VMs vs Containers

Garett

What does it take to run a program?

What does it take to run a program?

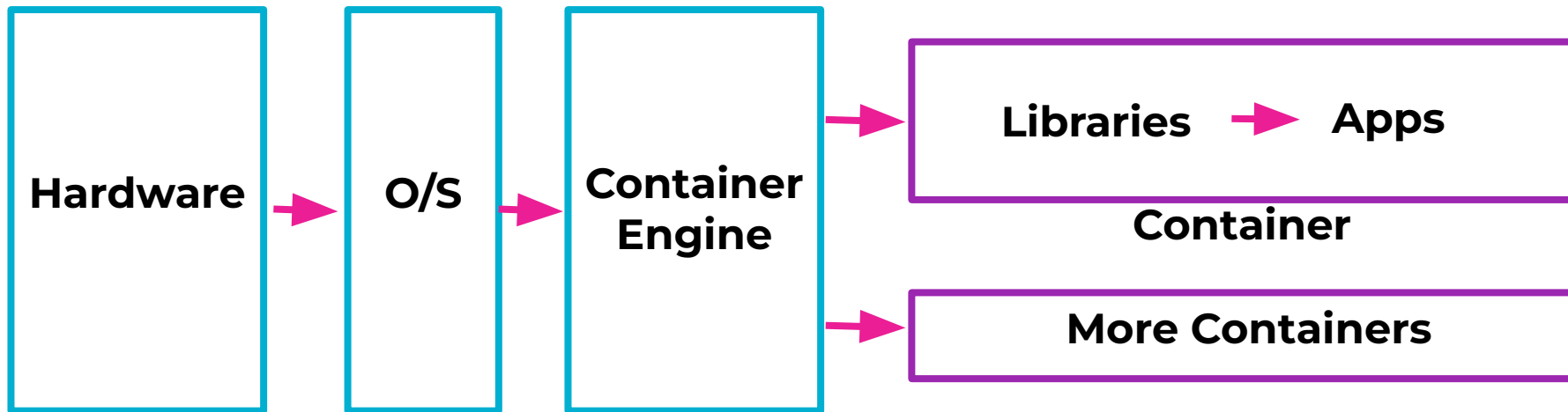| Hardware | → | OS | → | Libs | → | App |
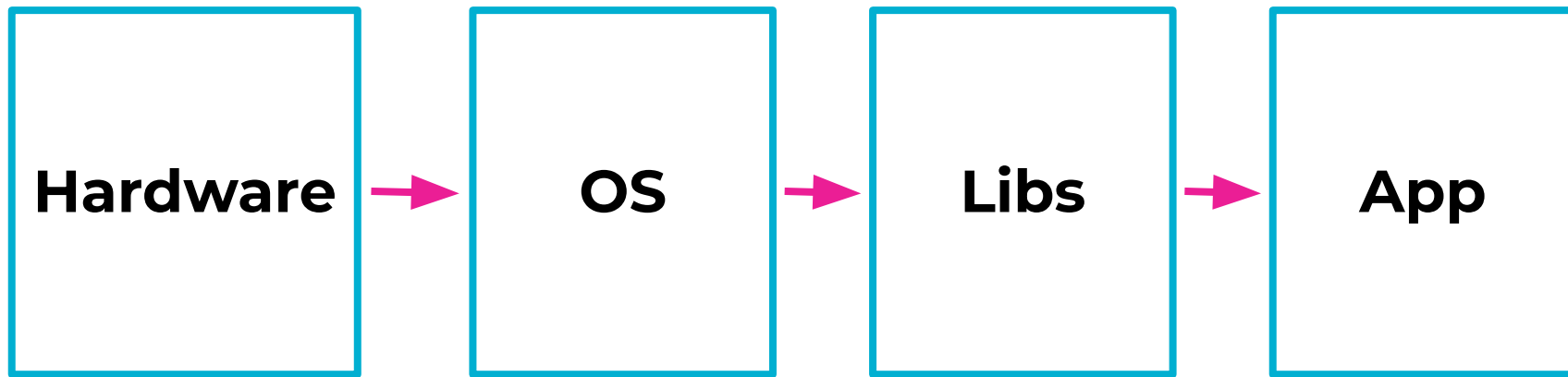
How do we ensure that a program can run consistently across different systems?

Containerization (n.) –

fancy term for packing together everything a process needs to run, in an isolated environment.

Containers

Idea: Virtualize the layers above the host O/S.

Hardware	→	O/S	→	Container Engine	→	Libraries → Apps

Container

More Containers

Containers

Idea: Virtualize the layers above the host O/S.

Example:
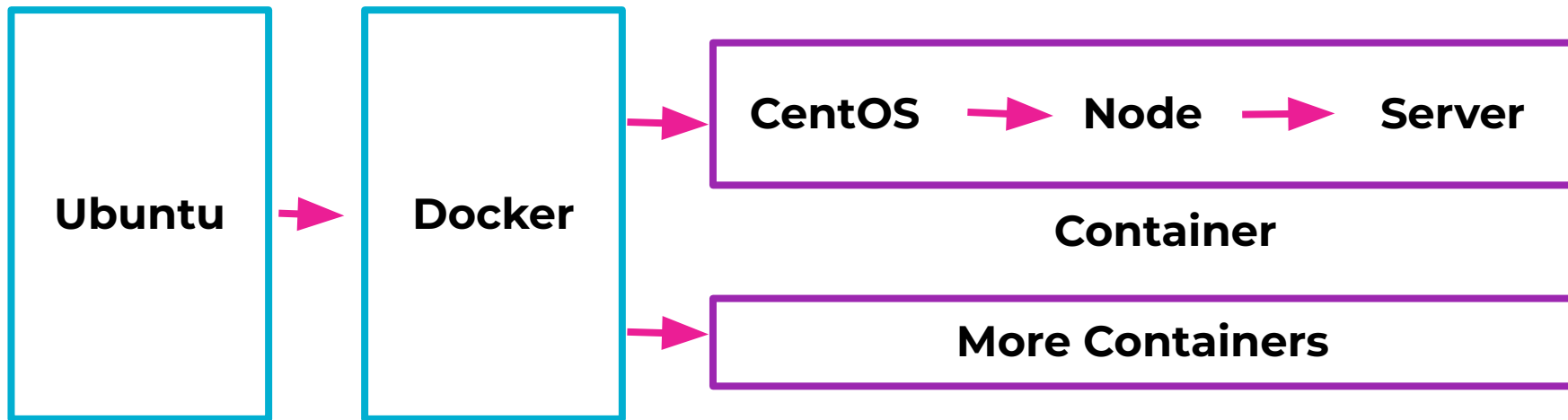
Why Containers?

Pros:

1. Compartmentalization
2. Portability - size, ease of defining a container, versioning
3. Great Ecosystem
4. Uses host kernel for allocation of resources
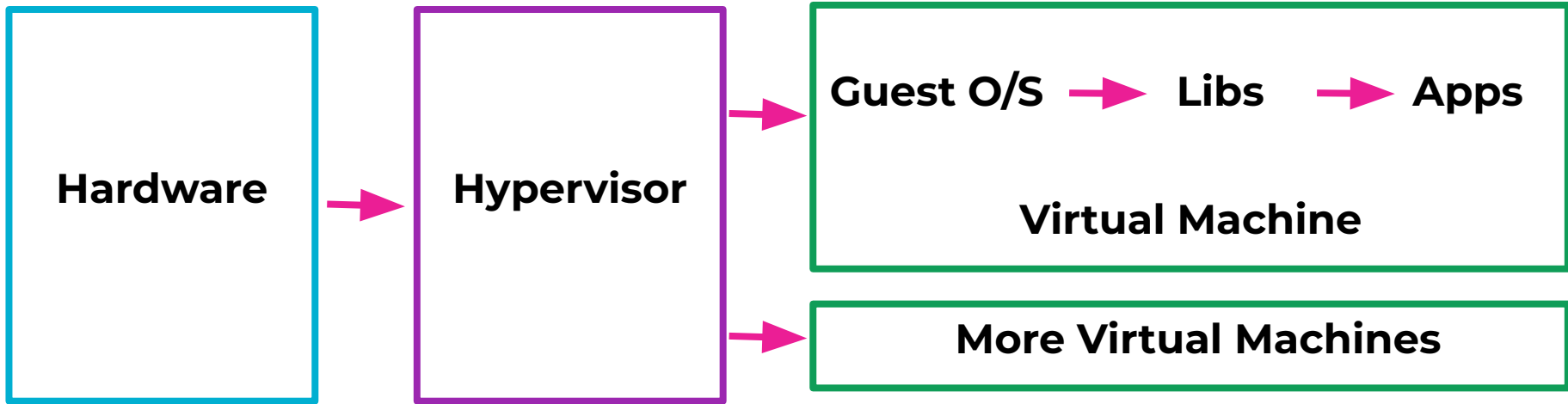
Cons:

1. All containers must run atop the same kernel
2. Less secure due to sharing of underlying OS (e.g. Meltdown)
3. Less flexibility w.r.t hardware requirements

Virtualization (n.) –

abstracting the computer hardware, allowing it to be divided into multiple virtual computers

Virtual Machines (VMs)

Idea: Create a small layer between hardware and operating system that performs this abstraction called the *hypervisor.*

```
┌──────────────┐      ┌──────────────┐      ┌─────────────────────────────────────┐
│              │      │              │      │                                     │
│              │      │              │  ──► │  Guest O/S  ──►  Libs  ──►  Apps     │
│   Hardware   │  ──► │  Hypervisor  │      │                                     │
│              │      │              │      │          Virtual Machine            │
│              │      │              │      └─────────────────────────────────────┘
│              │      │              │      ┌─────────────────────────────────────┐
│              │      │              │  ──► │         More Virtual Machines       │
└──────────────┘      └──────────────┘      └─────────────────────────────────────┘
```

Hypervisors

1. Type 1 (Bare-Metal)
   a. Installed directly on top of the physical server, takes the place of host OS
   b. Most frequently used/most secure, lowest latency
   c. Examples: Hyper-V, KVM, VMWare ESXi

2. Type 2 (Hosted)
   a. Sits on host OS layer, runs as an application in an OS
   b. Allows for interaction between host/guest OS
   c. Higher latency since resources have to be accessed via host OS
   d. Examples: Oracle VirtualBox, VMWare Workstation

Why Virtual Machines?

Pros:

1. Separation of *virtual machines* in terms of computation, logic, and storage
2. Capable of running VMs with different guest OSes
3. Hypervisor has greater control over the amount of system resources each VM is allocated.
4. Full isolation security

Cons:

1. Larger size, less portable
2. Time consuming to build and regenerate

Best of Both Worlds?

Using VMs in tandem with Containers:

1. Emulate a specific hardware configuration with VM
2. Install a container onto the VM

Additional Resources

1. https://www.ibm.com/cloud/learn/hypervisors
2. https://www.atlassian.com/microservices/cloud-computing/containers-vs-vms
3. https://www.ibm.com/cloud/blog/containers-vs-vms
4. https://www.ibm.com/cloud/learn/vmware
5. https://stackoverflow.com/questions/16047306/how-is-docker-different-from-a-virtual-machine
6. https://itnext.io/getting-started-with-docker-1-b4dc83e64389