COMP5349– Cloud Computing
Week 3: Virtualization & EC2

Dr. Ying Zhou

The University of Sydney

Table of Contents

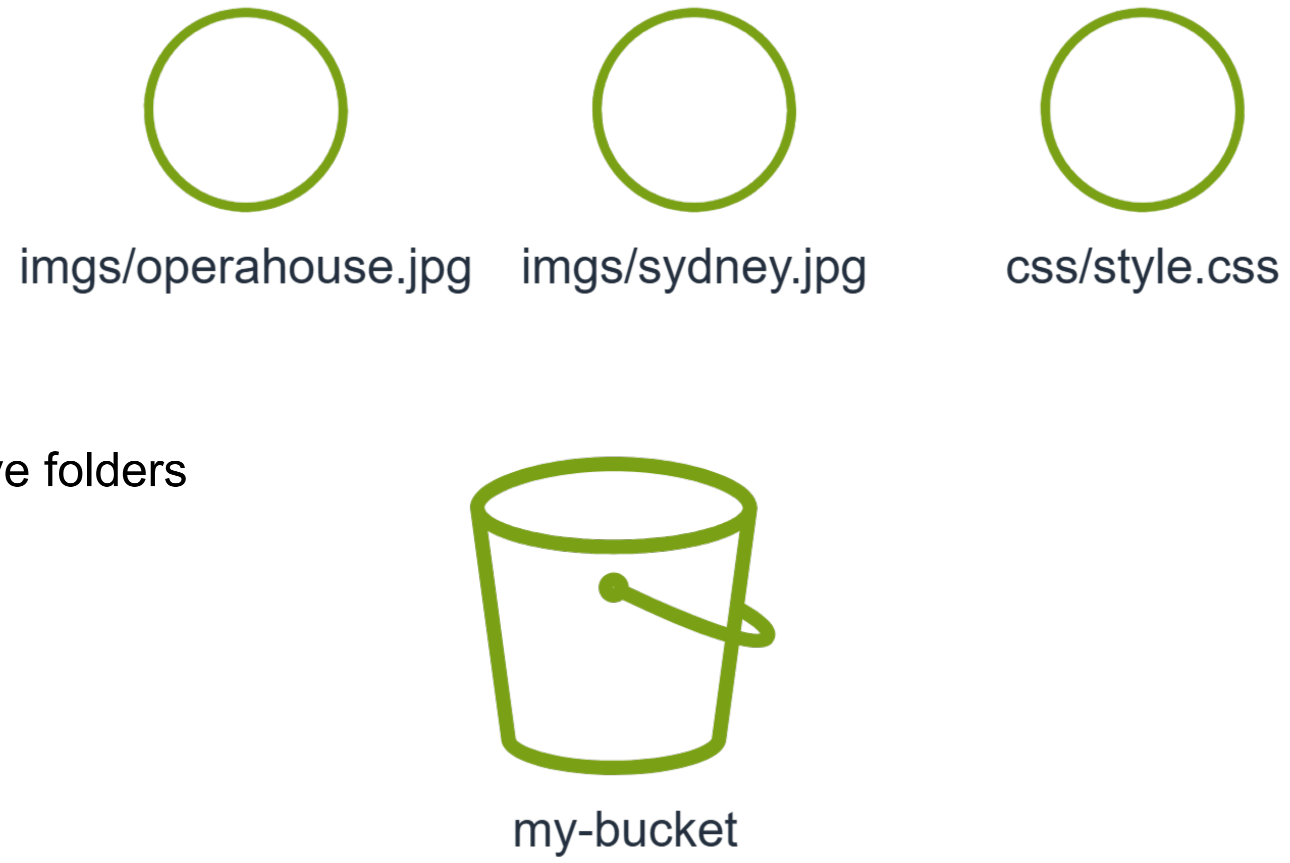
S3 Review

The pseudo folder structure concept - 1

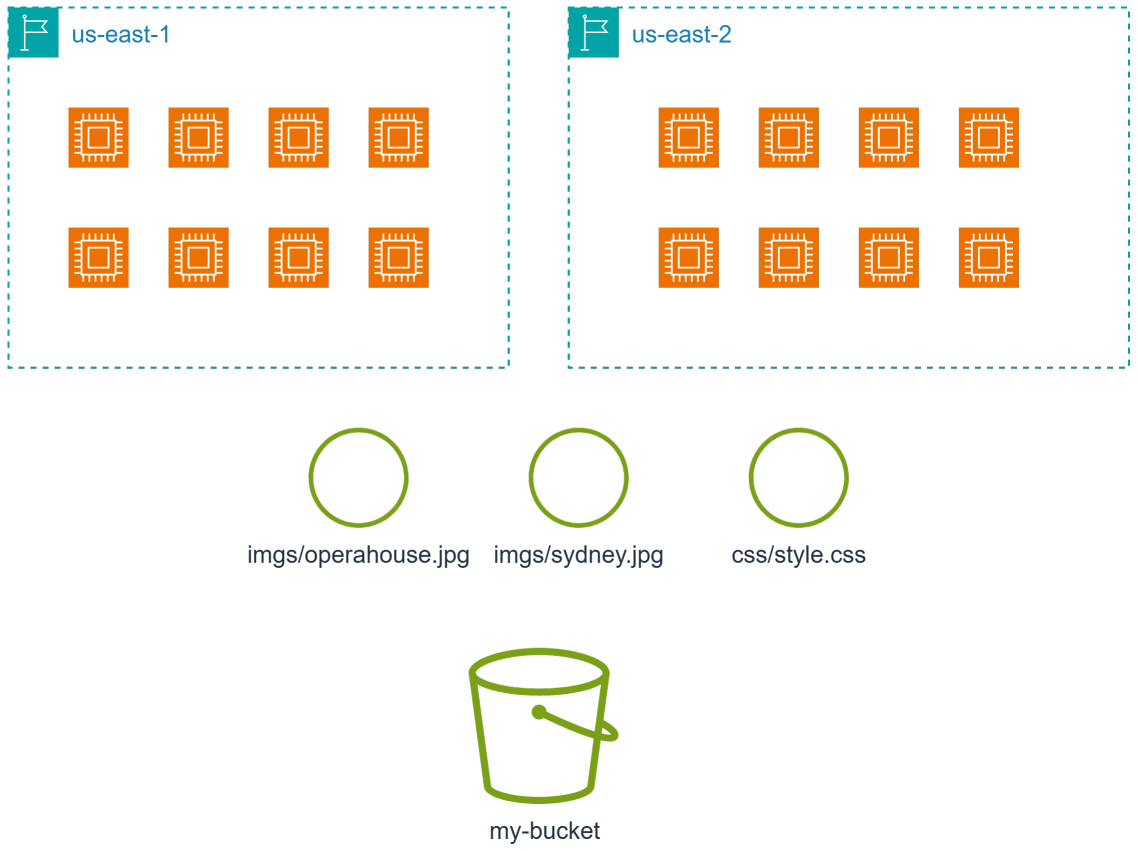
Assuming we have a bucket: **my-bucket**

We create two “folders” : **imgs , css**

Three files are uploaded to their respective folders



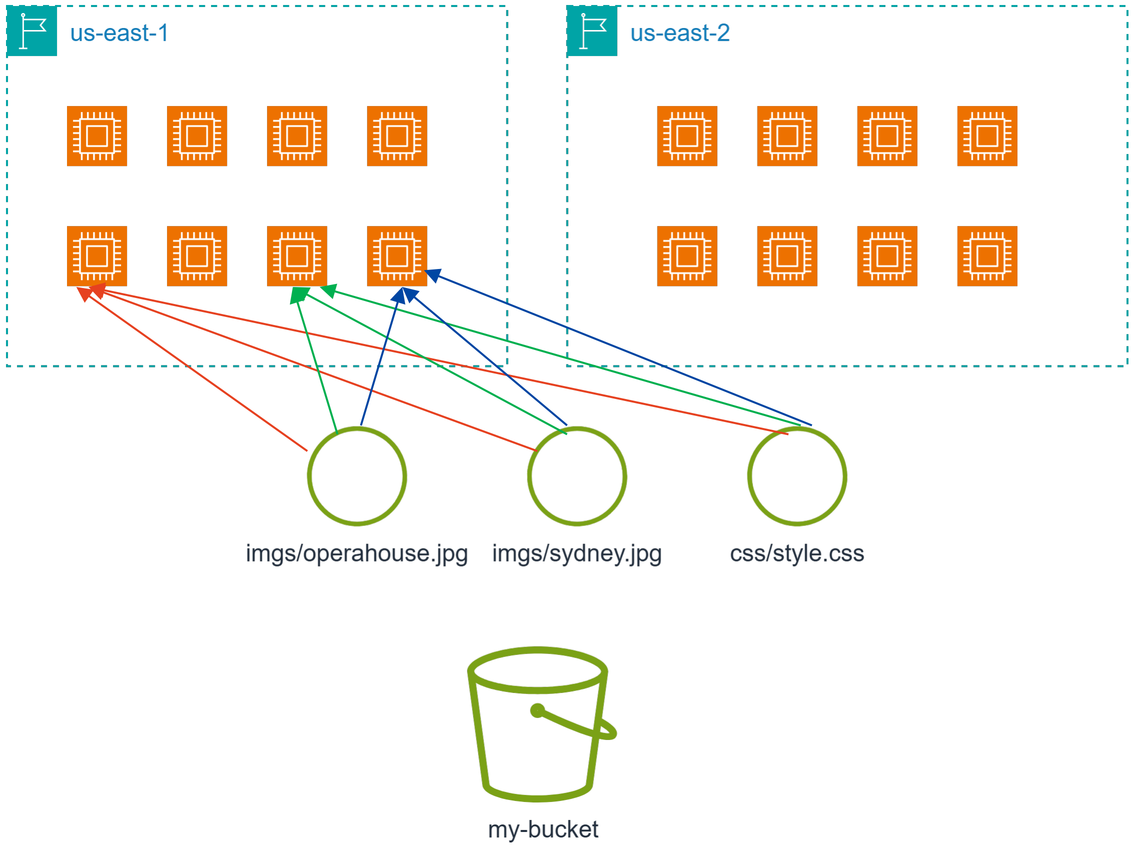
The pseudo folder structure concept - 2



Assuming each object is duplicated 3 times

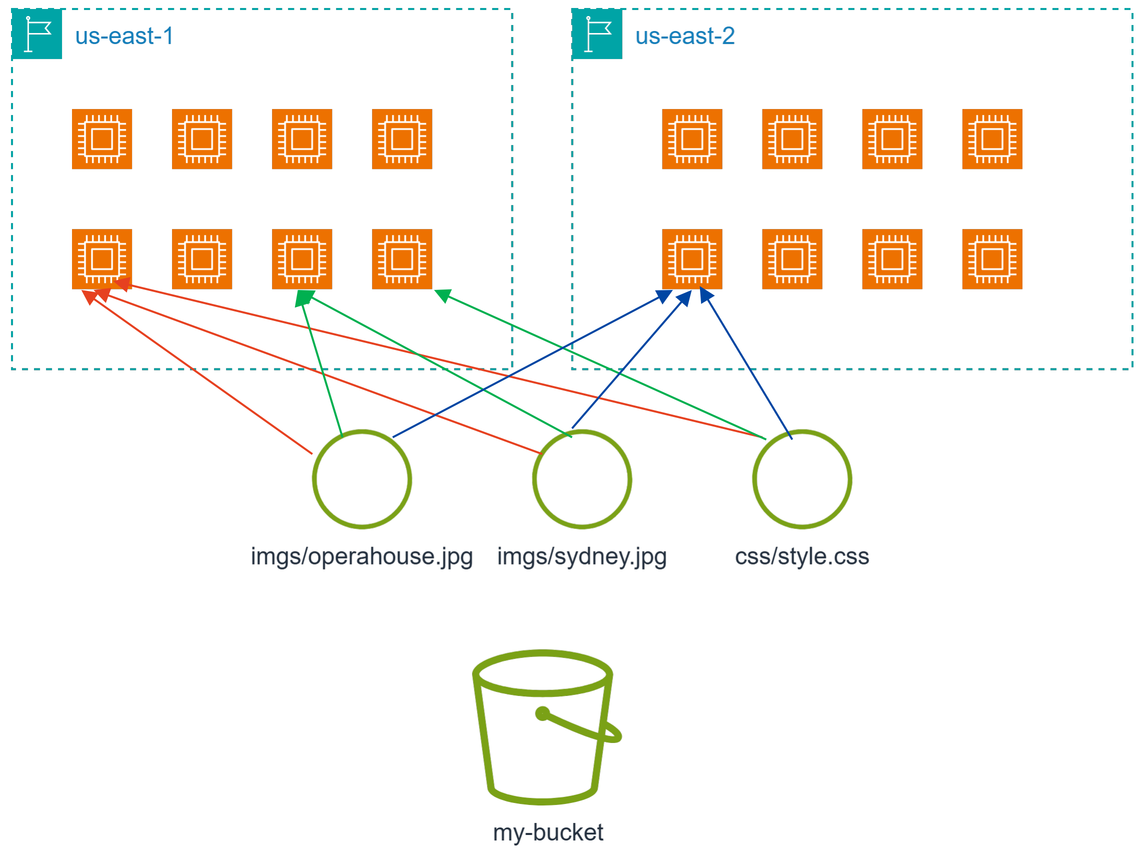
There are storage nodes on each region to store S3 objects

The pseudo folder structure concept - 3



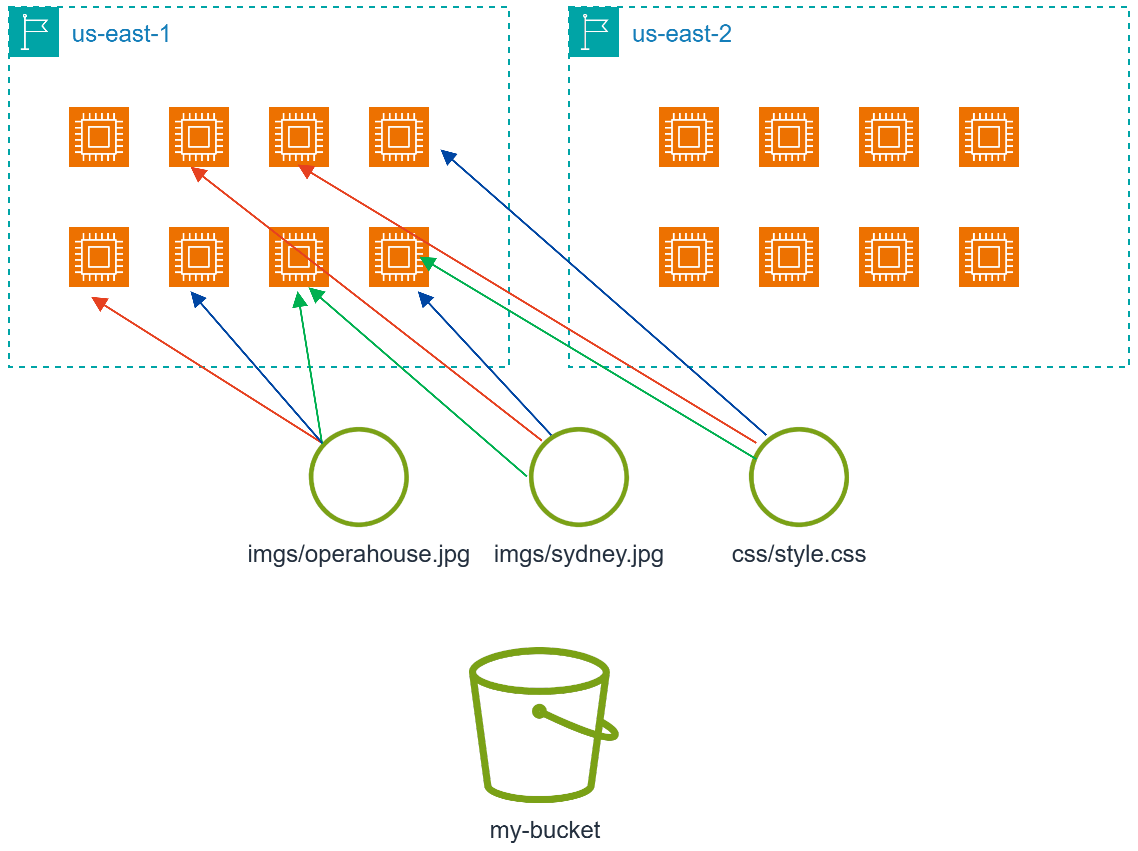
Is this storage plan work

The pseudo folder structure concept - 4



Does this plan work

The pseudo folder structure concept - 5



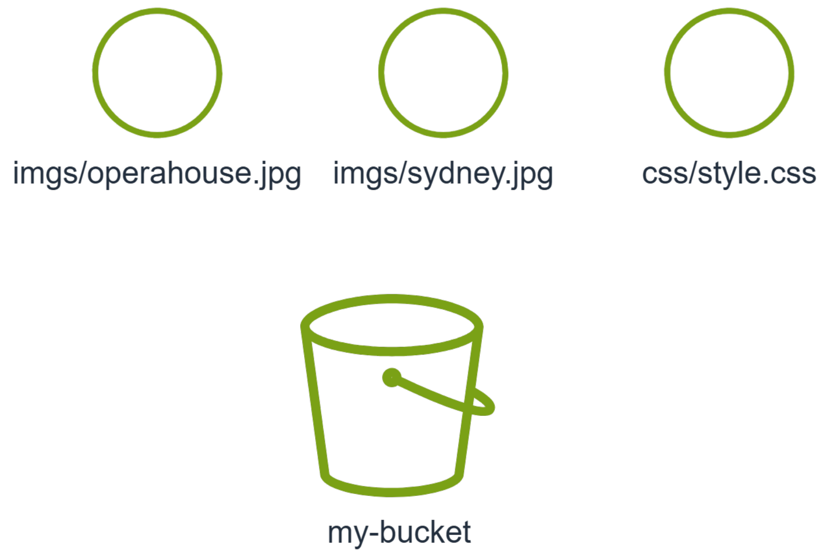
Does this plan work

The benefits of flat structure

- A S3 bucket can store an unlimited number of objects
- Objects belonging to the same bucket do not necessarily reside on the same physical machine
- Objects in the same “folder” do not necessarily reside on the same physical machine
- Object key (bucket_name + object_prefix + object_file name) determines the physical location of the objet

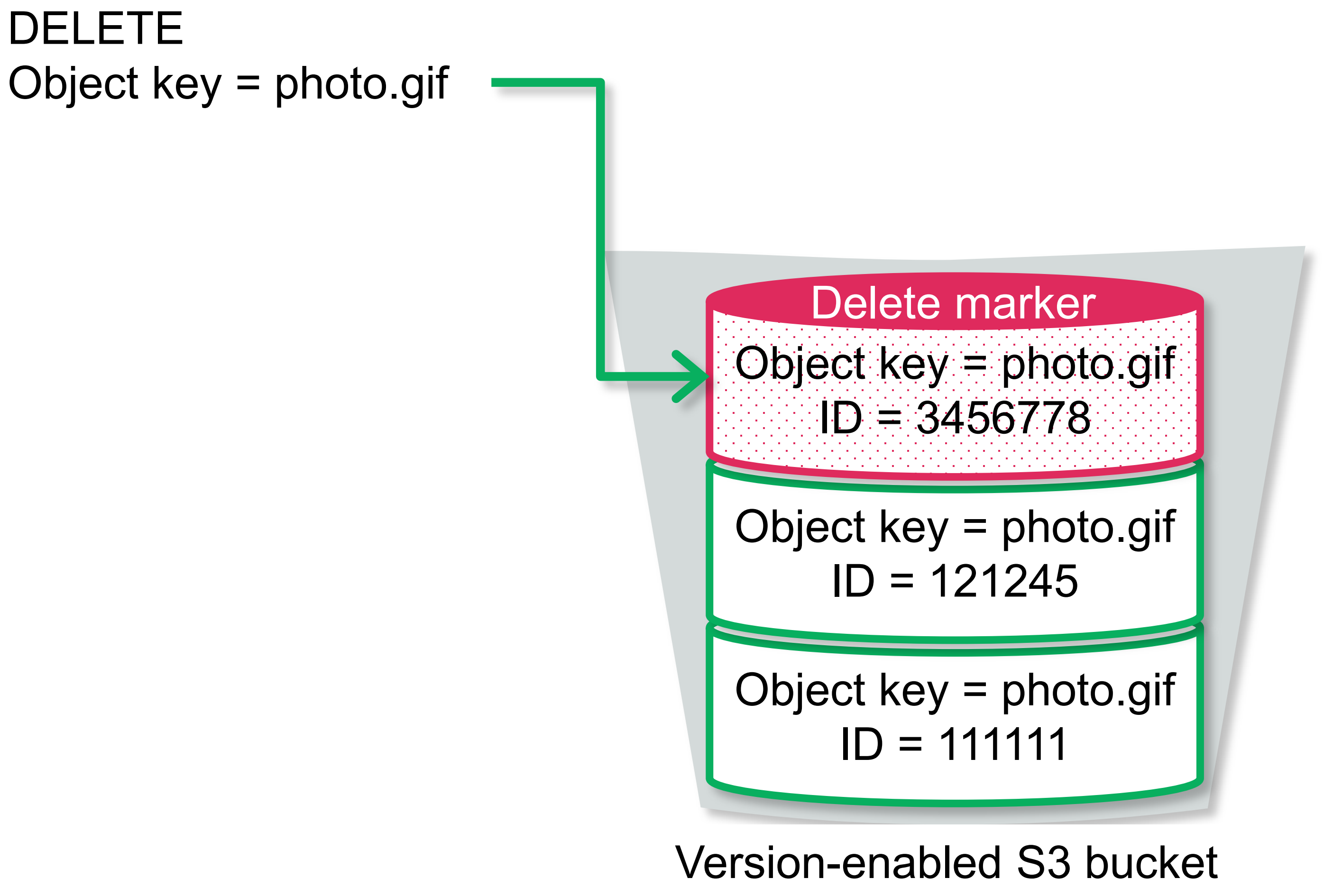
The object concept - 1

- Most objects stored in S3 are files, uploaded by user or stored by other AWS services
- There are some special objects
  - Folder is a 0 byte object
  - Delete marker in versioned S3 bucket is a special object



There are five objects in my-bucket

The object concept - 2



There are three objects after the delete operation

The delete operation creates a special object delete marker

All three objects have the same key

They are differentiated by the version ID

Virtualization Overview

Cloud and enabling technologies

- Cloud computing allows users to rent various resources from providers on hourly(secondly), daily, monthly, yearly base
- A few general models, e.g. XaaS, are used to describe the spectrum of cloud computing
- IaaS and many PaaS services use virtualization technology to present one or many virtual machines to the user
  - Virtualization technology is relatively standard and mature, particularly with respect to allocating CPU and memories

What is virtualization?

- Narrow and very common perception of virtualization in cloud era
    - “Virtualization lets you run *multiple virtual machines* on a *single physical machine*, with each virtual machine sharing the resources of that *one physical computer* across *multiple environments*.”
- Broadly, the term is used in many other contexts, some are closely related with each other
    - E.g. virtual memory, storage, network, virtual reality

Early virtualization examples I

- Virtual Memory
  - Provide more accessible memories to CPU than those physically presented on the board
  - An important component of modern operating systems
    - Benefit of expansion and increasing code reusability.
  - Actions involved
    - Address translation
    - Data transfer
    - Data placement strategy for swap in/out

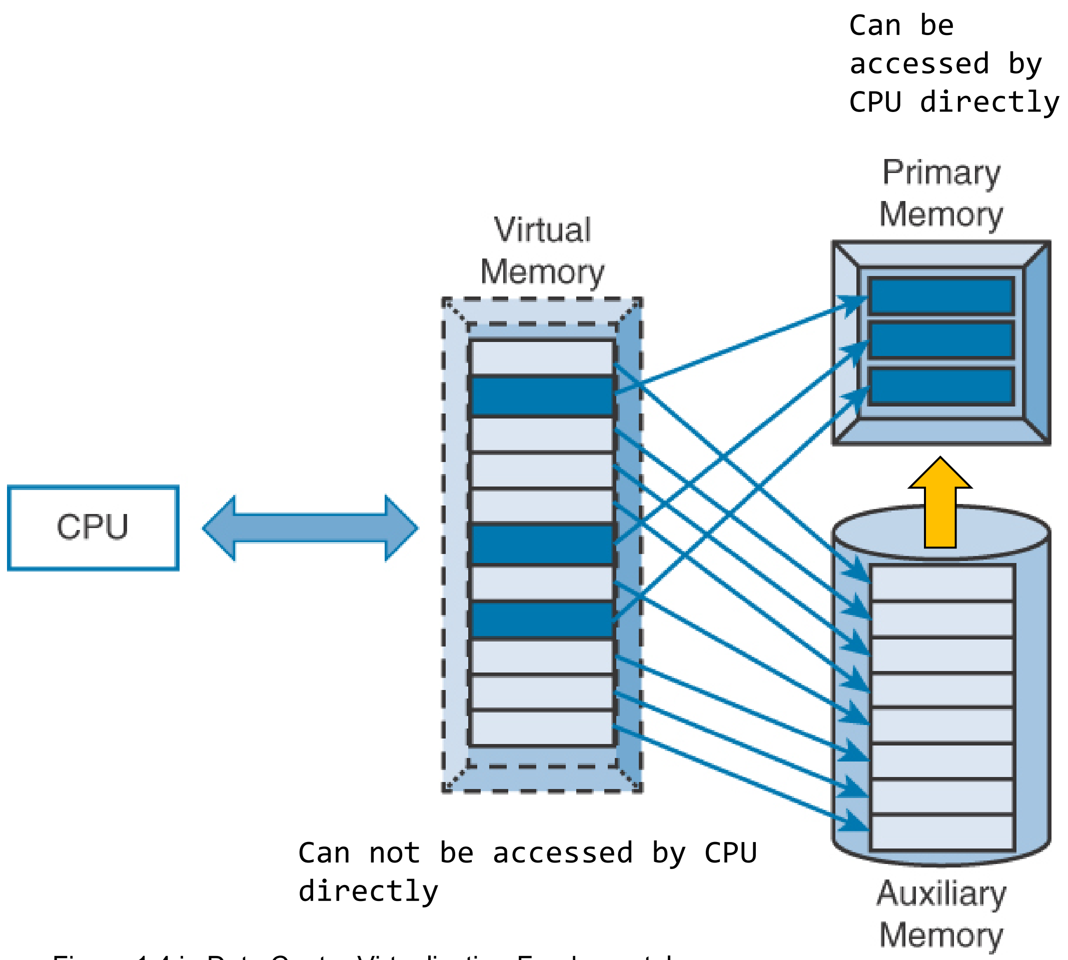

Can be accessed by CPU directly
Primary Memory
Virtual Memory
CPU
Can not be accessed by CPU directly
Auxiliary Memory

Figure 1.4 in Data Center Virtualization Fundamentals

Early virtualization examples II

- Mainframe Virtualization
  - IBM first released **mainframe virtualization solution in 1972**
  - The overall architecture looks similar to modern system VM
  - Solve the problem of OS migration triggered by hardware release
  - It was initialized by the **time-sharing technologies in 1960s**

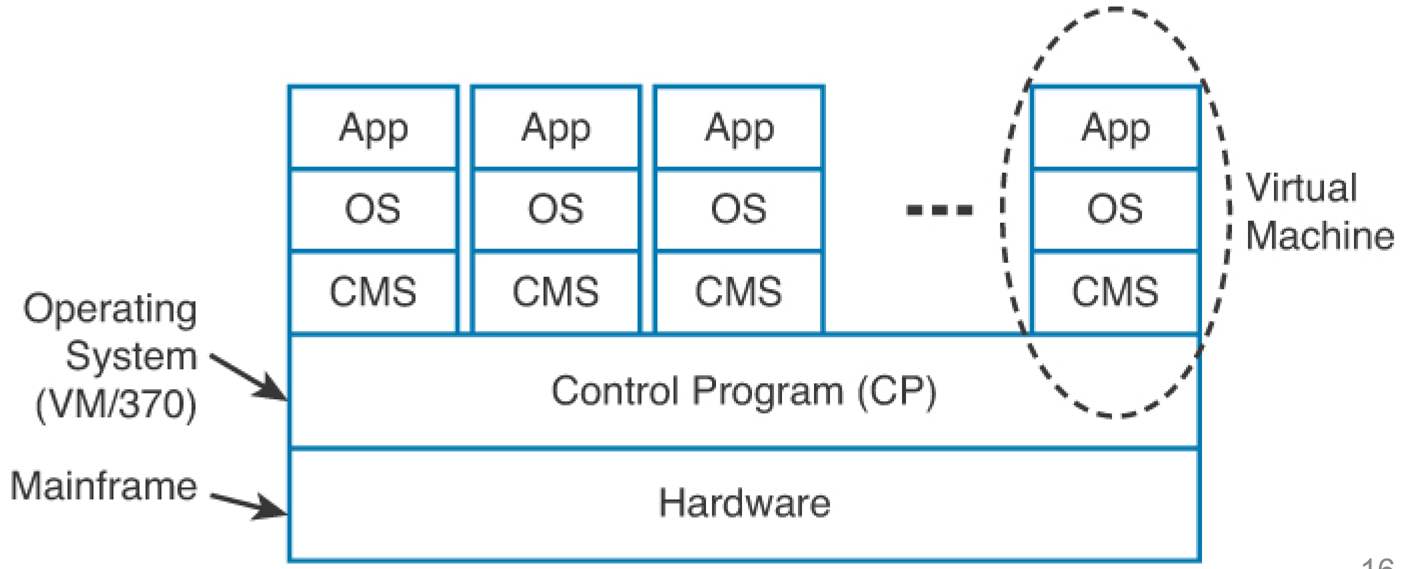


Figure 1.5 in Data Center Virtualization Fundamentals

Early virtualization examples III

- Hot Standby Router Protocol
  - Solve the problem of single point of failure
    - One host can only define a default gateway
  - Use virtual IP and preconfigured priority to decide which physical router would act as the default gateway
  - Other router with the same virtual IP can take over if the default one fails

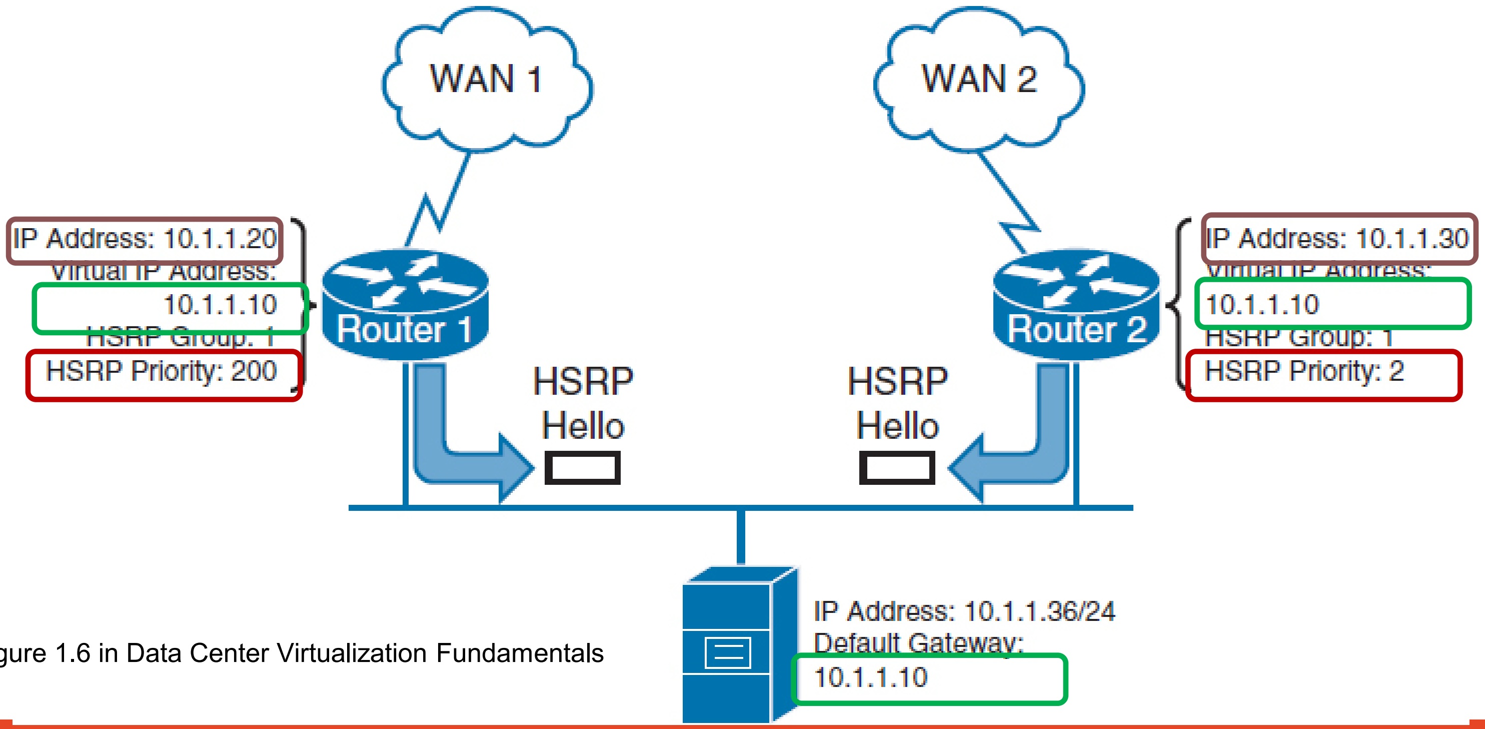
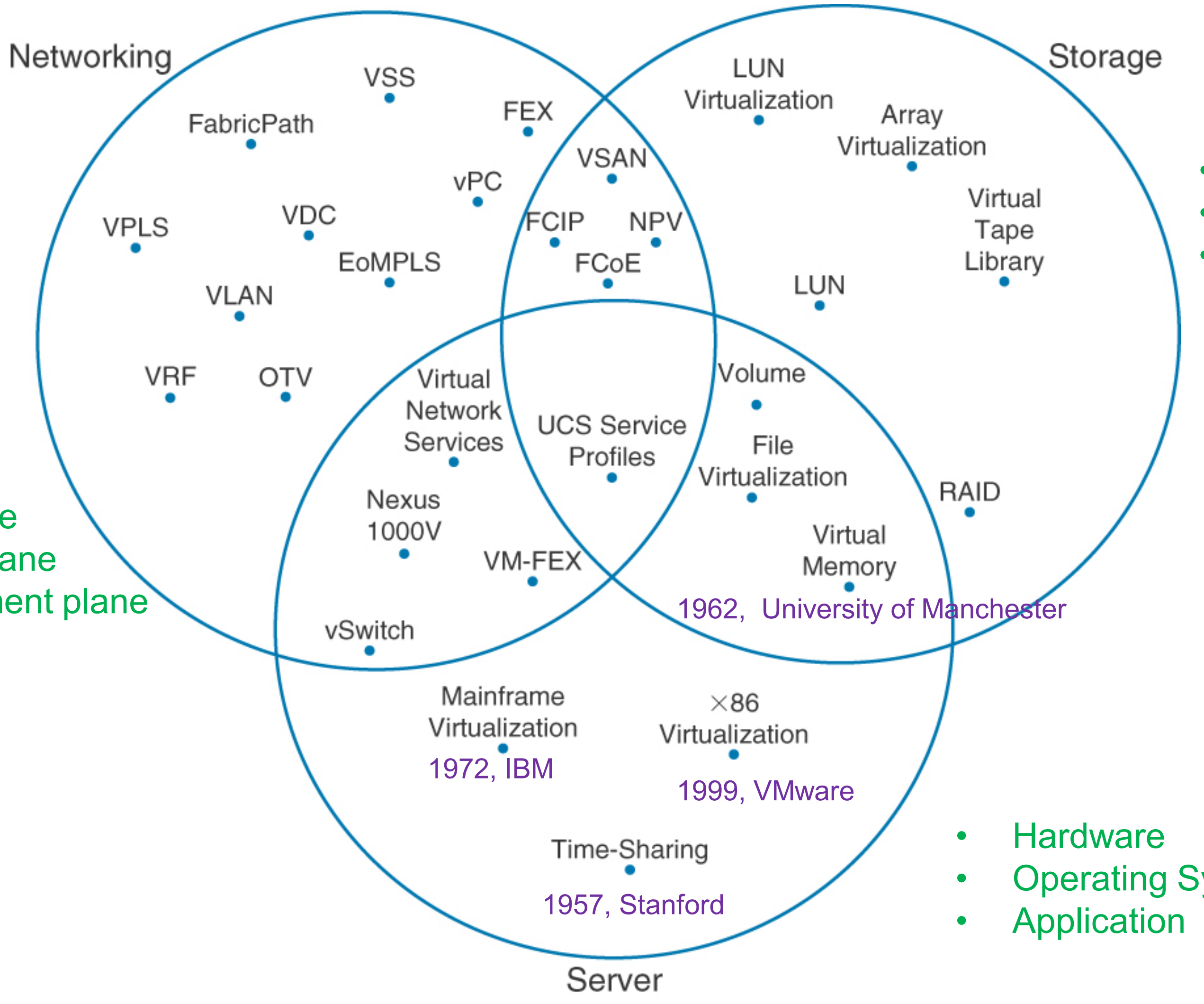


Figure 1.6 in Data Center Virtualization Fundamentals

Common features of virtualization

- Emulation
  - A preexisiting IT resources was emulated (memory, mainframe, IP address, etc)
- Transparency
  - The consumers of the resources (CPU, mainframe users, network hosts) cannot distinguish between real and emulated resources
- Benefits
  - Compared with directly using actual resources, virtualization brings various benefits (memory expansion, resource optimization, high availability, etc)
- A broader definition of virtualization
  - “Virtualization is the transparent emulation of an IT resource producing to its consumers benefits that were unavailable in its physical form.”

Virtualization Technology Areas



- Storage device
- Host
- Interconnect

- Data plane
- Control plane
- Management plane

- Hardware
- Operating System
- Application

Figure 1.9 in Data Center Virtualization Fundamentals

Server Virtualisation

“

A *virtual machine* is a software-defined computer that runs on a physical computer with a separate operating system and computing resources. The physical computer is called the *host machine* and virtual machines are *guest machines*. Multiple virtual machines can run on a single physical machine. Virtual machines are abstracted from the computer hardware by a hypervisor.

https://aws.amazon.com/what-is/virtualization/

“

The *hypervisor* is a software component that manages multiple virtual machines in a computer. It ensures that each virtual machine gets the allocated resources and does not interfere with the operation of other virtual machines.

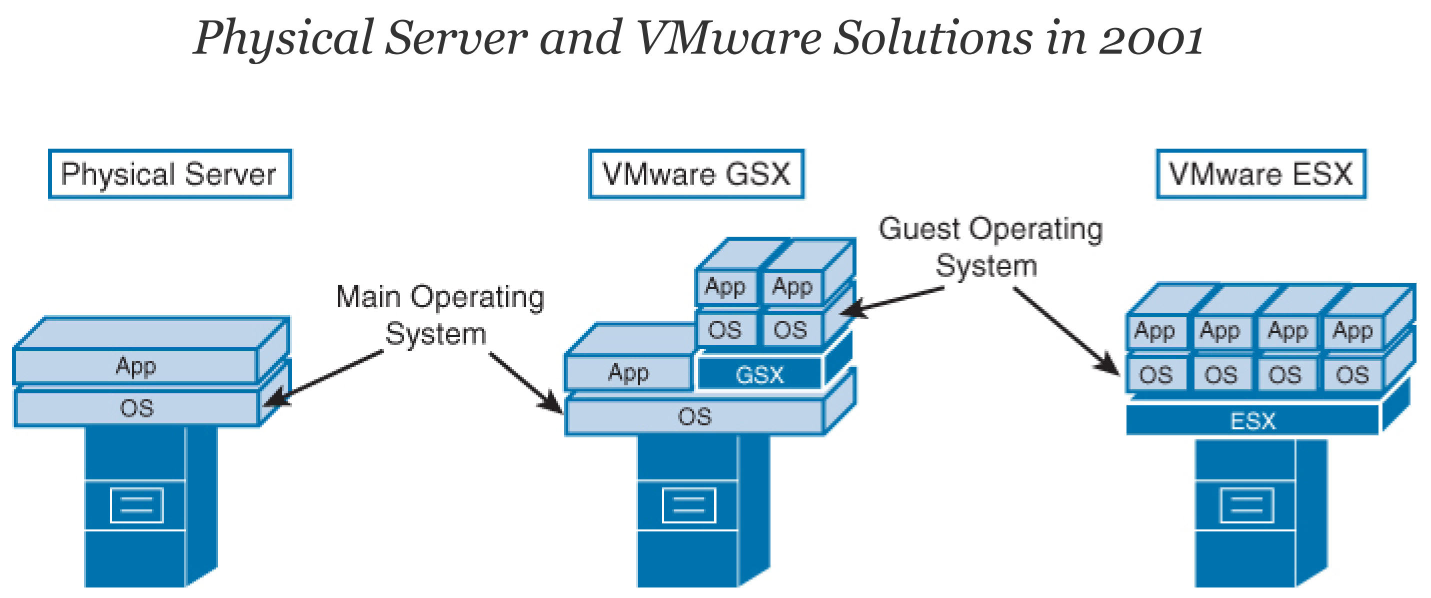
https://aws.amazon.com/what-is/virtualization/

Hypervisors

Hypervisor and OS

- Hypervisor performs a lot of traditional operating system tasks
  - *resource sharing*,
  - *device management*
  - *virtual storage management*

Two types of Hypervisor



Type 2

Hosted

Type 1

Bare-metal

Figure 13.6 in Data Center Virtualization Fundamentals

The role of a Hypervisor

- Provide an environment <u>identical</u> to the physical environment.
- Provide that environment with <u>minimal performance cost</u>.
- Retain <u>complete control</u> of the system resources

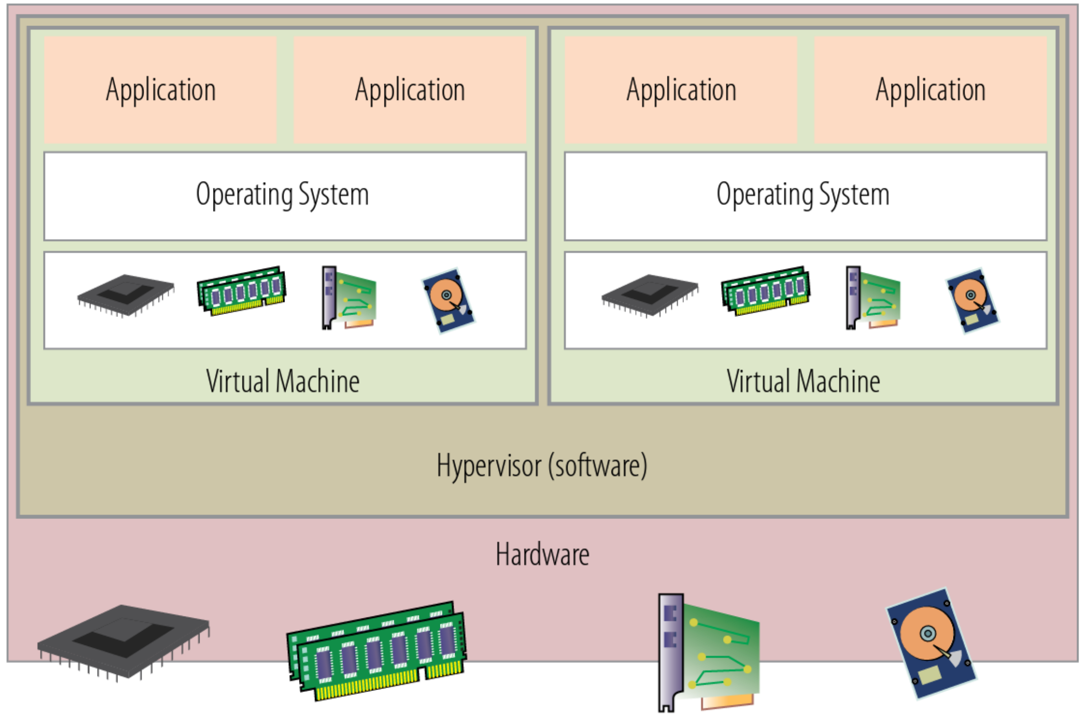
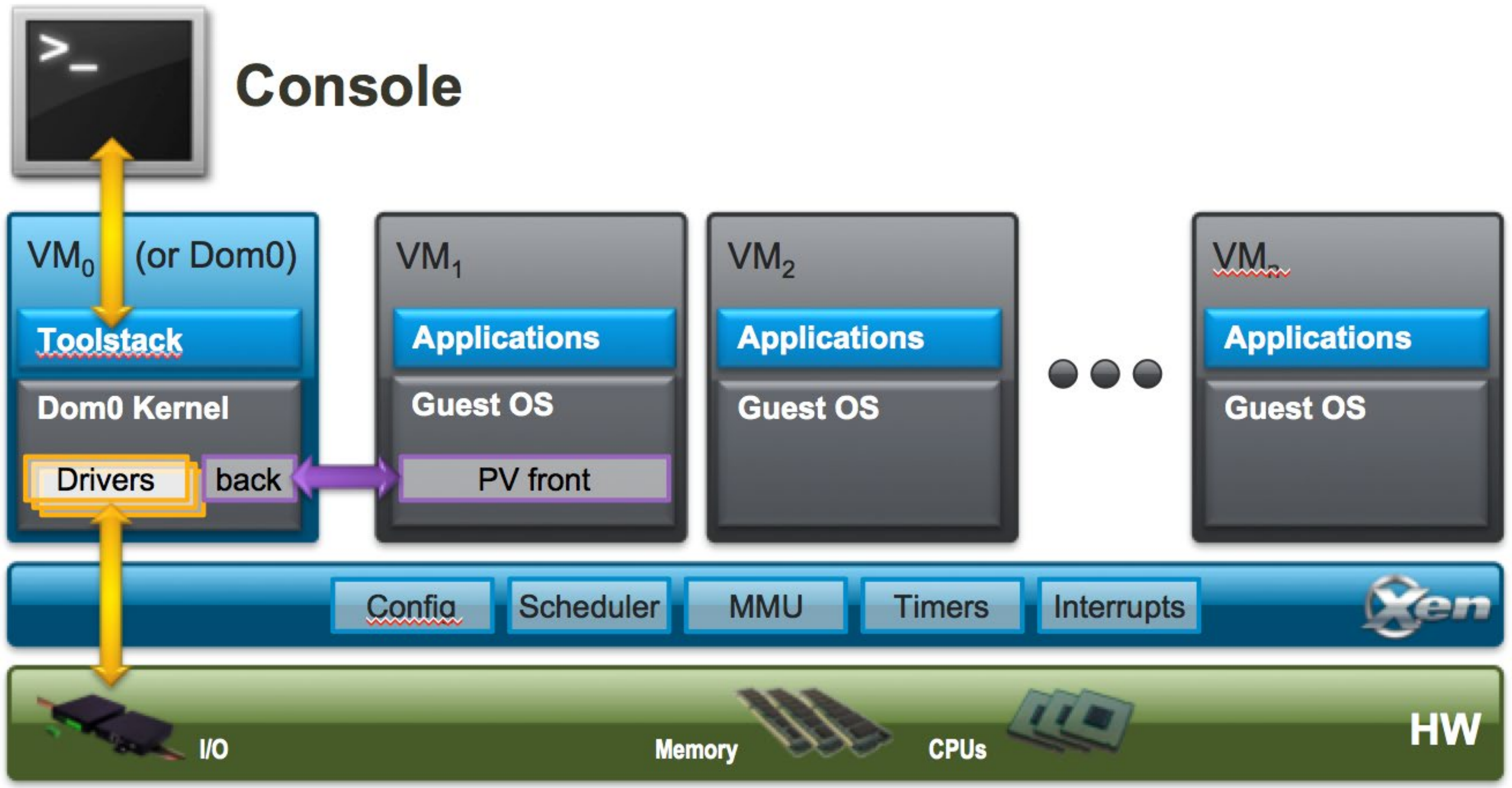


Figure 2.6 in Virtualization Essentials

Managing Resources

- Each VM is presented with a fraction of the resources of the physical host
  - E.g. A host may have 256 GB of physical memory installed in its frame, but a guest VM may believe that it has 64 GB
- Hypervisor abstracts the hardware resources and controls access to them to ensure each VM get its allocated resources
  - All access to hardware resources goes through hypervisor
    - Access to memory, storage, network, etc..

Example Hypervisors: Xen

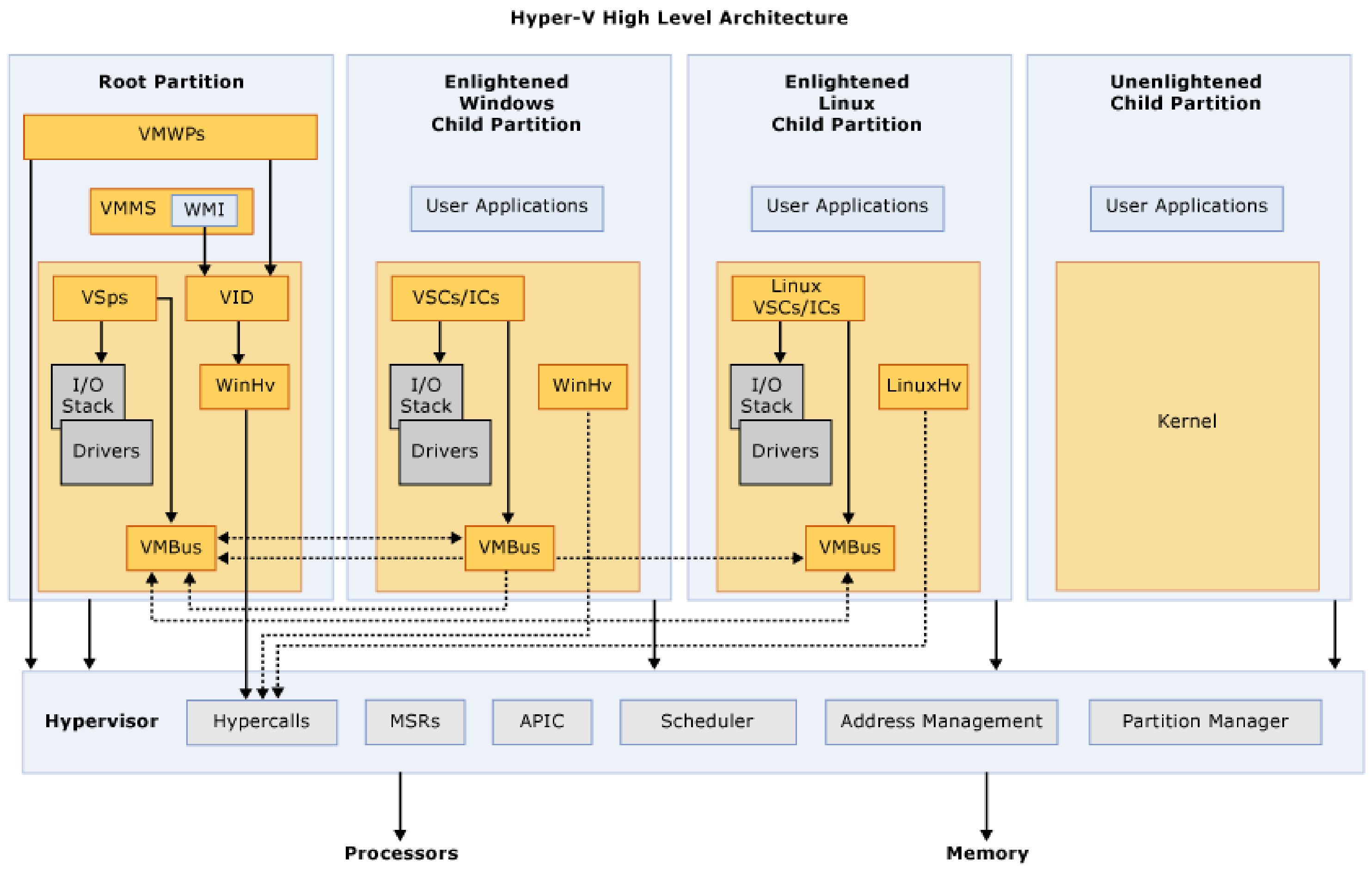


https://wiki.xen.org/wiki/Xen_Project_Software_Overview

Xen Hypervisor: Component

- **The Xen Hypervisor**
  - A thin software layer runs directly on the hardware and is
  - Responsible for managing CPU, memory, and interrupts
- **The Control Domain (or Domain 0)** is
  - A specialized Virtual Machine for handling I/O and for interacting with the other Virtual Machines.
  - It also exposes a control interface to the outside world, through which the system is controlled.
- **Guest Domains/Virtual Machines**
  - The VM allocated to users
- **Toolstack and Console**:
  - Admin interface for creating, destroying and configuring guest domains

Example Hypervisor: Hyper-V



https://docs.microsoft.com/en-us/biztalk/technical-guides/appendix-b-hyper-v-architecture-and-feature-overview

Hyper-V Components

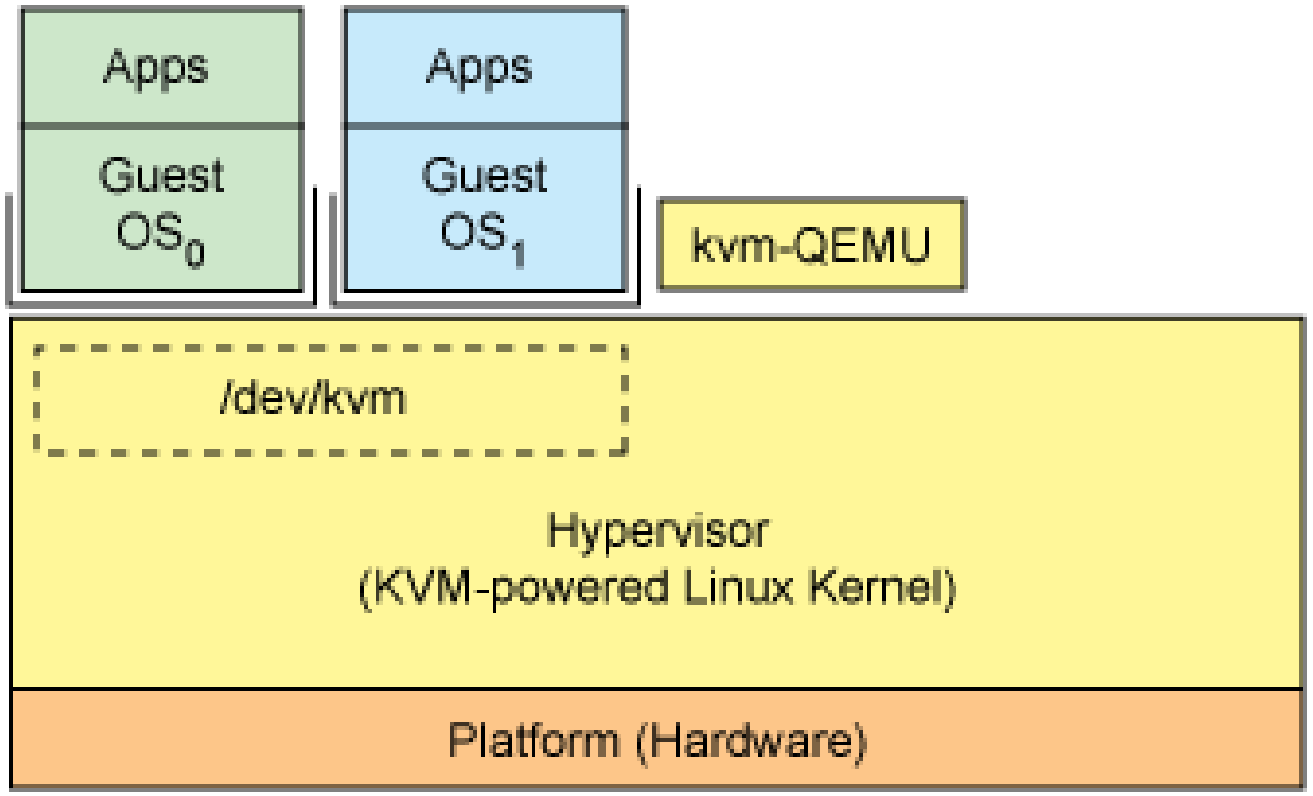
- **Hypervisor**
  - A layer of software that sits on top of the hardware
  - Its primary job is to provide isolated execution environments called partitions. The hypervisor controls and arbitrates access to the underlying hardware.
- **Partition**
  - The virtual machine running on hypervisor
- **Root Partition**
  - Similar to Xen’s domain 0, which manages I/O and communicate with other partitions
- **Child Partition**
  - All other guest VMs

Example Hypervisor: KVM

- Hypervisor performs lots of traditional operating system tasks
- Kernel-based Virtual Machine (KVM) development was started mid 2006 at Qumranet, which was acquired by Red Hat in 2008.
- It turns Linux kernel into a hypervisor with the help of a few modules: **KVM**, **QEMU** and **libvert**
- The development started soon after major chip manufacturers introduced hardware support for virtualization at processor level
    - Intel introduced VT-x in November 2005
    - AMD introduced AMD-V in May 2006
- Both Google Cloud Platform and AWS use KMV based virtualization

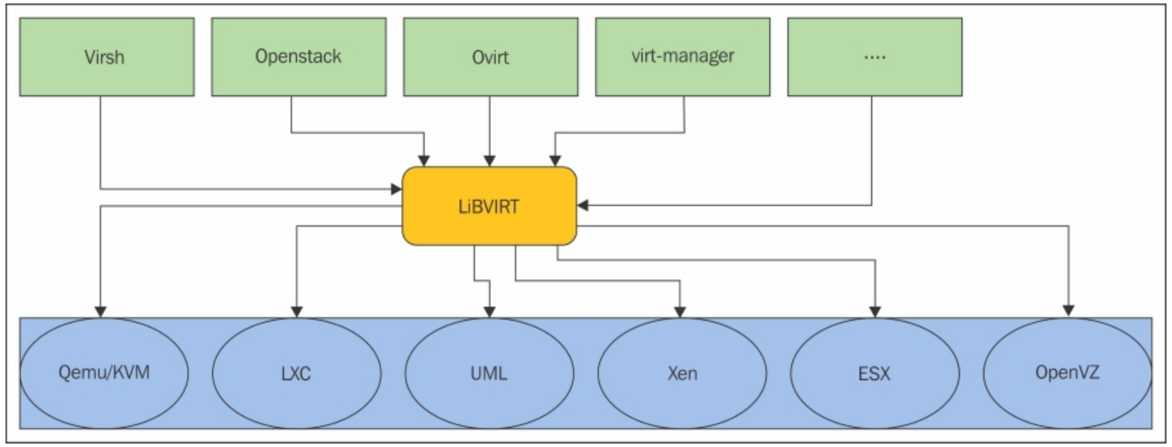
KVM architecture

- **KVM** turns a Linux kernel into a hypervisor
  - Several hardware-aware versions
  - When a new operating system is booted on KVM, it becomes a *process* of the host operating system and therefore schedulable like any other process. But it is in executed in different mode to regular OS



https://developer.ibm.com/tutorials/l-hypervisor/

QEMU and Libvert

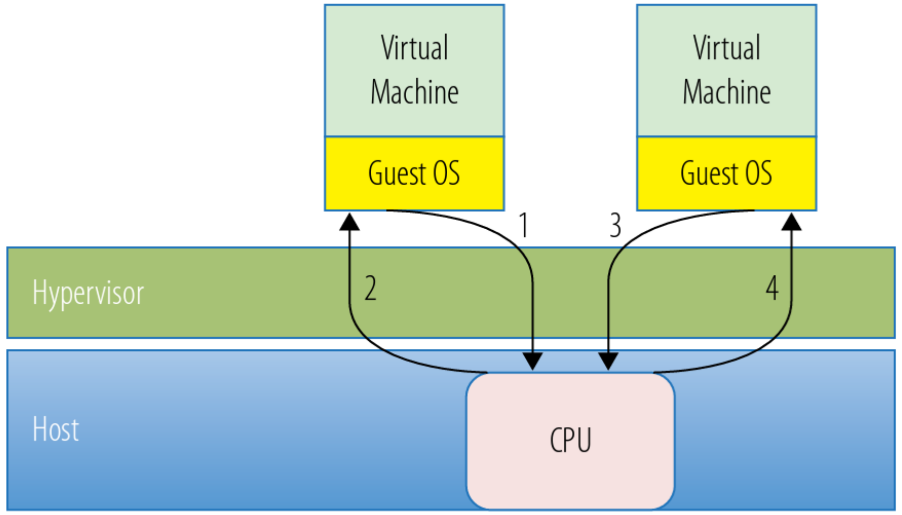


- `QEMU` (Quick Emulator)
  - A generic emulator that virtualizes I/O devices as well
- `Libvert` provides a common management layer to manage virtual machines running on hypervisors

VM Resource Management

Managing CPUs for a VM

- Physical CPU cores are time shared by virtual CPUs(vCPU) in server virtualization
- The dynamic scheduling of vCPUs on physical CPU is managed by hypervisor following customized scheduler algorithm
    - E.g. credit scheduler of Xen



Multiprocessors, cores and hyperthreading

- A host server may be configured with multiple processor chips, each with multiple core
- The core may support hyperthreading to have two or more logic cores for one physical core

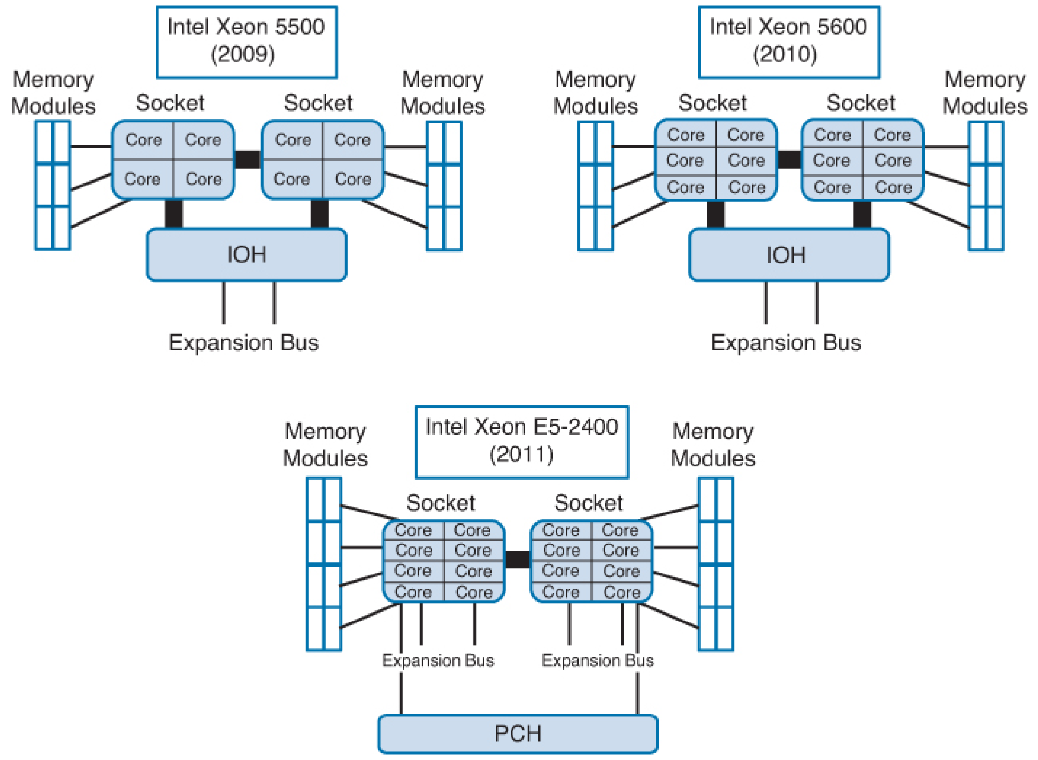


Figure 13.2 in Data Center Virtualization Fundamentals

VM with multiple vCPUs

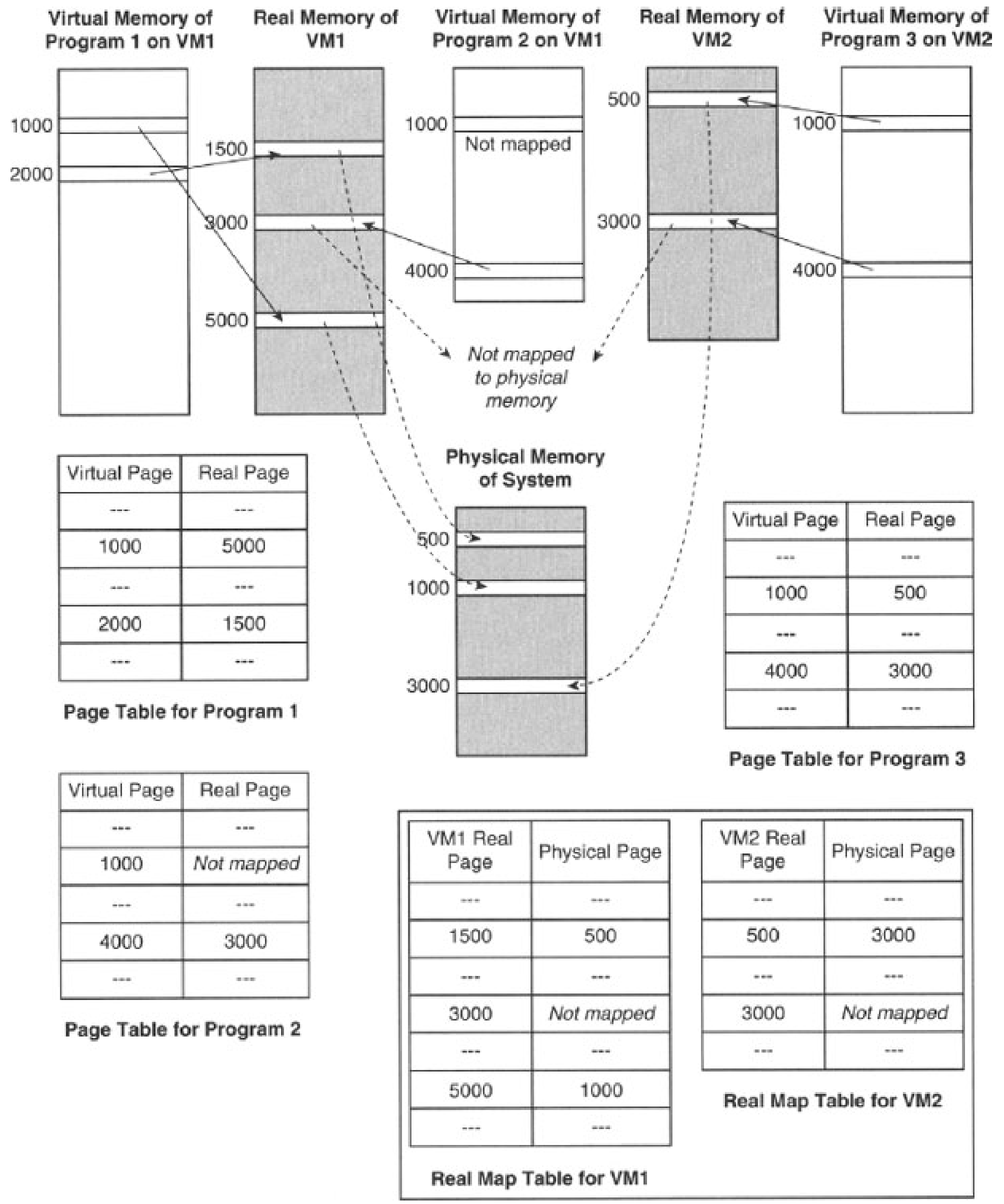
- vCPU count of VM
  - Means the maximum number of threads that the VM can run at any given moment.
- vCPU and physical CPU(pCPU)
  - A VM can run on any and all of the host CPUs over a period of time
    - Some hypervisor may support pinning
  - For a virtual machine with multiple vCPUs, the hypervisor will need to schedule each vCPU on a different pCPU (logic core also counts)
    - The maximum number of vCPU you can assign to a single VM is bound by the hardware capacity e.g. maximum number of cores
    - A VM with more vCPUs may have longer waiting time to get scheduled
  - Most applications are not designed to run on multiple threads
    - Configure a VM with too many vCPUs may not have any benefit
    - VM with 2 vCPU is a very typical setting

Virtualizing Memory Management

- Memory Management
  - OS is designed to manage the mapping of **virtual memory** and **physical memory** through TLB and page tables
  - Guest OS usually gets a dedicated portion of the physical memory to ensure strong isolation
  - Guest OS does not know other part of the memory and should not interfere other VM’s memory
  - It should not even know which part of the physical memory it is using
  - Only VMM has access to the physical memory and can manage the mapping

Memory Mapping in Guest OS

- Two level mapping/translation


Virtual Memory of Program 1 on VM1
Real Memory of VM1
Virtual Memory of Program 2 on VM1
Real Memory of VM2
Virtual Memory of Program 3 on VM2
1000
2000
1500
3000
5000
1000
Not mapped
4000
500
3000
1000
4000
Not mapped to physical memory
Physical Memory of System
500
1000
3000

Virtual Page	Real Page
---	---
1000	5000
---	---
2000	1500
---	---

Page Table for Program 1

Virtual Page	Real Page
---	---
1000	*Not mapped*
---	---
4000	3000
---	---

Page Table for Program 2

Virtual Page	Real Page
---	---
1000	500
---	---
4000	3000
---	---

Page Table for Program 3

VM1 Real Page	Physical Page
---	---
1500	500
---	---
3000	*Not mapped*
---	---
5000	1000
---	---

Real Map Table for VM1

VM2 Real Page	Physical Page
---	---
500	3000
---	---
3000	*Not mapped*
---	---

Real Map Table for VM2

Virtualizing I/O

- I/O refers to all input/output devices
    - Disk, network, printer, monitor, mouse
- On the one hand there are many devices with different ways of controlling them
- On the other hand, operating systems have efficient ways to deal with I/O by providing standard interface and by loading various drivers.

Disk and Network

- Disk is a partitioned device
  - Each VM can get a partition of it exposed as virtual disk
  - External disk drivers can be mounted easily
- Network adapter is a shared device
  - The physical adapter can be time shared by each VM
  - A virtual network adapter is presented to the VM
- The request are passed to and managed by VMM

I/O pathway

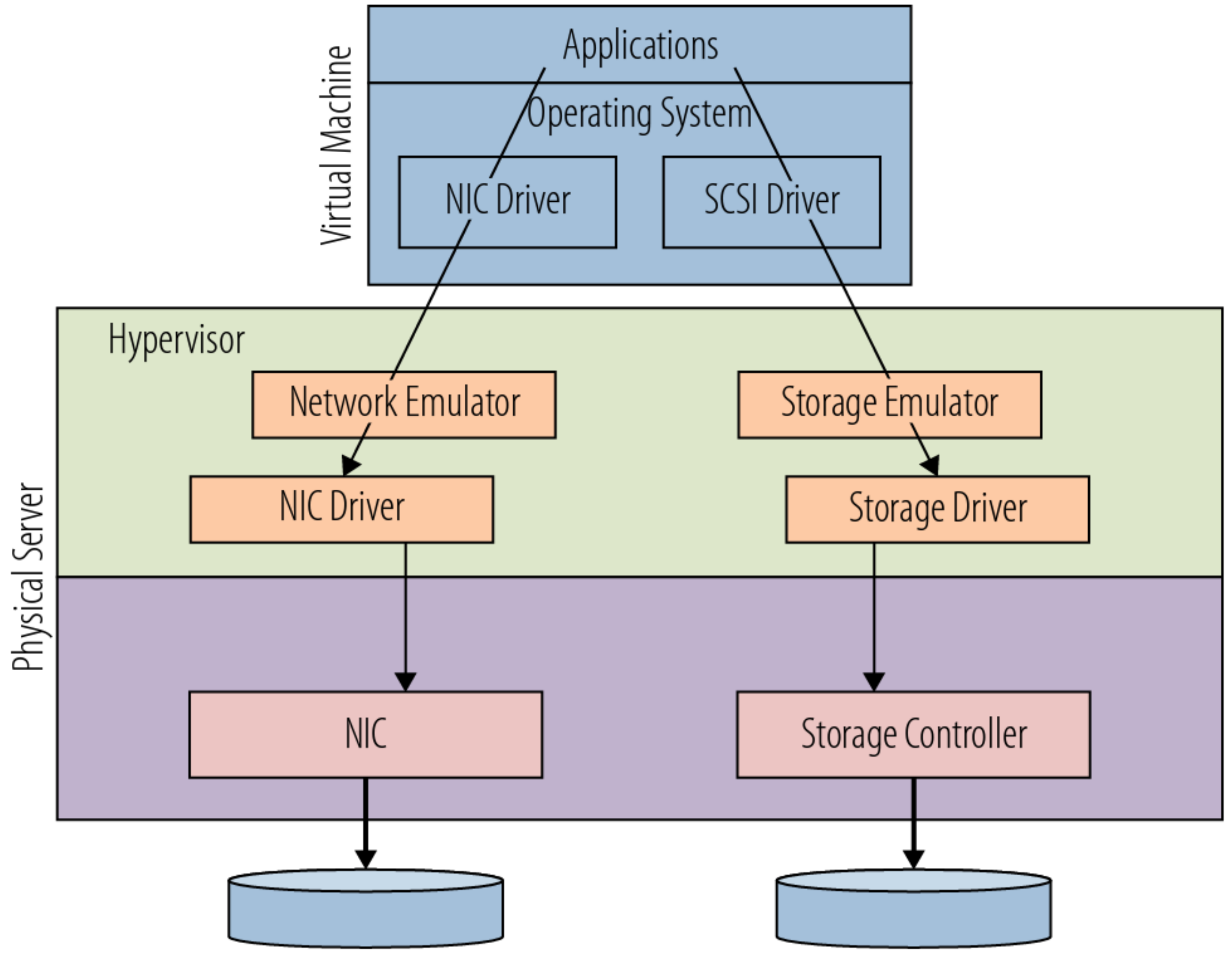


Figure 9.1 in Virtualization Essentials

I/O pathway in Xen

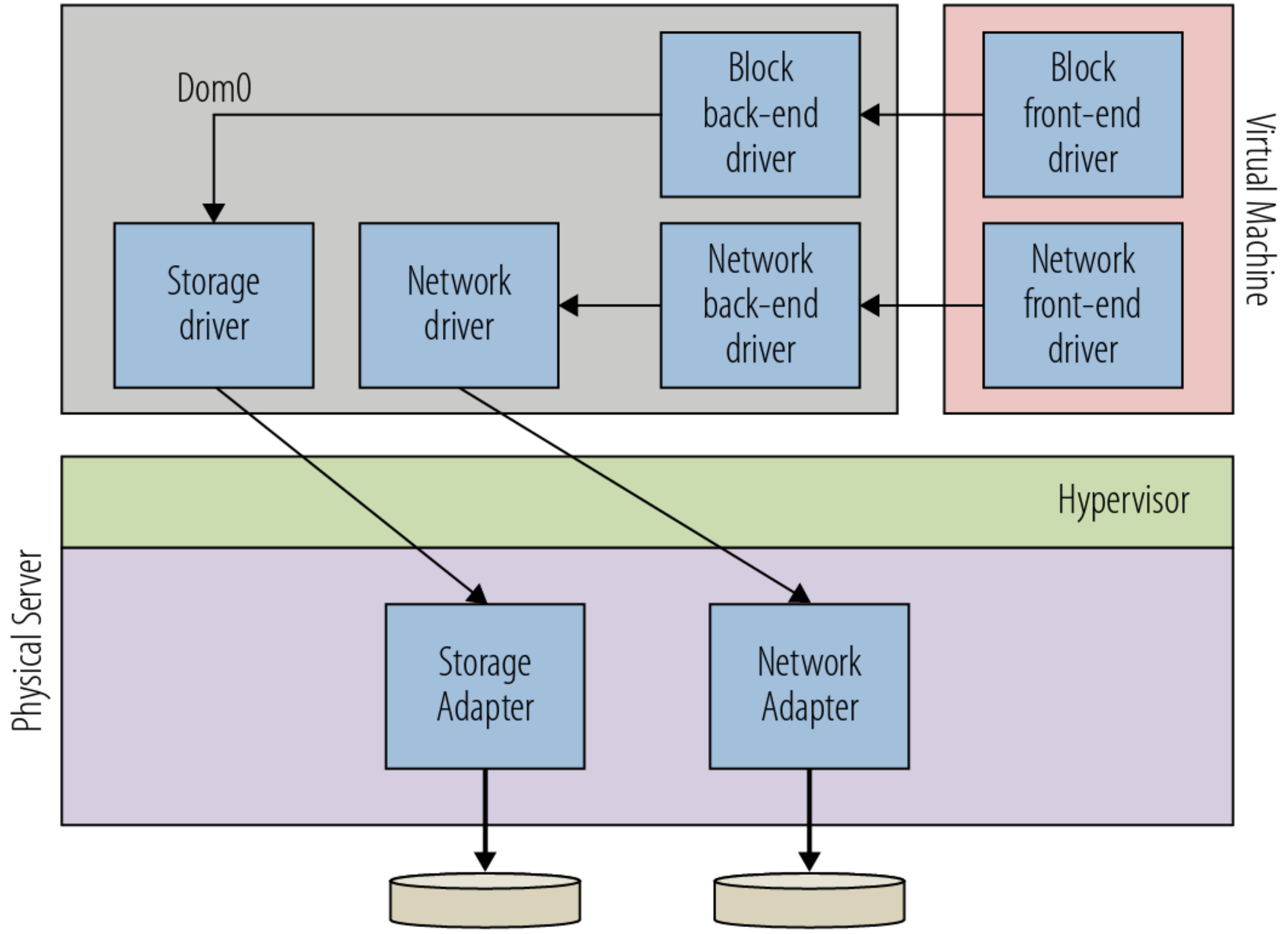


Figure 9.2 in Virtualization Essentials

Amazon EC2 Overview

Amazon EC2

Amazon Elastic Compute Cloud (Amazon EC2)

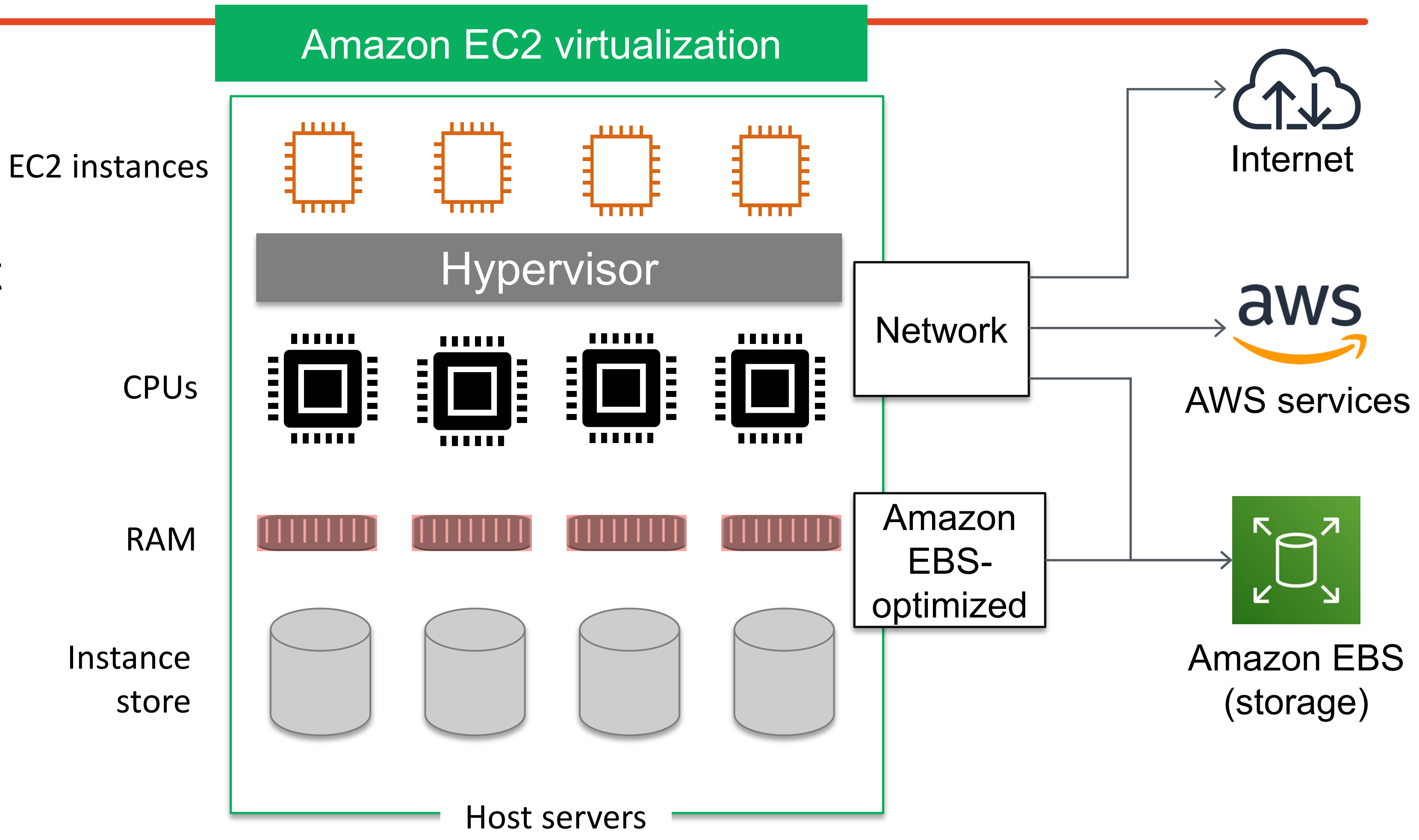
- Amazon EC2 provides resizable compute capacity in the cloud.
  - Provides virtual machines (servers)
  - Provisions servers in minutes
  - Can automatically scale capacity up or down as needed
  - Enables you to pay only for the capacity that you use

Most slides in this section are adopted from AWS Academy Cloud Foundations Course Module 6

EC2 instances

An EC2 instance is a virtual machine that runs on a physical host.

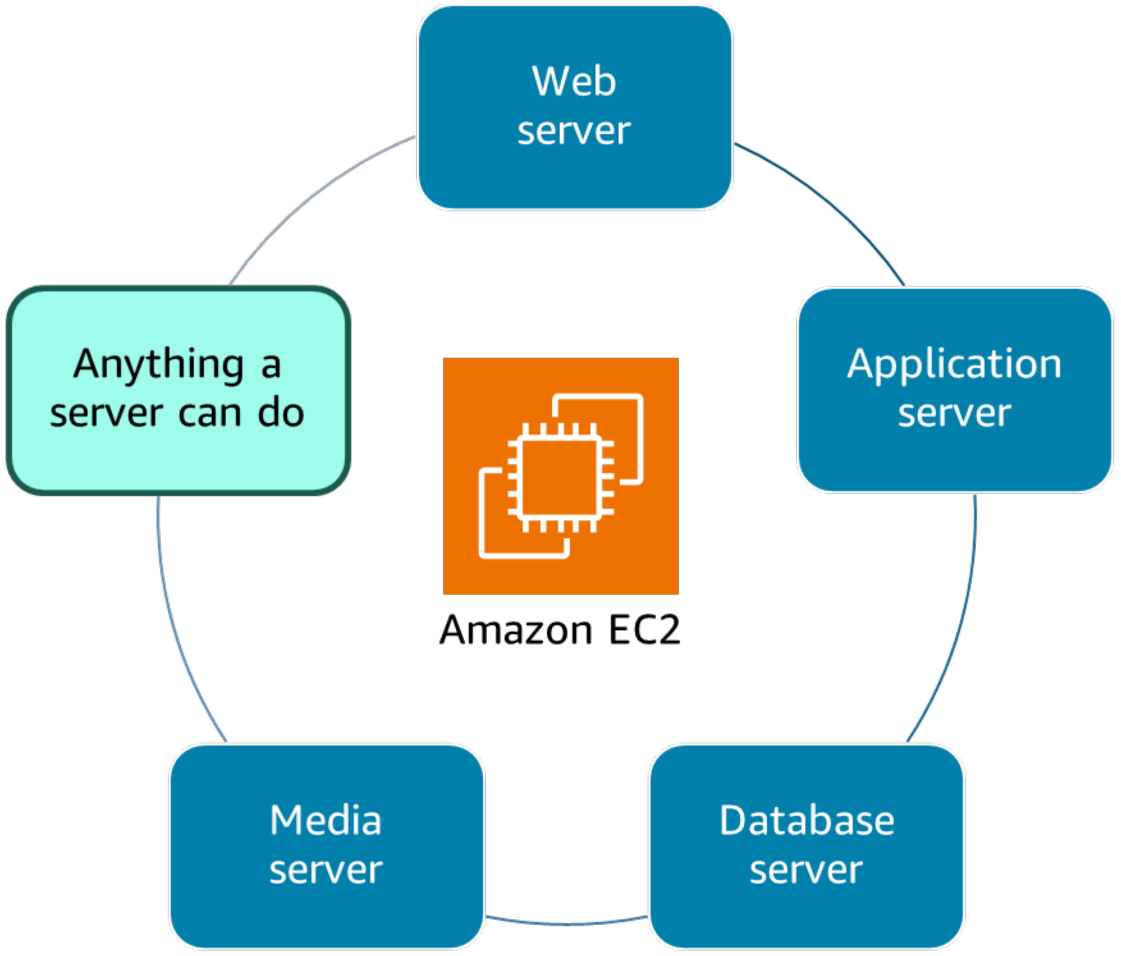
- You can choose different configurations of CPU and memory capacity
- Supports different storage options
  - Instance store
  - Amazon Elastic Block Store (Amazon EBS)
- Provides network connectivity



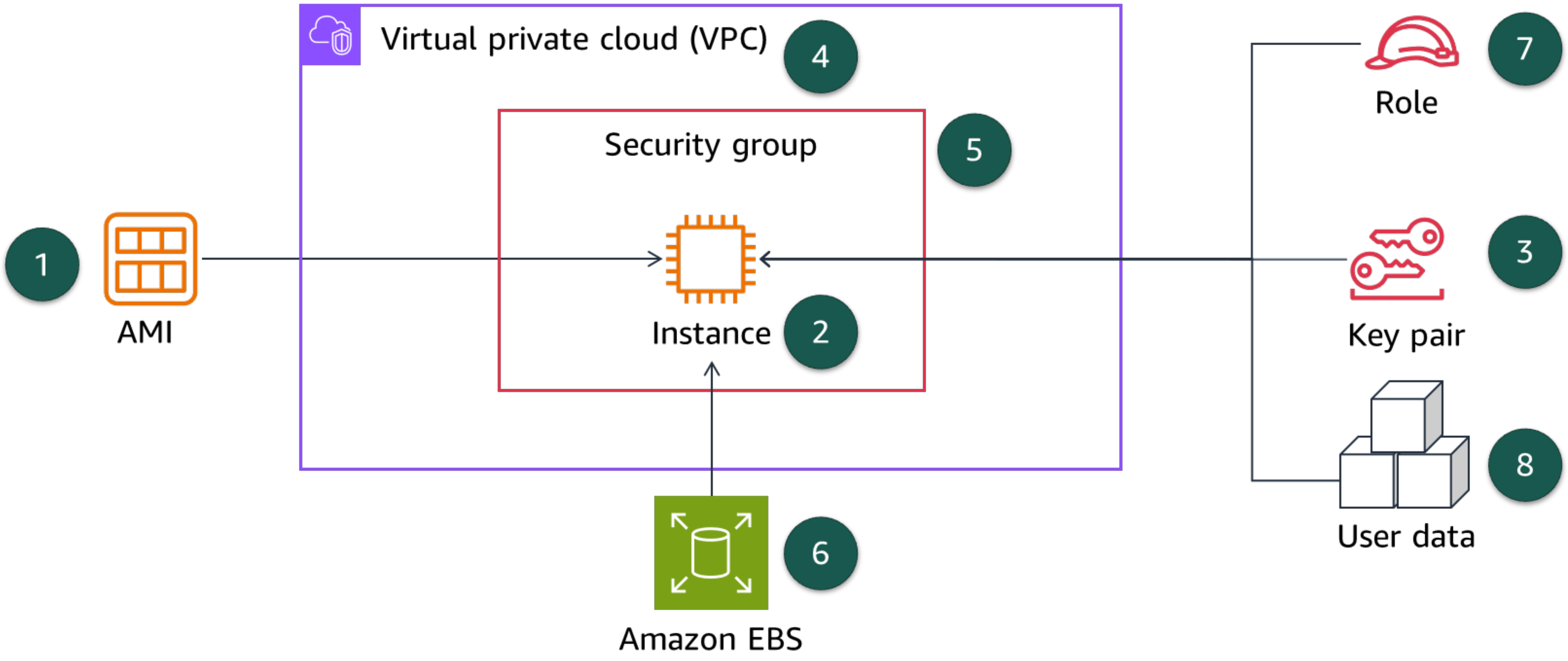
Amazon EC2 use cases

Use Amazon EC2 when you need the following:

- Complete control of your computing resources, including operating system and processor type.
- Options for optimizing your compute costs
  - On-Demand Instances, Reserved Instances, and Spot Instances
  - Savings Plans
- Ability to run any type of workload
  - Simple websites
  - Enterprise applications
  - Generative artificial intelligence (generative AI) applications



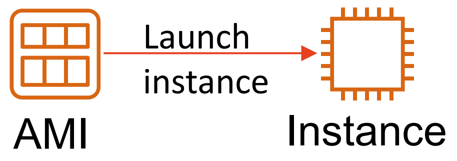
Steps for provisioning an EC2 instance



1. Select an AMI

Choices made using the Launch Instance Wizard:

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



- 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*

AMI benefits

Repeatability

An AMI can be used repeatedly to launch instances with efficiency and precision.

Reusability

Instances launched from the same AMI are identically configured.

Recoverability

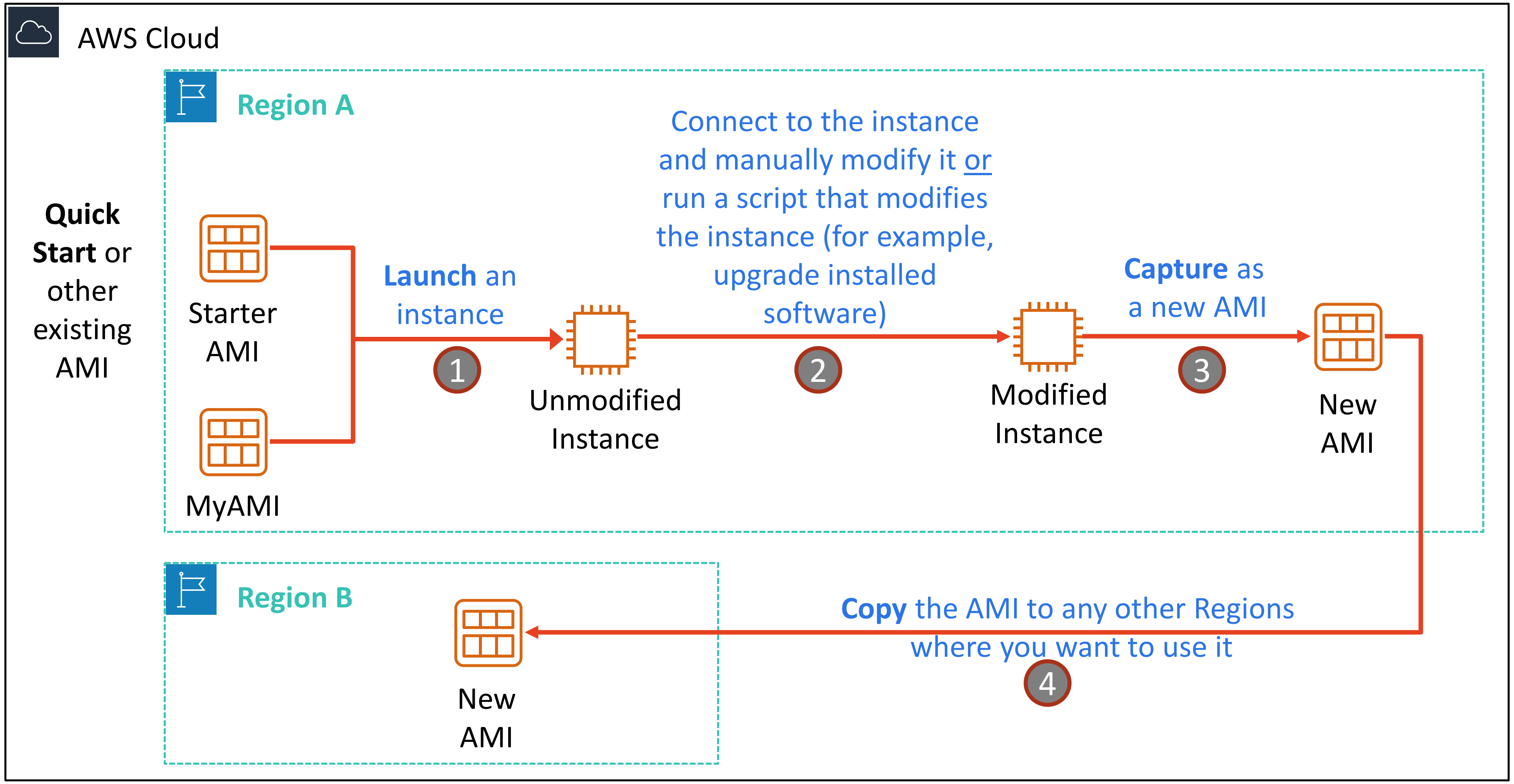
You can create an AMI from a configured instance as a restorable backup.

You can replace a failed instance by launching a new instance from the same AMI.

Creating a new AMI: Example

AMI details

(Optional) Import a virtual machine


AWS Cloud
Region A
Quick Start or other existing AMI
Starter AMI
MyAMI
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. **Key pair**
4. **Network settings**
5. **Security group**
6. **Storage options**
7. **IAM role**
8. **User data**
9. **Tags**

- 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 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

	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.

Choosing an instance type

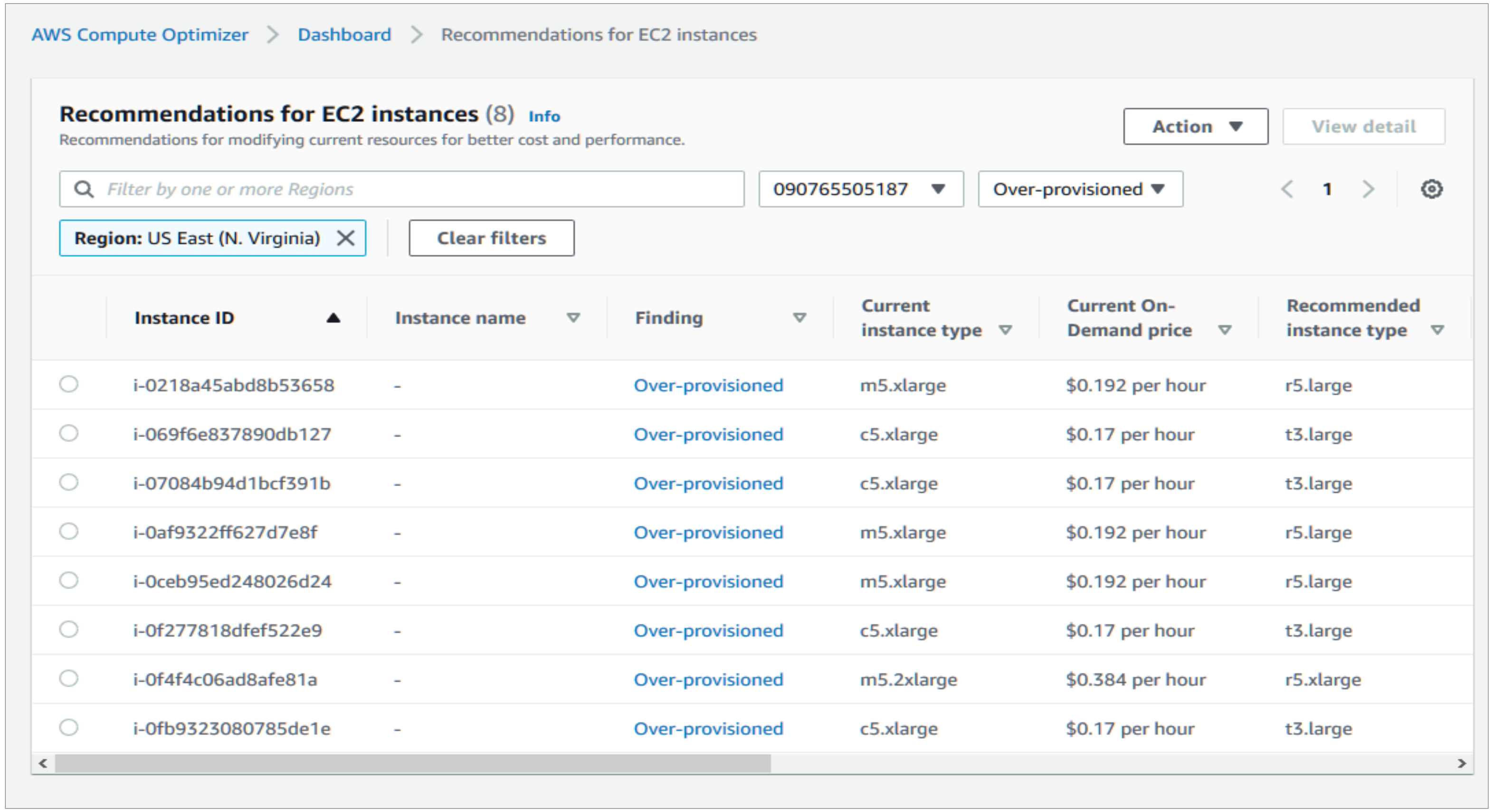
- With over 270 available instance types, how do you choose the right type?
  - Consider both performance requirements and cost requirements.
  - Use available resources to get recommended options.

Task	Solution
Creating a new instance	• In the EC2 console, use the Instance Types page to filter by characteristics that you choose. • Recommendation: The latest generation in an instance family typically has a better price-to-performance ratio.
Optimizing an existing instance	• You can get recommendations for optimizing the instance type by using the AWS Compute Optimizer. • You can evaluate recommendations and modify the instance accordingly.

AWS Compute Optimizer

AWS Compute Optimizer

- Recommends optimal instance type, instance size, and Auto Scaling group configuration
- Analyzes workload patterns and makes recommendations
- Classifies instance findings as Under-provisioned, Over-provisioned, Optimized, or None



AWS Compute Optimizer > Dashboard > Recommendations for EC2 instances

Recommendations for EC2 instances (8) Info

Recommendations for modifying current resources for better cost and performance.

Filter by one or more Regions | 090765505187 | Over-provisioned

Region: US East (N. Virginia)

Instance ID	Instance name	Finding	Current instance type	Current On-Demand price	Recommended instance type
i-0218a45abd8b53658	-	Over-provisioned	m5.xlarge	$0.192 per hour	r5.large
i-069f6e837890db127	-	Over-provisioned	c5.xlarge	$0.17 per hour	t3.large
i-07084b94d1bcf391b	-	Over-provisioned	c5.xlarge	$0.17 per hour	t3.large
i-0af9322ff627d7e8f	-	Over-provisioned	m5.xlarge	$0.192 per hour	r5.large
i-0ceb95ed248026d24	-	Over-provisioned	m5.xlarge	$0.192 per hour	r5.large
i-0f277818dfef522e9	-	Over-provisioned	c5.xlarge	$0.17 per hour	t3.large
i-0f4f4c06ad8afe81a	-	Over-provisioned	m5.2xlarge	$0.384 per hour	r5.xlarge
i-0fb9323080785de1e	-	Over-provisioned	c5.xlarge	$0.17 per hour	t3.large

3. Identify or create the key pair

Choices made by using the Launch Instance Wizard:

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

- 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

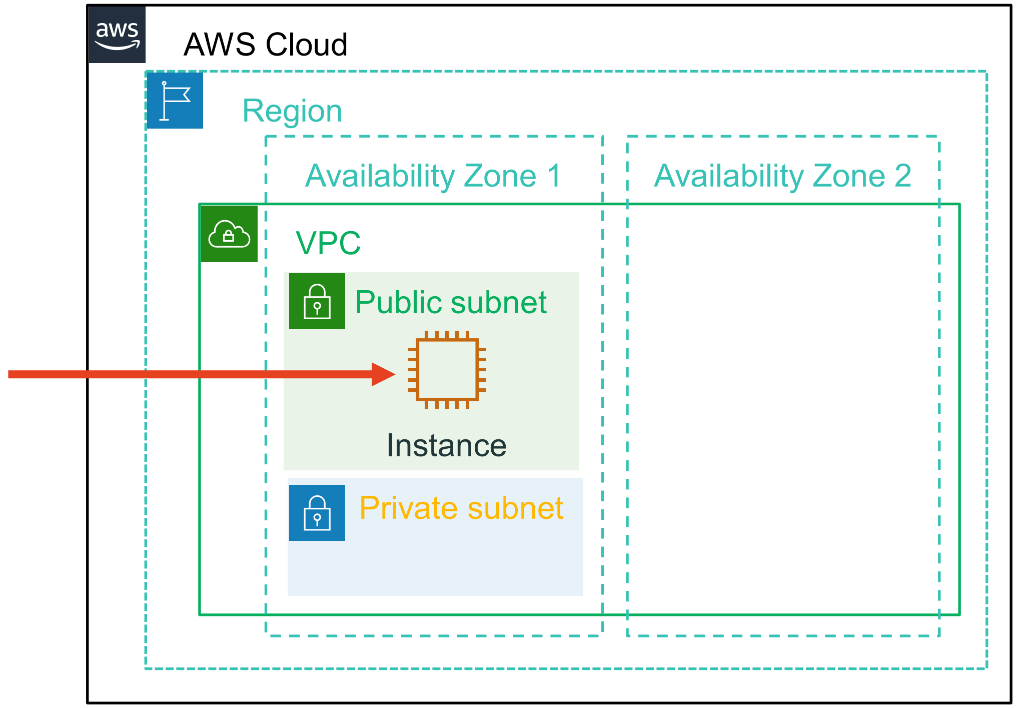
4. Specify network settings

Choices made by using the Launch Instance Wizard:

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

- 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



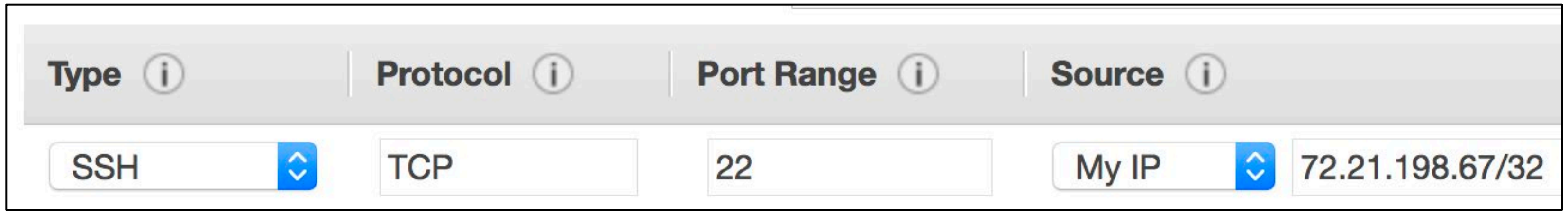
5. Security group settings

Choices made by using the Launch Instance Wizard:

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

- 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

6. Specify storage

Choices made by using the Launch Instance Wizard:

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

- 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 overview

AWS EC2 storage resource	Root Volume	Data volumes for a single instance	Data volumes for data that is accessible from multiple Linux instances	Data volumes for data that is accessible from multiple Windows instances
Amazon EBS (SSD-backed only)	Yes	Yes	No	No
Instance store	Yes	Yes	No	No
Amazon Elastic File System (Amazon EFS) [Linux]	No	No	Yes	No
Amazon FSx for Windows File Server	No	No	No	Yes

An EC2 instance will *always* have a root volume, and can *optionally* have one or more data volumes.

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)**.

Instance store

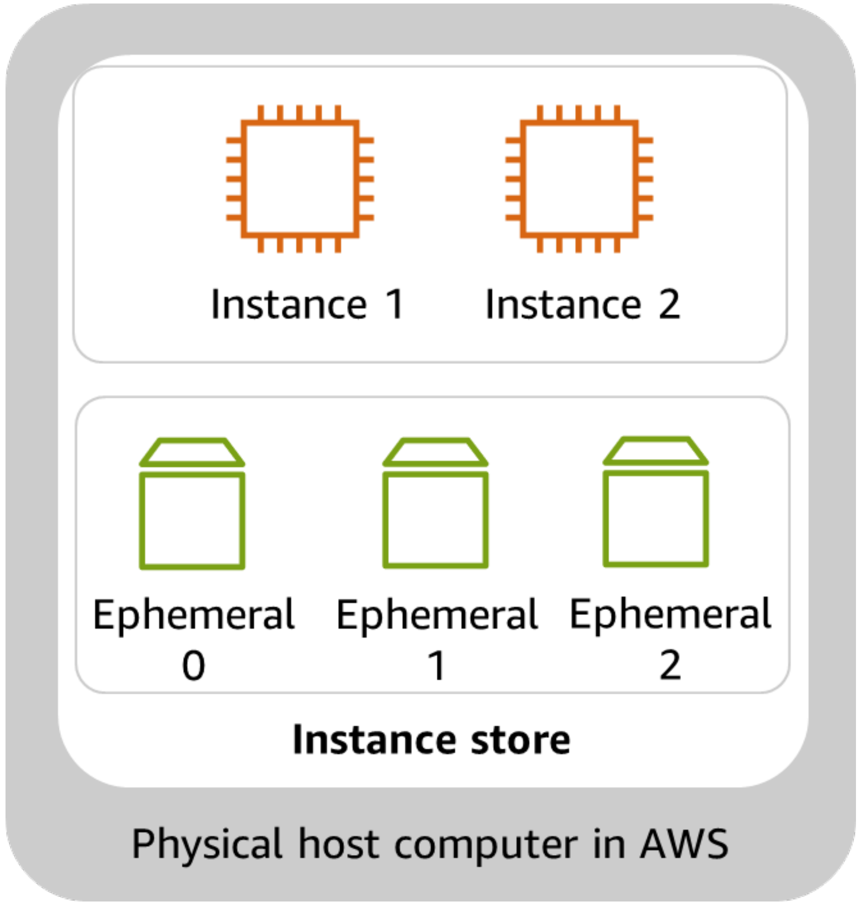
An instance store provides non-persistent storage to an instance. The data volume is stored on the same physical server where the instance runs.

Characteristics

- Temporary block-level storage
- Uses HDD or SSD
- Instance store data is lost when the instance is stopped or terminated.

Example use cases

- Buffers
- Cache
- Scratch data



Amazon EBS

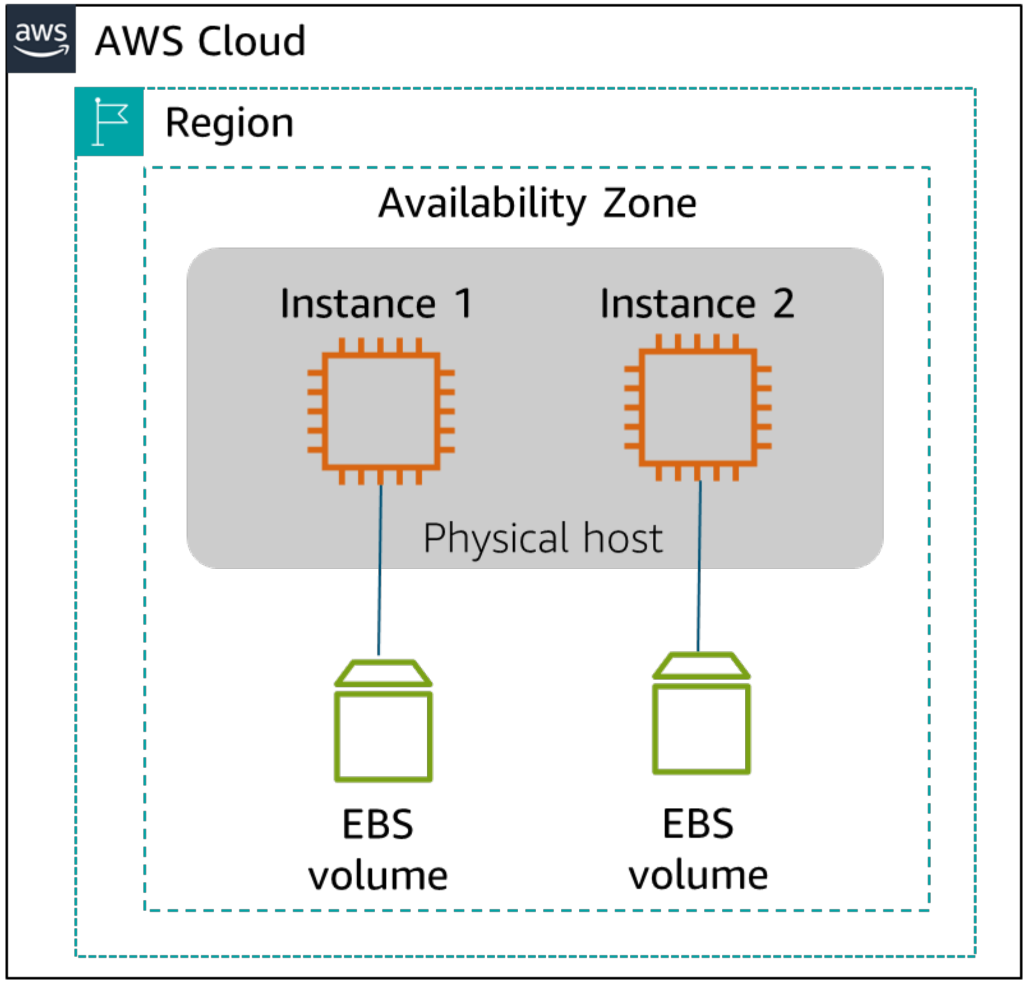
Amazon EBS volumes provide network-attached persistent storage to an EC2 instance.

Characteristics

- Is persistent block-level storage
- Can attach to any instance in the same Availability Zone
- Uses HDD or SSD
- Can be encrypted
- Supports snapshots that are persisted to S3
- Data persists independently from the life of the instance

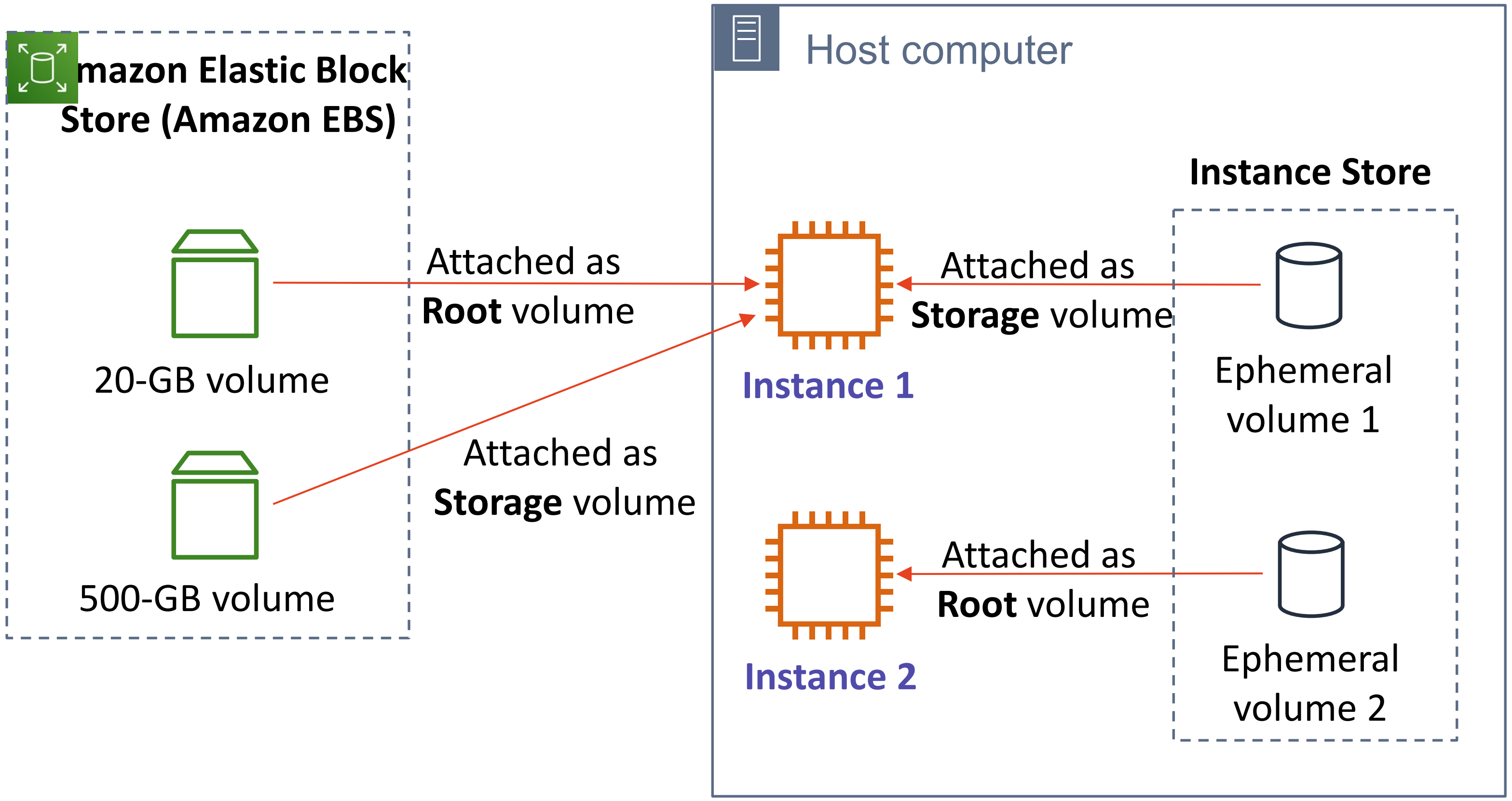
Example use cases

- Stand-alone database
- General application data storage



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)?



Shared file systems for EC2 instances

What if you have multiple instances that must use the same storage?

Not an option	Not ideal option	Best option	
Amazon EBS	Amazon S3	Amazon EFS (Linux)	Amazon FSx for Windows File Server (Windows)
Attaches only to one instance	Amazon S3: Is an option, but is not ideal	Amazon EFS *and* Amazon FSx for Windows File Server: Both satisfy the requirement	

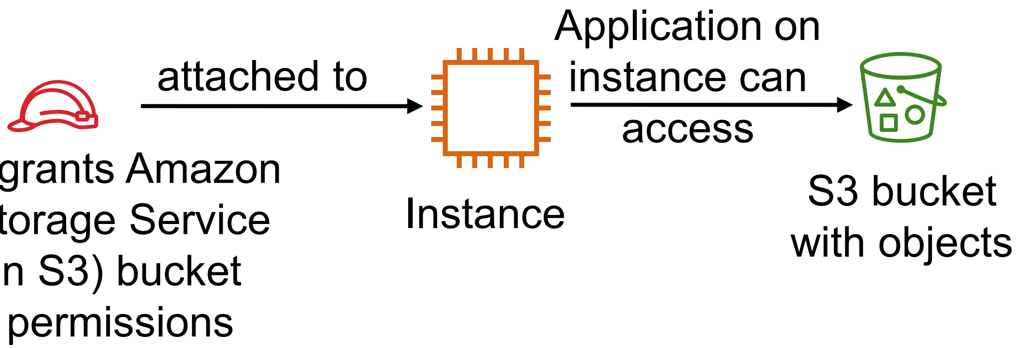
7. Attach IAM role (optional)

Choices made by using the Launch Instance Wizard:

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

- 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.

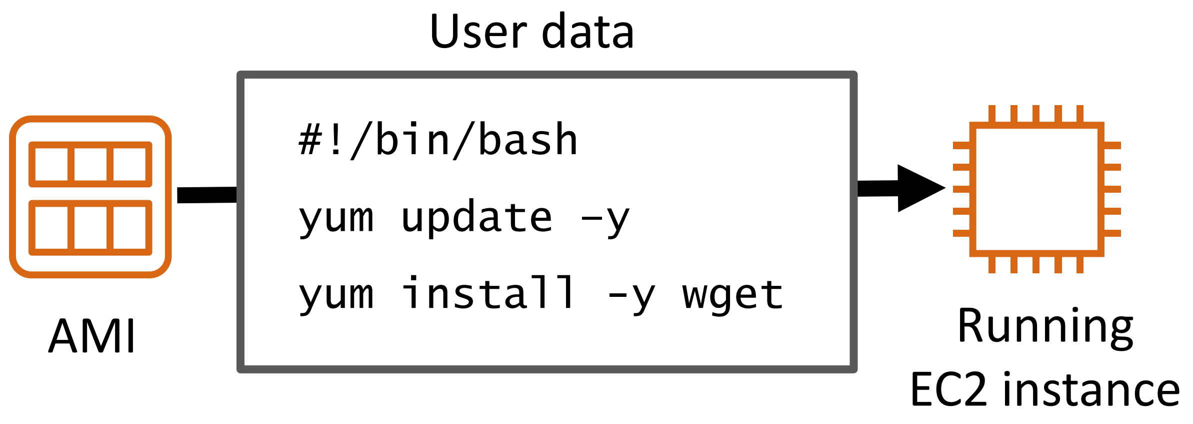
Example:


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

8. User data script (optional)

Choices made by using the Launch Instance Wizard:

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



```
#!/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

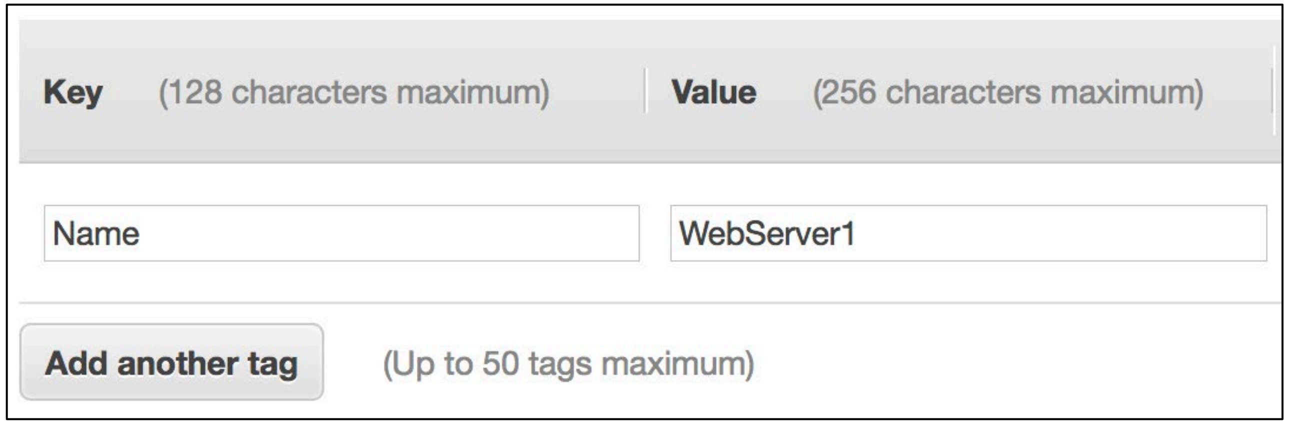
9. Add tags

Choices made by using the Launch Instance Wizard:

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

- 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.

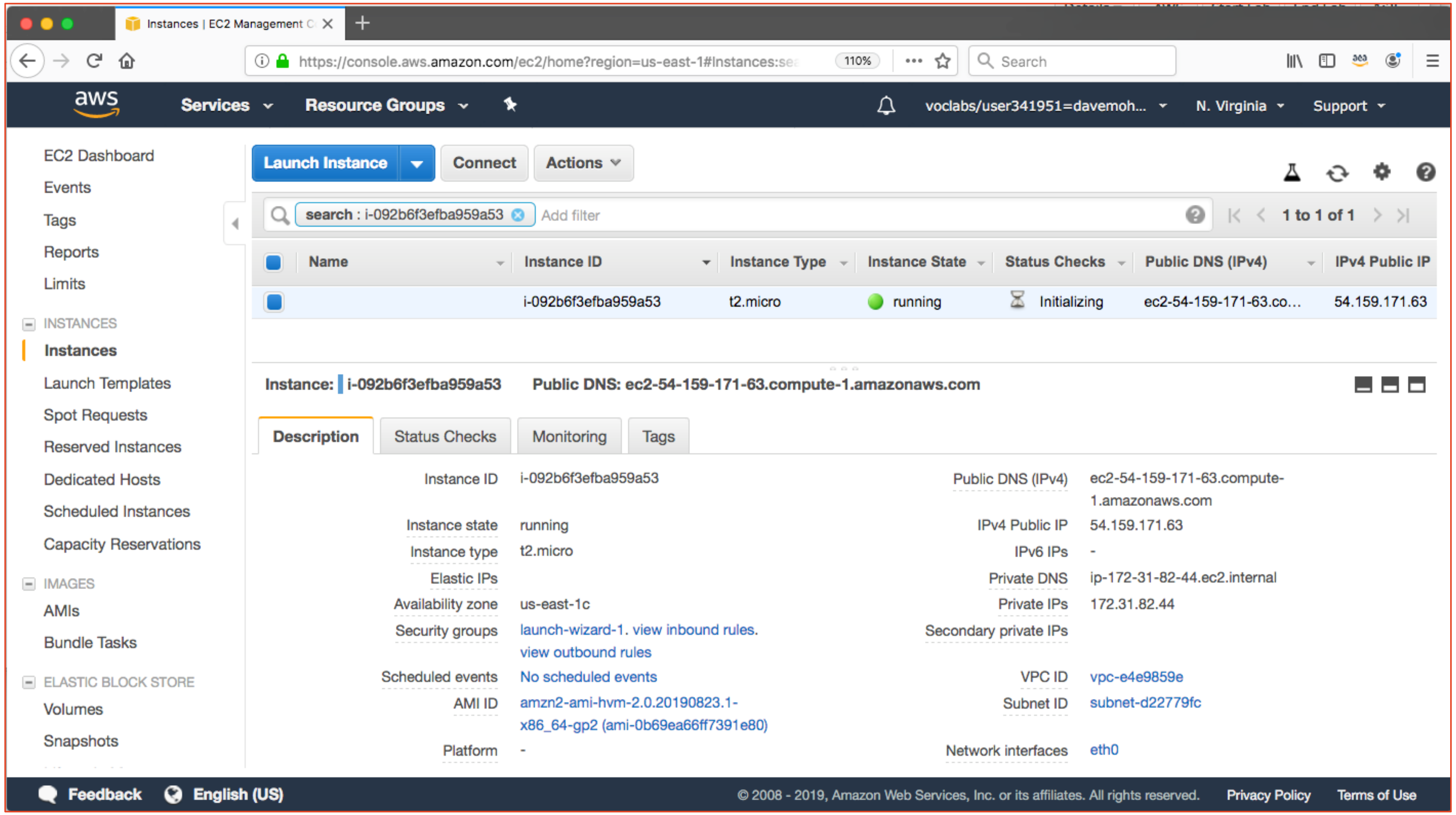
Example:



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

Add another tag (Up to 50 tags maximum)

Amazon EC2 console view of a running EC2 instance

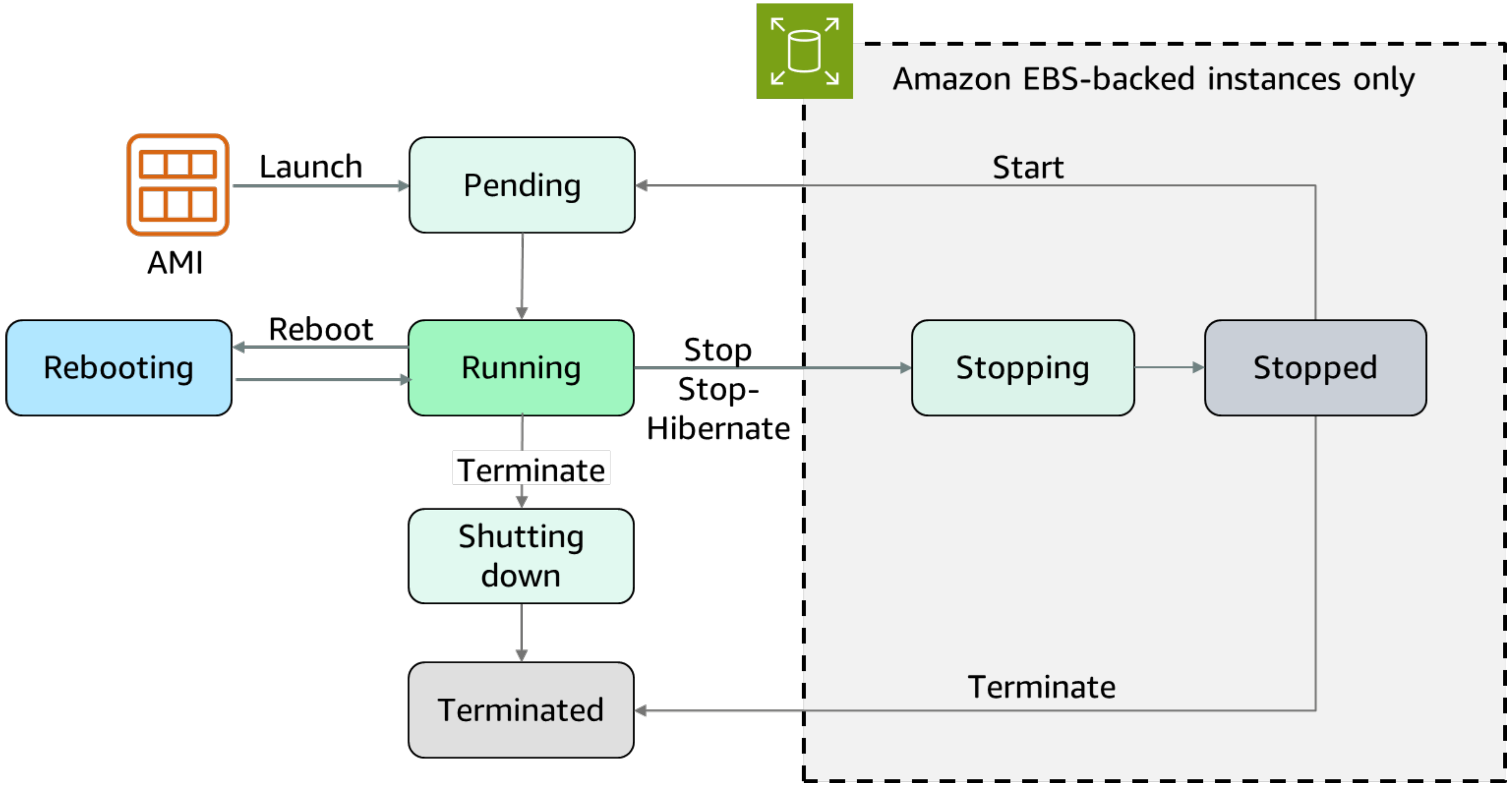


Name	Instance ID	Instance Type	Instance State	Status Checks	Public DNS (IPv4)	IPv4 Public IP
	i-092b6f3efba959a53	t2.micro	● running	Initializing	ec2-54-159-171-63.co...	54.159.171.63

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

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

Amazon EC2 instance lifecycle



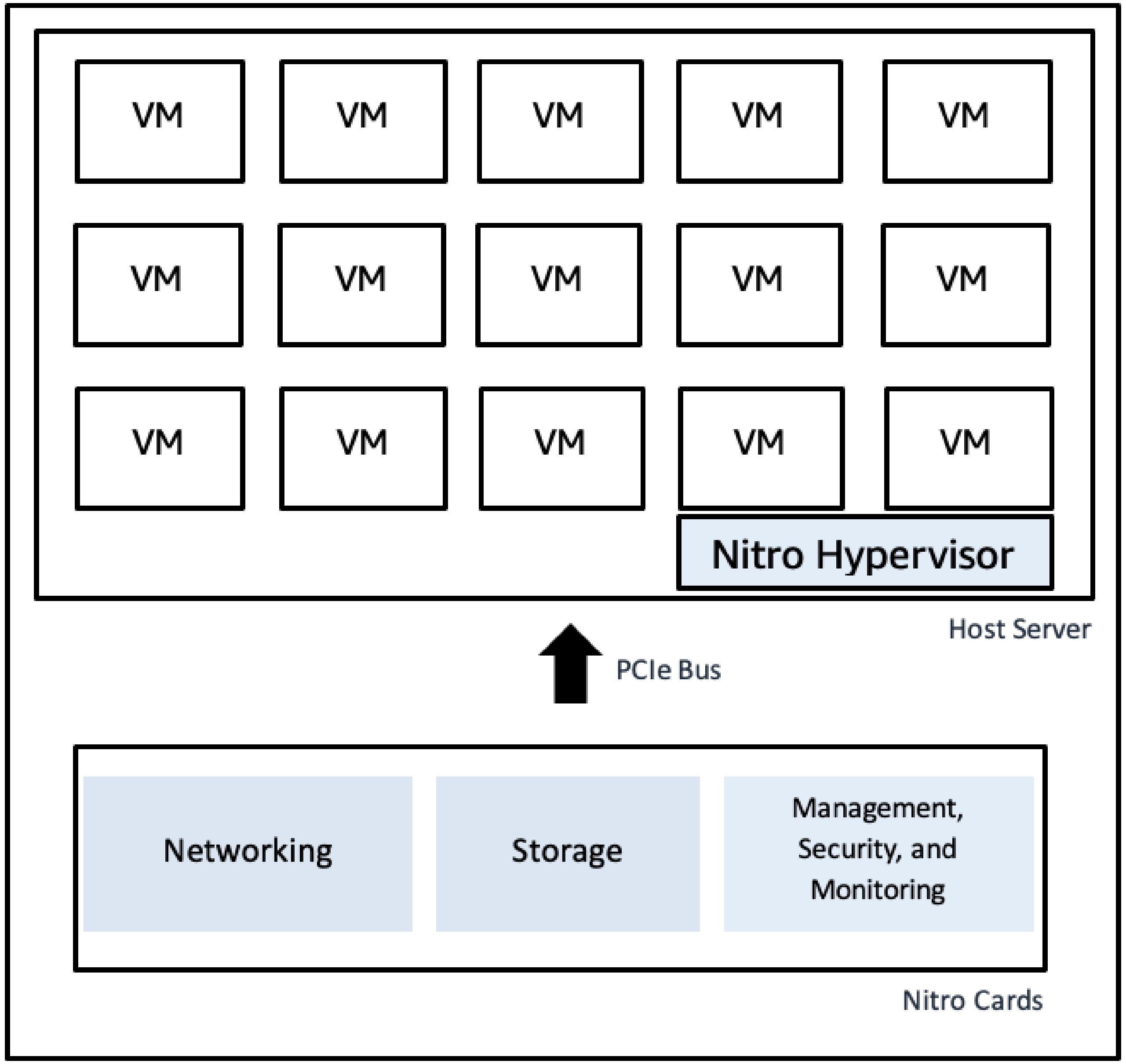
AWS Nitro System

Nitro System

- Nitro is the current virtualization system used in Amazon to provision EC2 and all related services
- AWS used Xen hypervisor in early days
- It moved to its own virtualization system Nitro in 2017
- Nitro itself is an example of micro services architecture
  - Very thin HVM based hypervisor mainly for partitioning and scheduling CPU and memory
  - Hardware support for all other resources, including monitoring and management, are built into Nitro cards
    - Domain 0 is no longer required

Nitro Architecture

CPUs and Memories on the main board



Nitro System

- hypervisor
  - CPU and memory
- one or more Nitro Cards
  - Networking
  - Storage
  - Management
  - Monitoring
  - Security
- Security Chips

Nitro card for networking

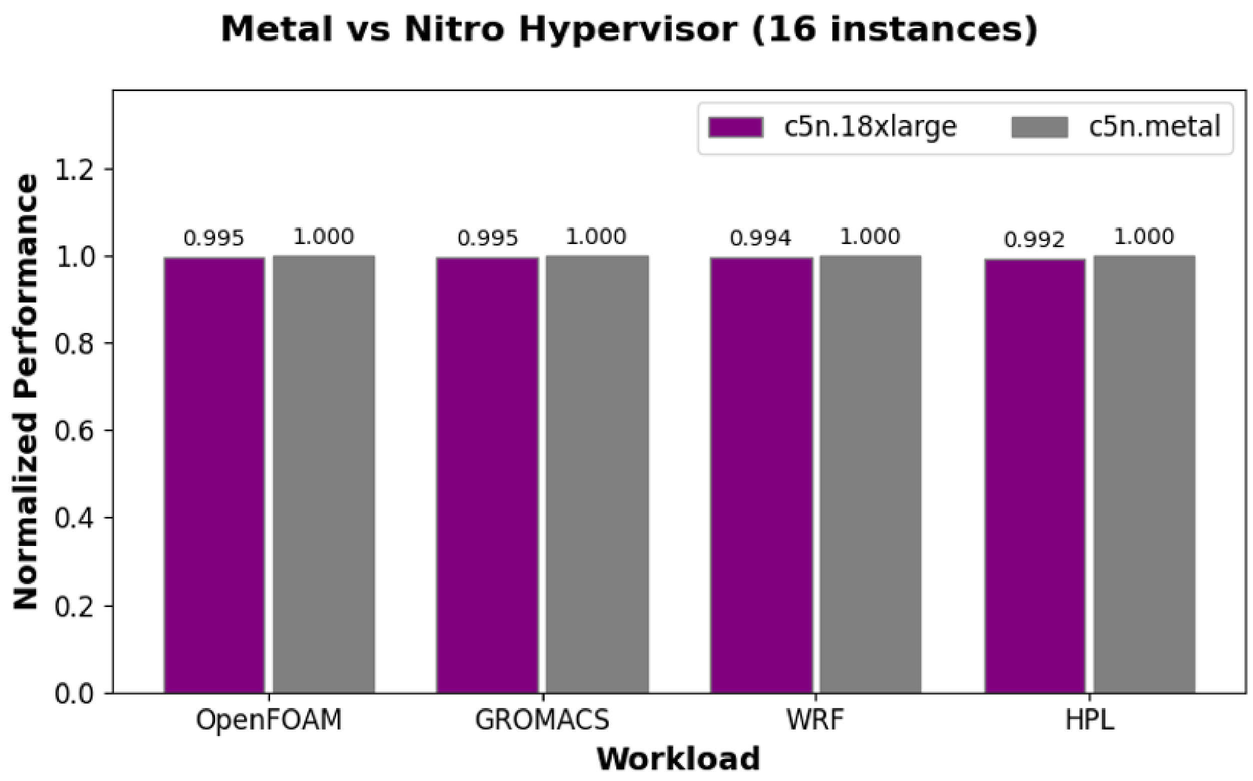
- Nitro card for VPC
- The network interface of EC2
- looks likes a network adapter
- Support AWS’s own Elastic Network Adapter (ENA) API
    - Driver independent of actual hardware capacity

Nitro card for storage

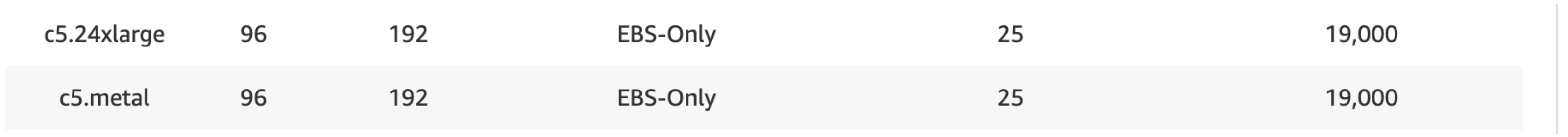
- Nitro card for EBS
  - NVMe controller
  - Encryption support
  - NVM to remote storage protocol
- Nitro card for instance storage
  - Instance storage has been on and off in AWS
  - Networking and storage evolved along different path
    - Offer local storage when I/O throughput is faster than network throughput
    - When network throughput is greater than storage throughput, offering network drive makes more sense
    - Local storage comes back with SSD and NVMe

AWS bare metal instance

- Bare metal instances do not use Nitro hypervisor
- They still uses Nitro cards to access VPC, EBS and other AWS services
- There is not much performance difference between hypervisor supported and bare metal instances
- Use instances backed by hypervisor in most cases



https://aws.amazon.com/blogs/hpc/bare-metal-performance-with-the-aws-nitro-system/



c5.24xlarge	96	192	EBS-Only	25	19,000
c5.metal	96	192	EBS-Only	25	19,000

Reference

- Gustavo A., A. Santana: **Data Center Virtualization Fundamentals: Understanding Techniques and Designs for Highly Efficient Data Centers with Cisco Nexus, UCS, MDS, and Beyond**
  - **Chapter 1, chapter 13**
- Matthew Portnoy: Virtualization Essentials, Second Edition
  - Various chapters
- Jim Smith, Ravi Nair , **Virtual Machines: Versatile Platforms for Systems and Processes,** Morgan Kaufmann; 1 edition (June 17, 2005)
  - Chapter 1
  - Chapter 8 System Virtual Machines
- *Xen and the Art of Virtualization*. Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, and Andrew Warfield. In Proceedings of the Nineteenth ACM Symposium on Operating systems principles (SOSP '03), 2003.
- AWS White Paper
  - https://docs.aws.amazon.com/whitepapers/latest/security-design-of-aws-nitro-system/security-design-of-aws-nitro-system.html