Kubernetes Essentials

FULL COURSE

VMs vs Containers

1.2

MARCOLENZO.EU

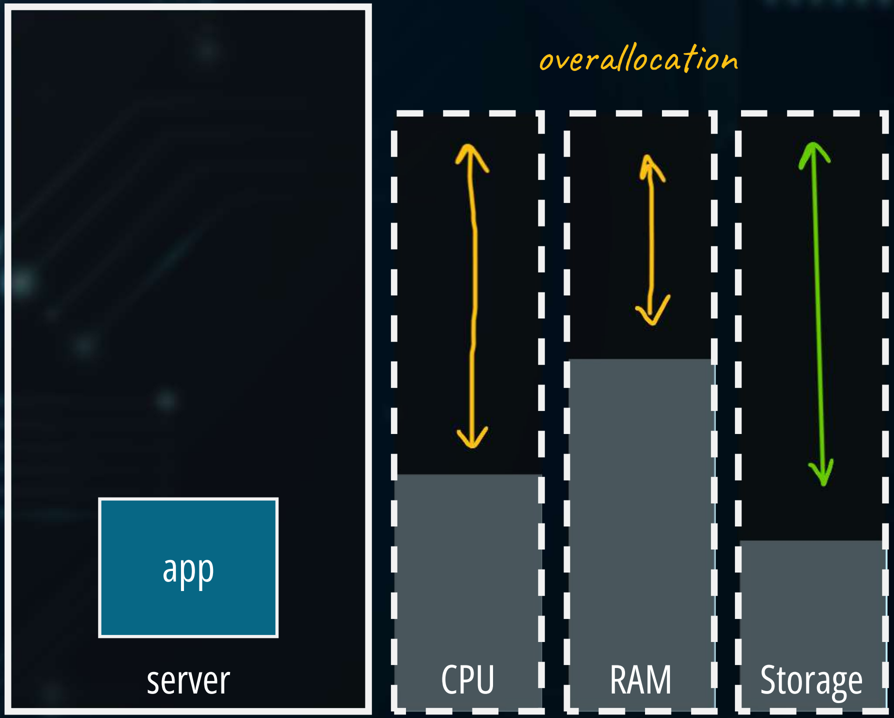
Unit 1.2

- **Theory**
  - Deployment challenges on bare metal
  - Virtual Machines and their limitations
  - Containers
- **Practical**
  - Docker Basics

Deployment Challenges on Bare Metal

MARCOLENZO.EU

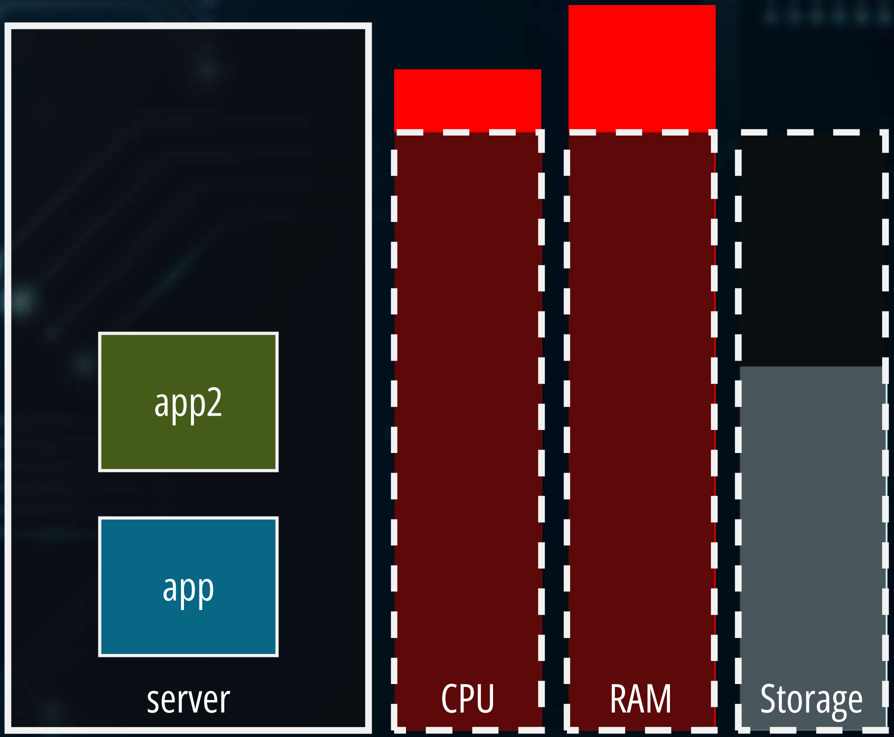
- Underutilization of resources



Deployment Challenges on Bare Metal

MARCOLENZO.EU

- Underutilization of resources
- Poor Isolation



Deployment Challenges on Bare Metal

MARCOLENZO.EU

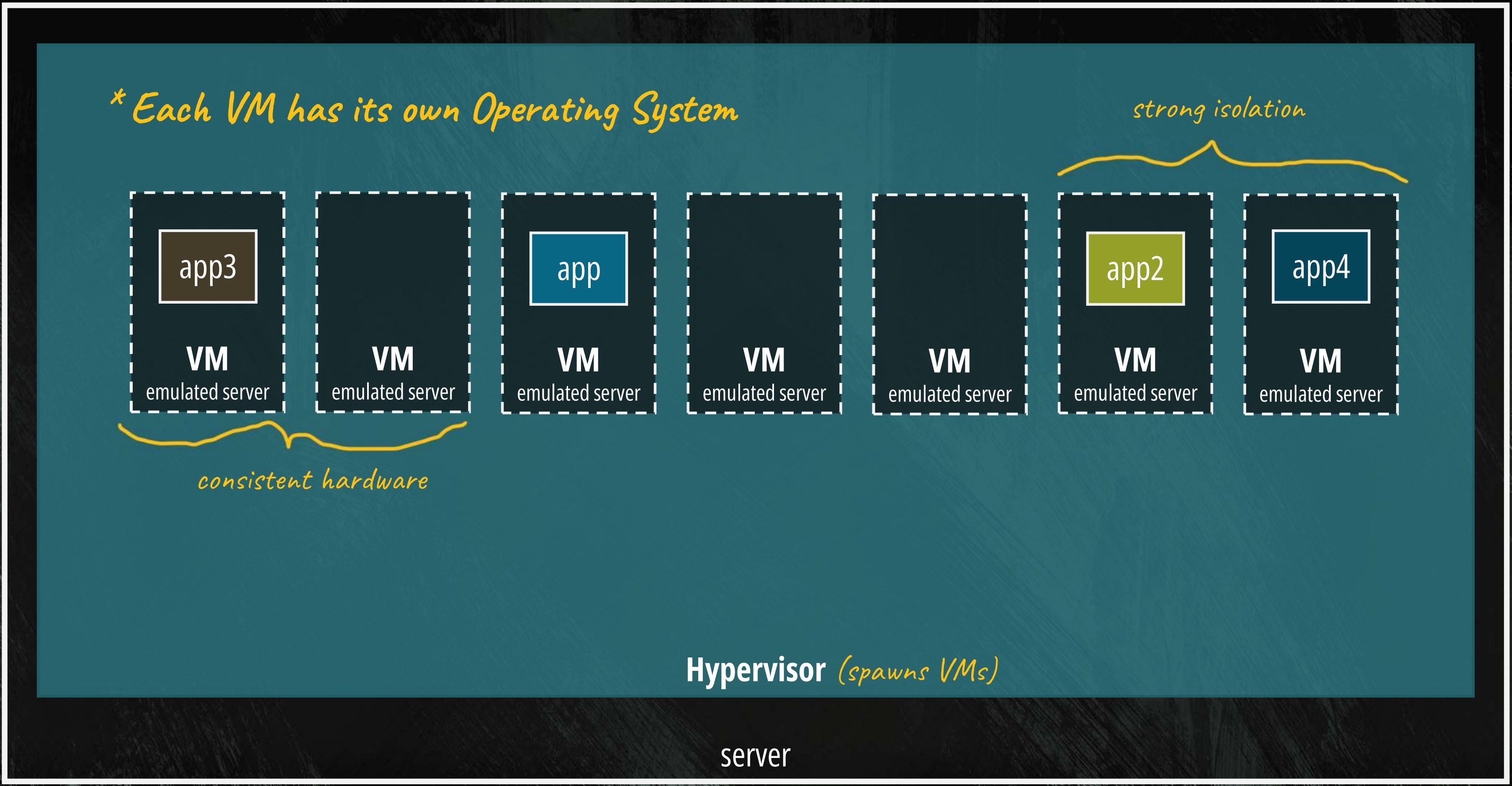
- Underutilization of resources
- Poor Isolation
- Dependency Hell
- Slow provisioning
- Compatibility

Virtual Machines

A **Virtual Machine** (VM) is a software-based emulation of a physical server that runs its own Operating System (OS) allowing multiple VMs to **share the same physical hardware**

Virtual Machines

MARCOLENZO.EU



Virtual Machines Benefits (over Bare Metal)

MARCOLENZO.EU

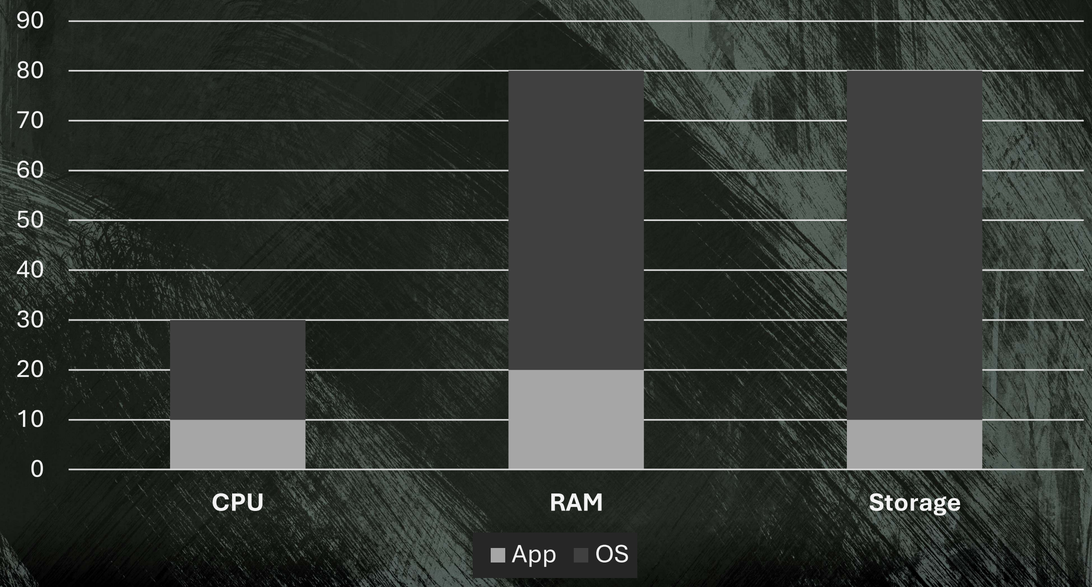
- Better utilization of resources
- Strong isolation (Dedicated OS) ** only if we deploy one application per VM*
  - Less resource contention
  - No Shared libraries
- Fast and automatable creation / scaling / deletion *** ideal for IaC, DevOps, GitOps*
- Compatibility through standardized (virtual) hardware

Virtual Machines Limitations

MARCOLENZO.EU

- OS requires its own resources

App vs OS Resource Consumption



Virtual Machines Limitations

MARCOLENZO.EU

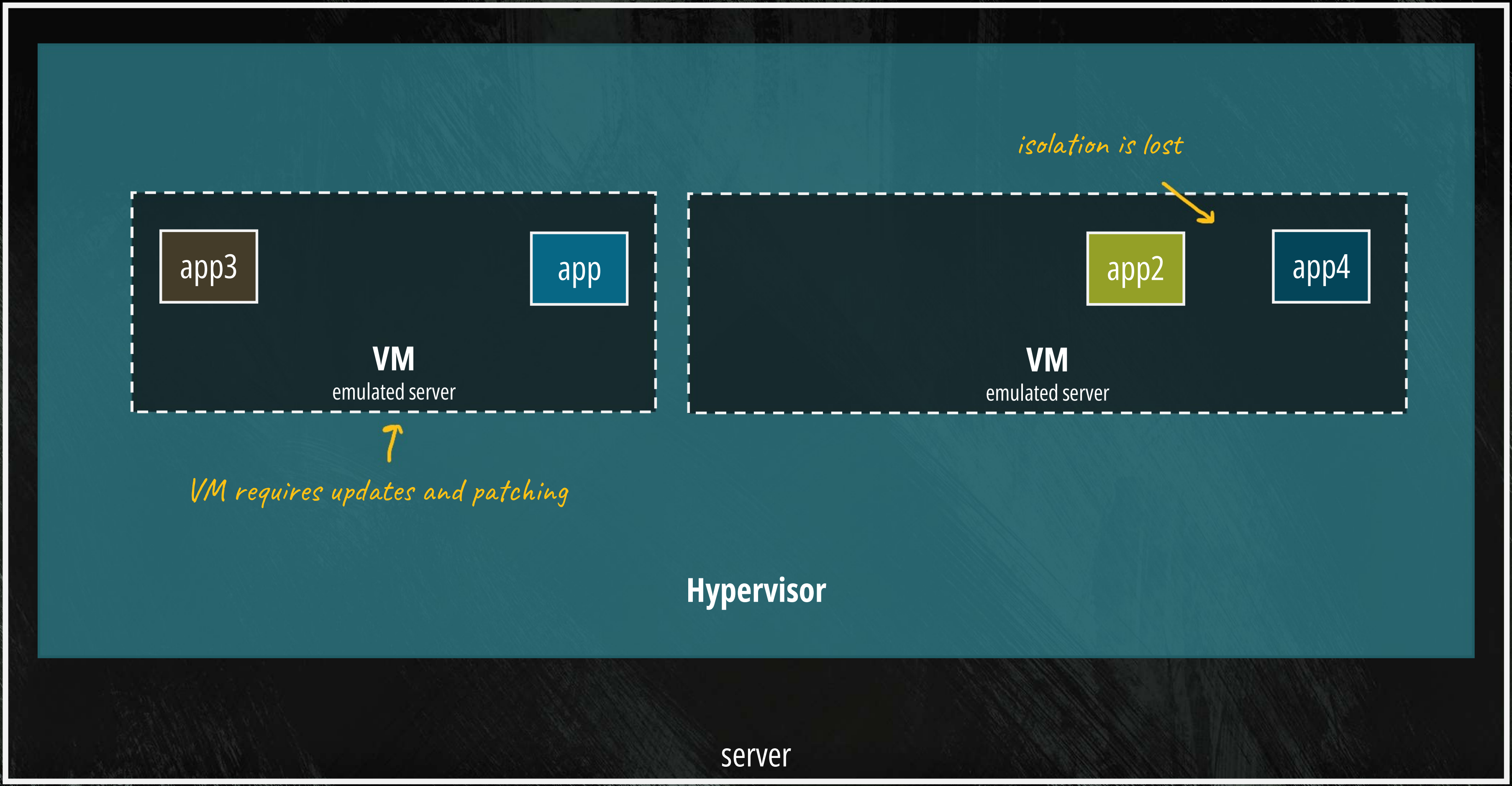
- OS requires its own resources
- Portability
- Slow boot time

Not ideal for highly dynamic environments

(e.g. system of microservices)

Virtual Machines

MARCOLENZO.EU



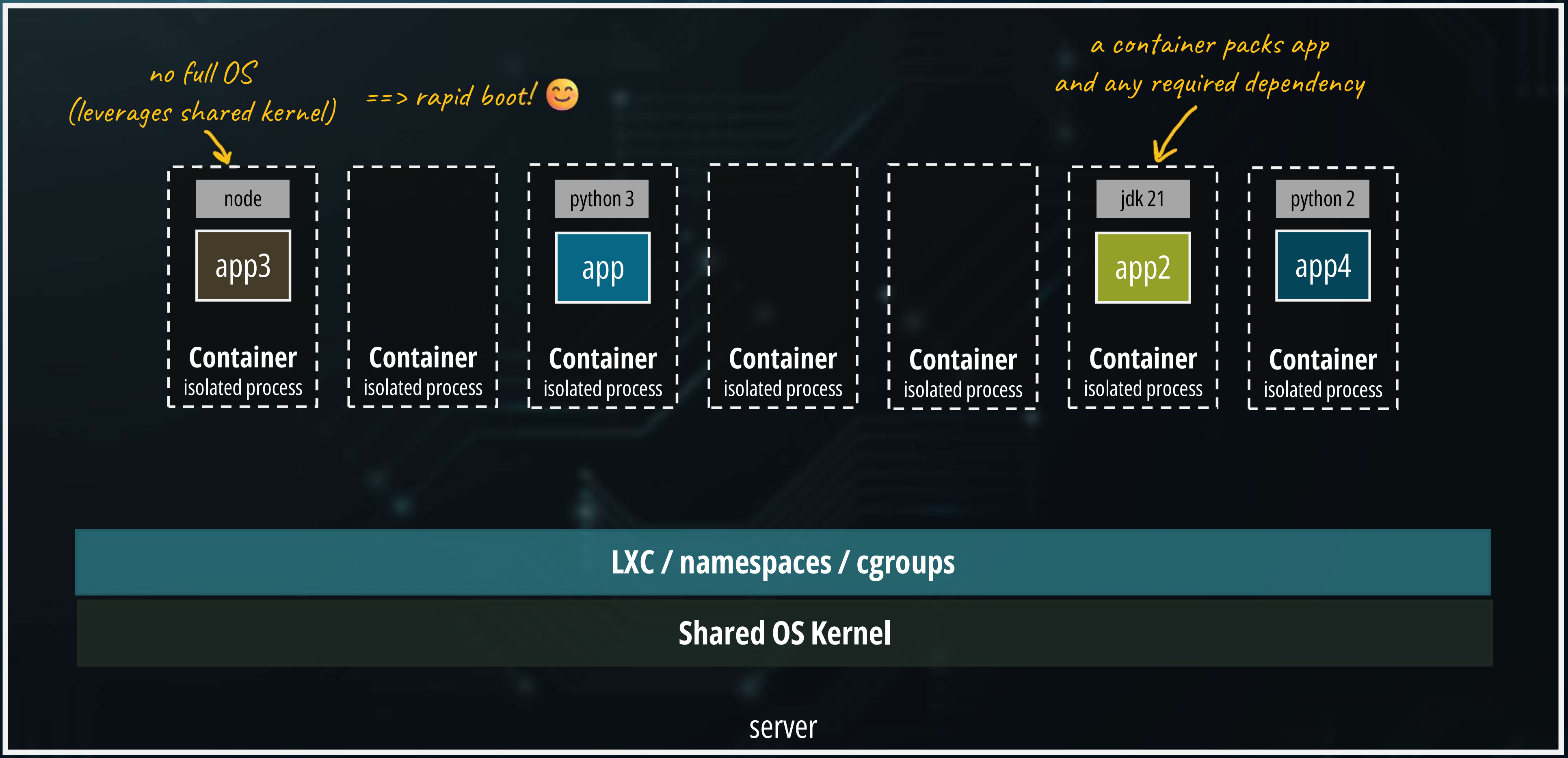
Containers

MARCOLENZO.EU

Containers are a form of virtualization where applications are executed in **isolated environments** running on a **shared kernel**

Containers

MARCOLENZO.EU



VMs vs Containers

MARCOLENZO.EU

	VMs	Containers
Portability	Low	High
Isolation	High	High *
Boot Time	Slow	Fast

* the shared kernel poses a security risk should an attacker escape the container isolation

Containers and Nomenclature

MARCOLENZO.EU

- Container Image
  - Standardized package that contains everything needed to run an application
- Container Runtime
  - Low-level component executing the container as a process on the host
- Container Engine
  - Set of tools that allow us to manage and interact with containers