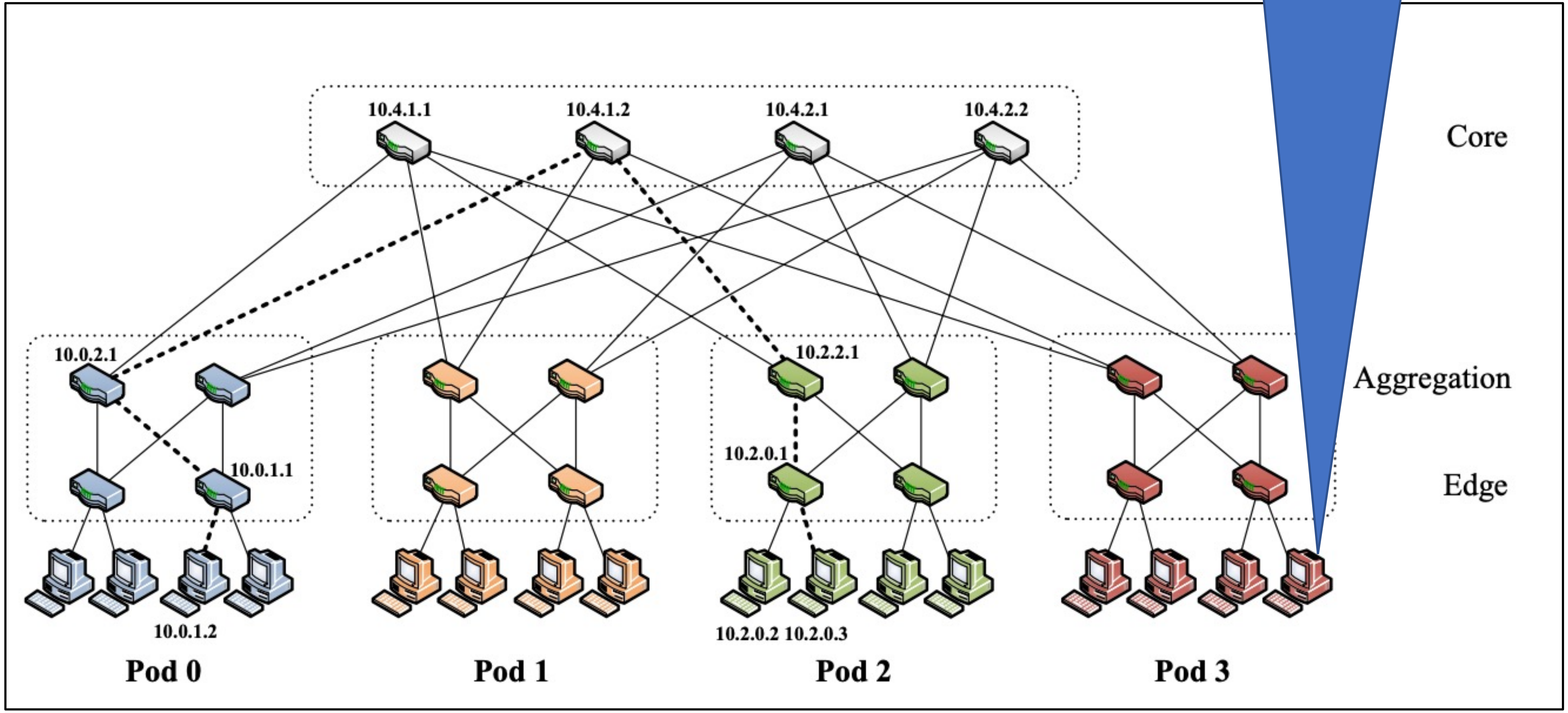
CSE 461: Computer networks

Spring 2021

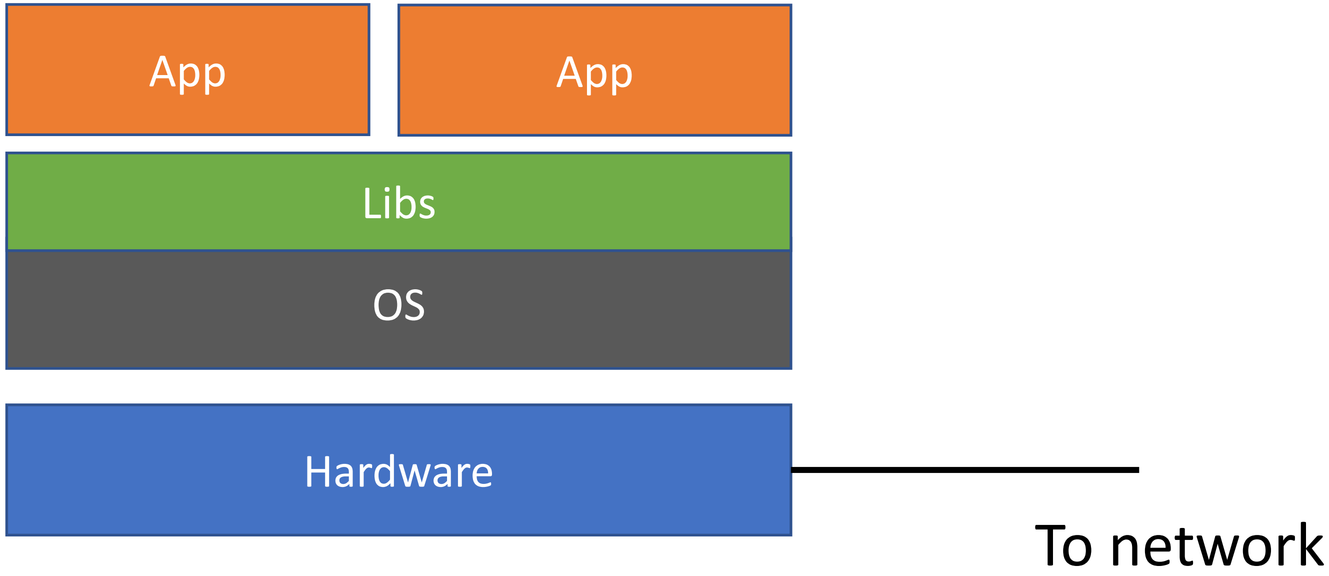
Ratul Mahajan

Containers, etc.

What is in this box?



Originally



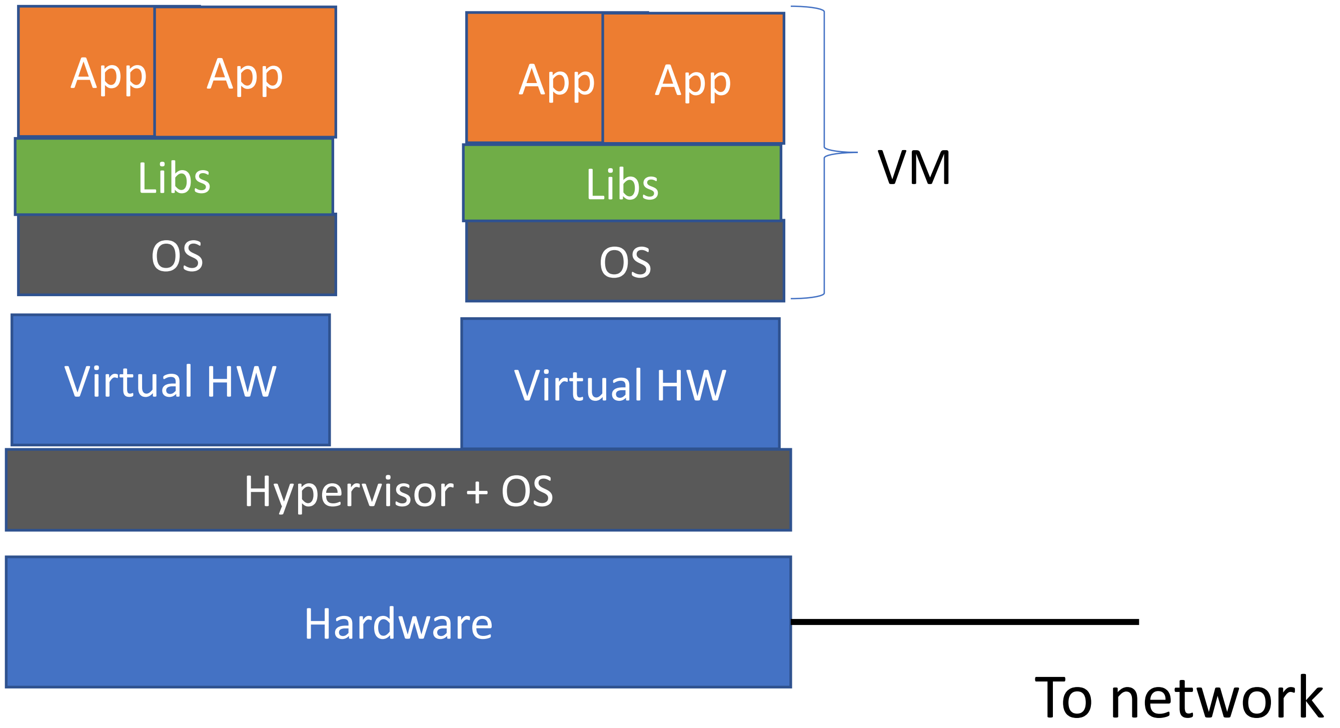
Then came virtual machines (VMs)

HW became too powerful

- Run multiple OSes on the same machine
- Cheaper that way

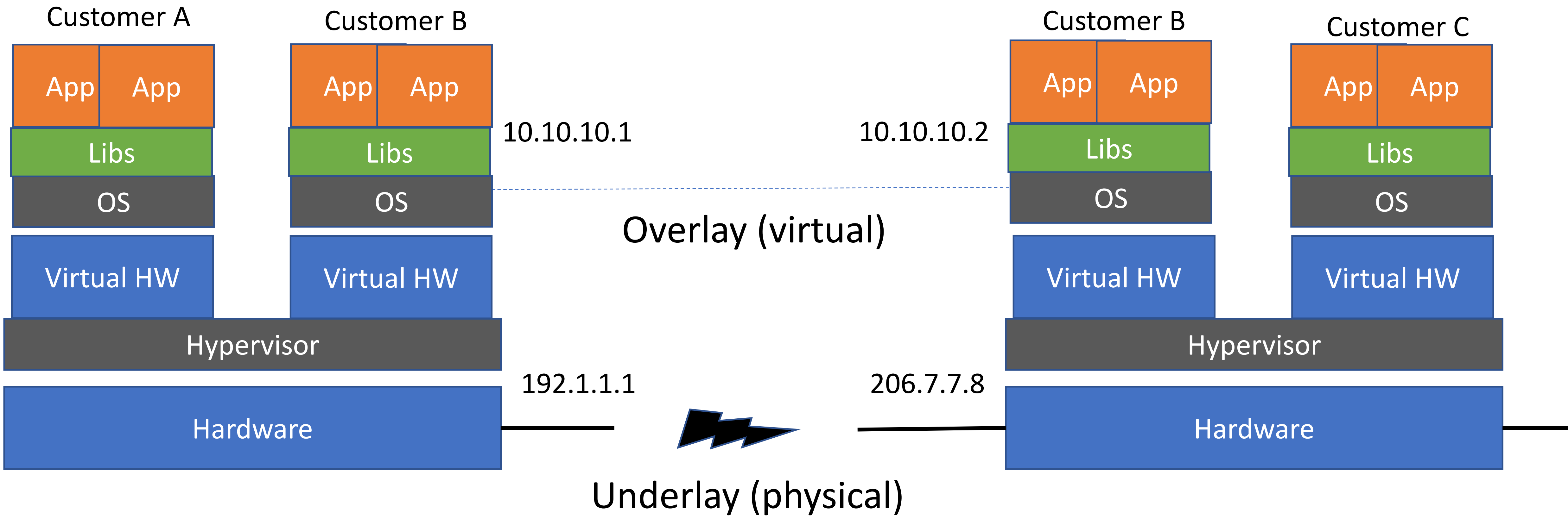
The hypervisor virtualizes the HW and fools the OS

- Provides isolation



The network thinks multiple hosts are connected
The hypervisor acts as a hub for inter-VM traffic

VMs in the cloud



Forwarding between VMs involves a lookup from overlay address to underlay location

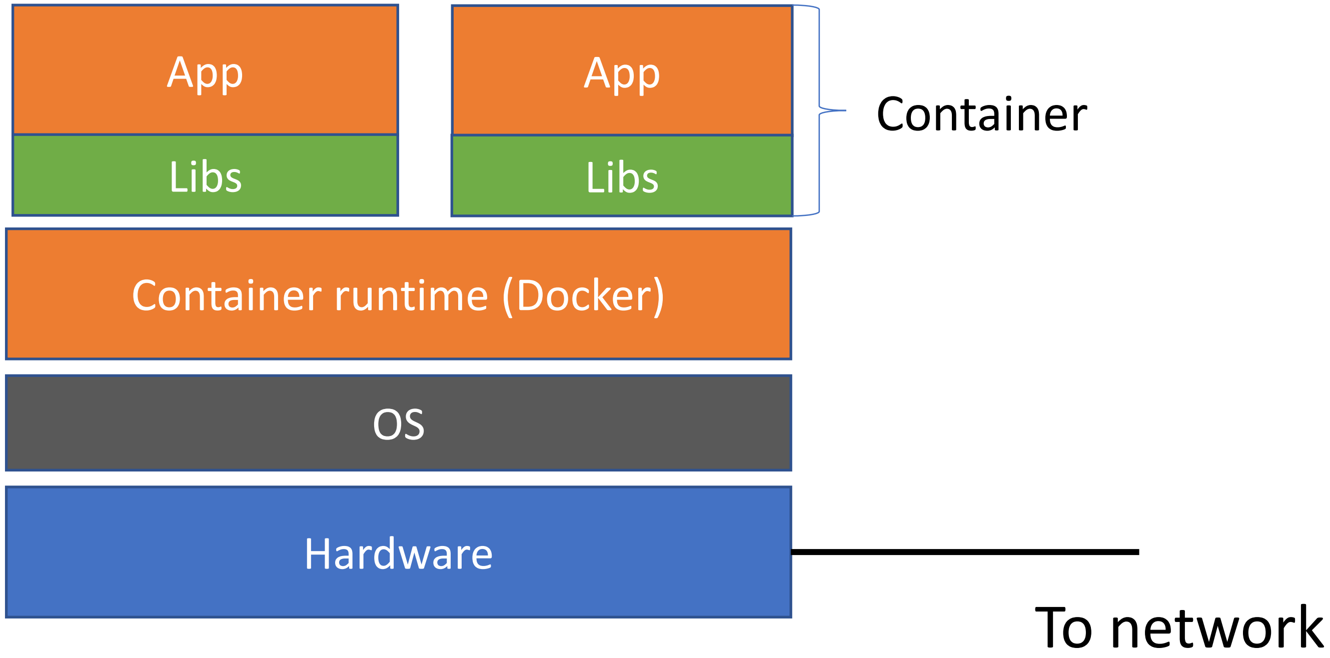
Enter containers

Lighter-weight virtualization than VMs

- Libraries, not the full OS

Better isolation and packaging than apps

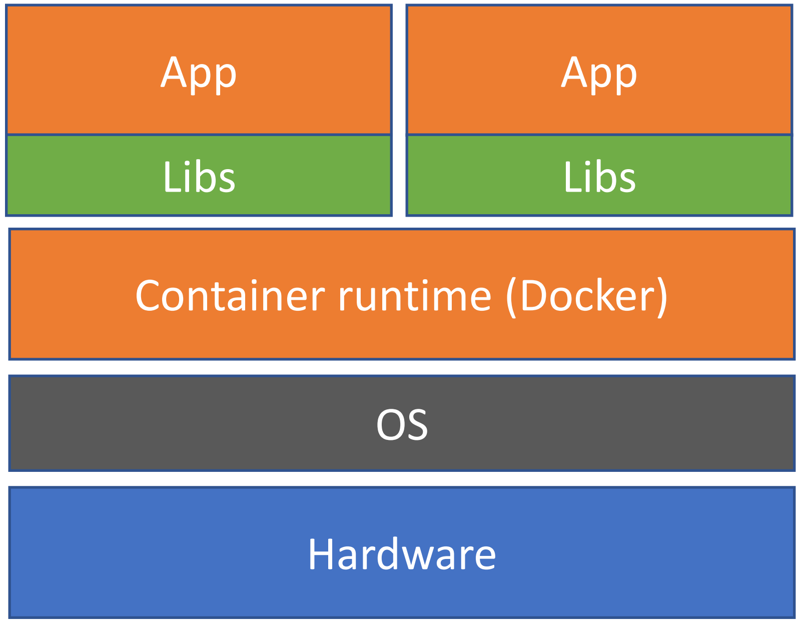
- Bundle the library versions you need



Container networking

Connect containers to the outside world and to each other

- Port conflicts among containers and other apps running on the same host
- High performance between containers on the same host
- (Virtual) private network between related containers (service mesh)



App

App

Libs

Libs

Container runtime (Docker)

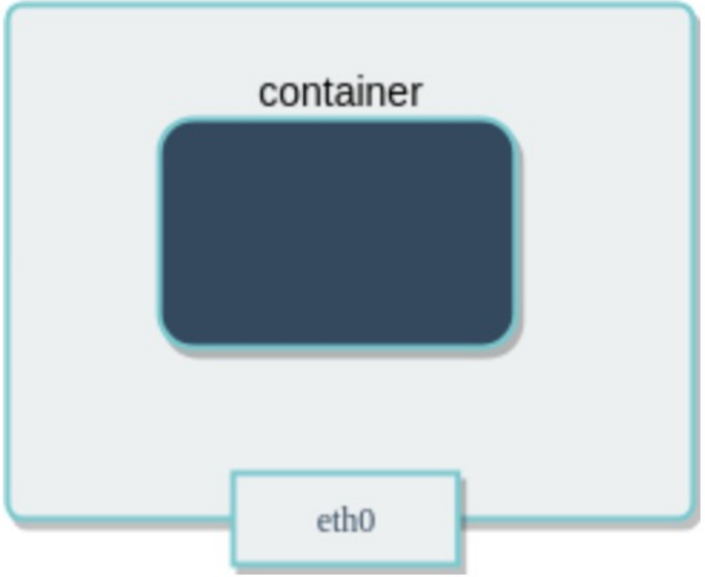
OS

Hardware

Container networking: Host

Containers share the IP address (and networking stack) of the host.

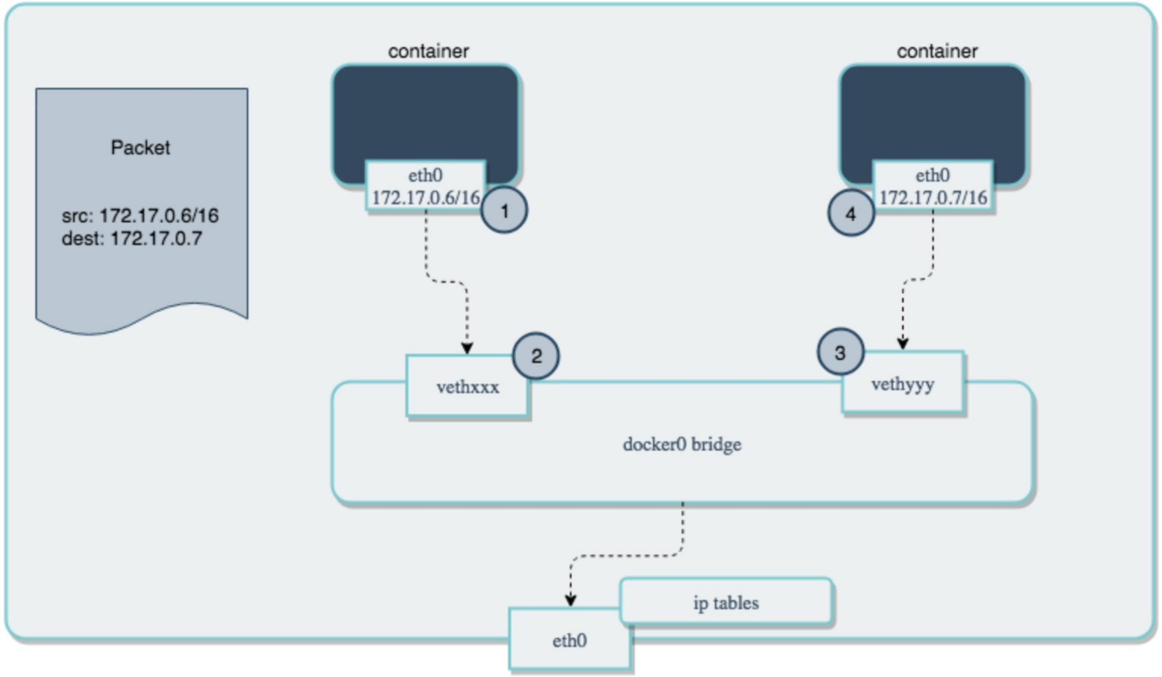
- Cannot handle port conflicts
- Minimal overhead



Container networking: Bridge

An internal network for containers on the same host.

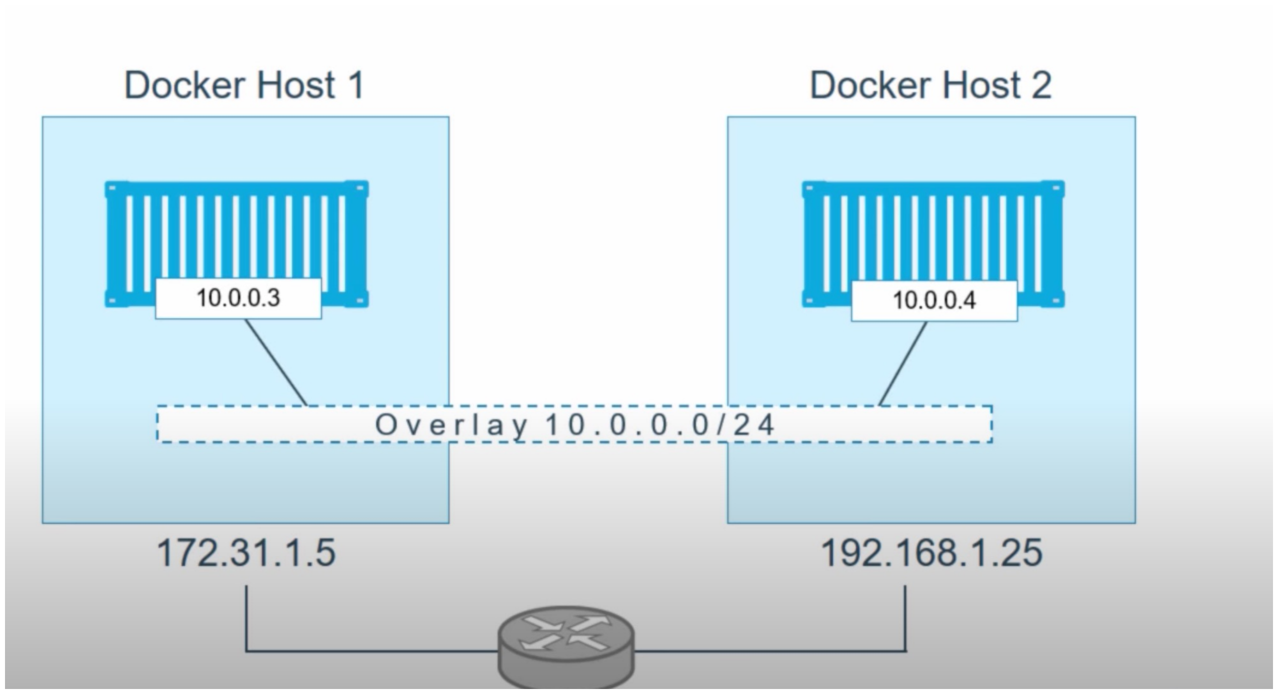
- Use NATs for outside world



Container networking: Overlay

Create a private network across containers on different hosts

- VXLAN is a common way to do that



CNI: Container networking interface

Specification for writing plugins to configure network interfaces

- Decouple runtime from network configuration
- Plugins provide an interface that orchestration engines can use
- GitHub repo: https://github.com/containernetworking/cni

Enter microservices

Instead of a developing a large monolithic application, structure the application as a bunch of communicating microservices

- Each microservice serves a (small) dedicated function, e.g., authentication
  - Can be written in any language
  - Can evolve independent of other microservices
  - Can be scaled independent of other microservices
- Each microservice gets a container

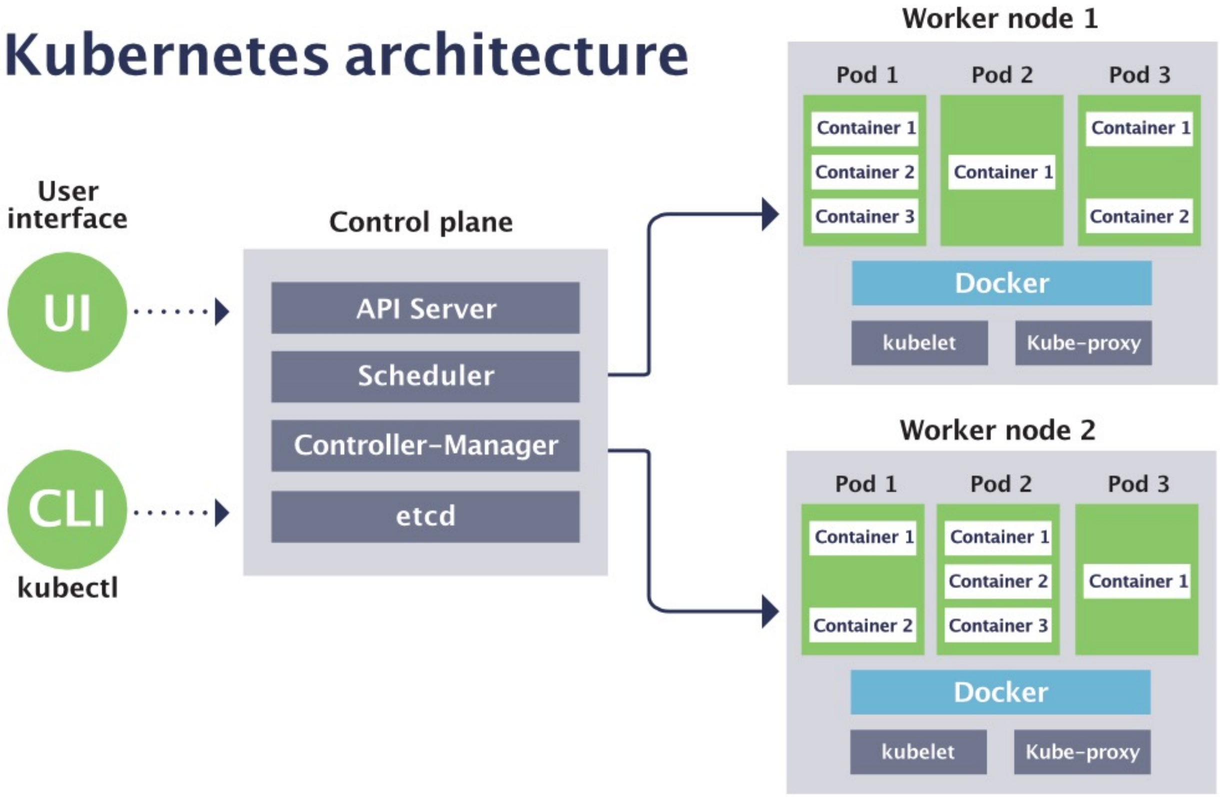
But now you may have lots of services across lots of containers

- Containers need to be deployed and scaled ➔ container orchestration
- Communication between services needs to be managed ➔ service meshes

Container orchestration (Kubernetes)

Containers are wrapped in **Pods** which are run on a **Cluster** of **Nodes**

Pods implement a **service**

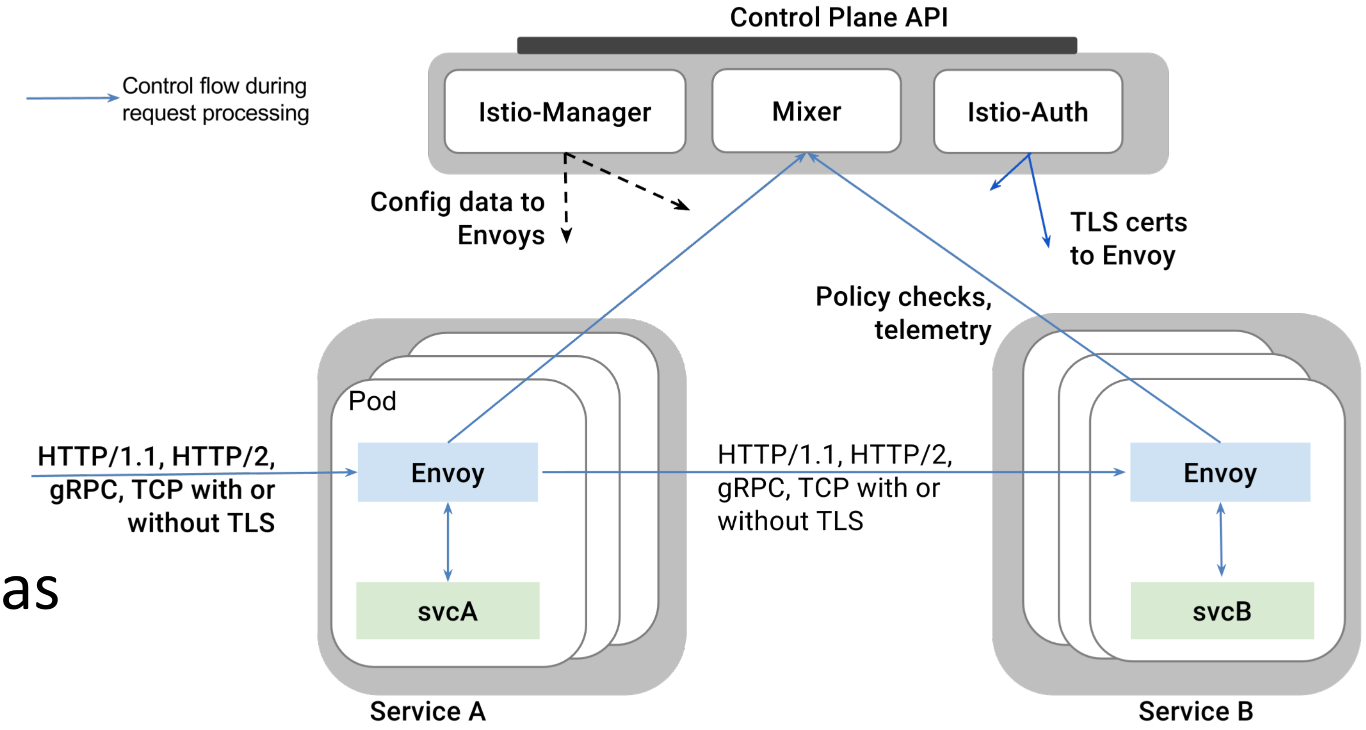


https://sensu.io/blog/how-kubernetes-works

Service meshes (Istio)

"Application defined networking"

- Secure inter-service communication
- Load balancing for HTTP, gRPC, WebSocket, and TCP traffic
- Traffic behavior (routing rules, retries, failover)
- Access control, rate limits, and quotas
- Metrics, logs, and traces



https://istio-releases.github.io/v0.1/docs/concepts/what-is-istio/overview.html