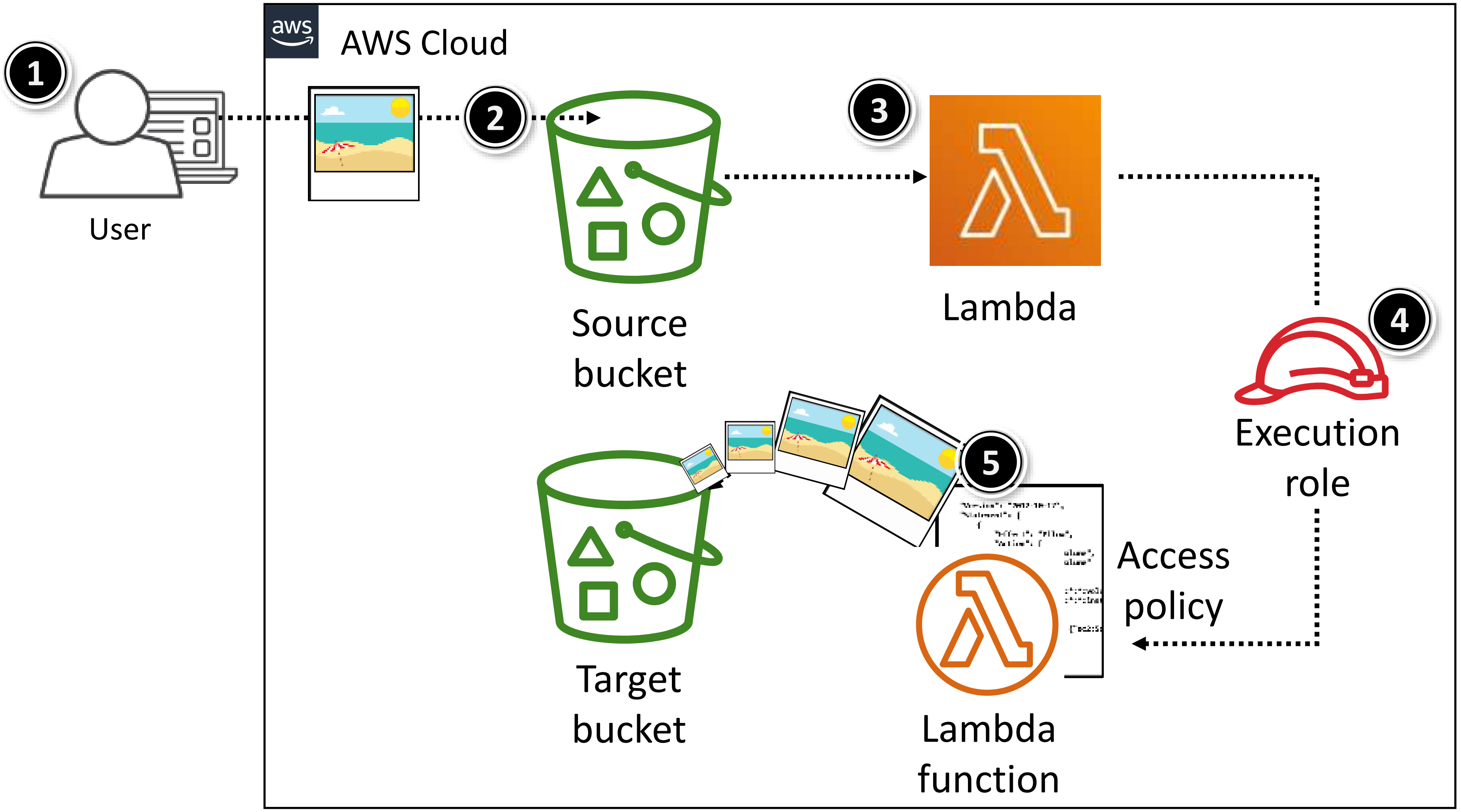
Stop instances example



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.

Compute

Section 6: Introduction to AWS Elastic Beanstalk

AWS Elastic Beanstalk

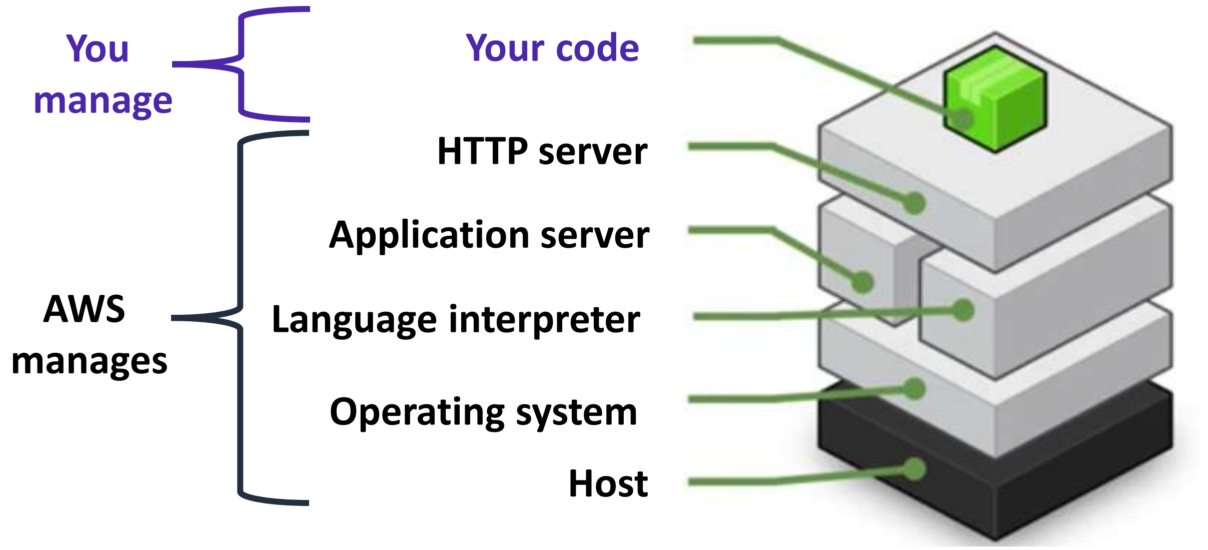
AWS Elastic Beanstalk

- 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

Developer productivity

Difficult to outgrow

Complete resource control

Module summary

In summary, in this module, you learned how to:

- Provide an overview of different AWS compute services in the cloud
- Demonstrate why to use Amazon Elastic Compute Cloud (Amazon EC2)
- Identify the functionality in the Amazon EC2 console
- Perform basic functions in Amazon EC2 to build a virtual computing environment
- Identify Amazon EC2 cost optimization elements
- Demonstrate when to use AWS Elastic Beanstalk
- Demonstrate when to use AWS Lambda
- Identify how to run containerized applications in a cluster of managed servers