Kubernetes Essential

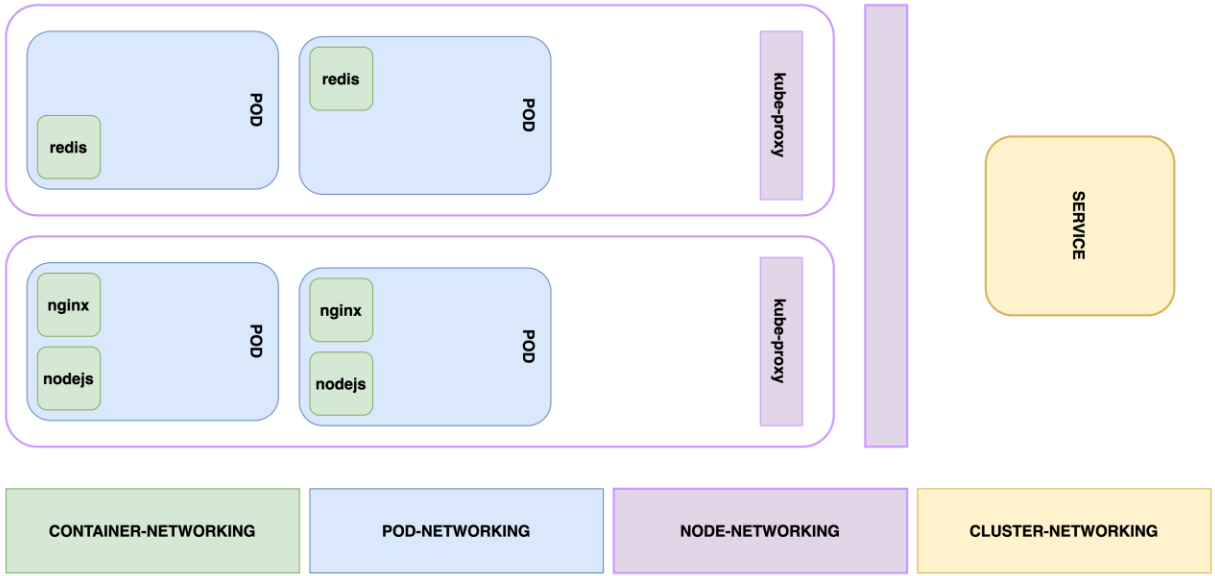
Agenda

- Assignment Review & Guides
- Kubernetes network model
- Network Topology
- Pod networking
- Container Network Interface
- CoreDNS

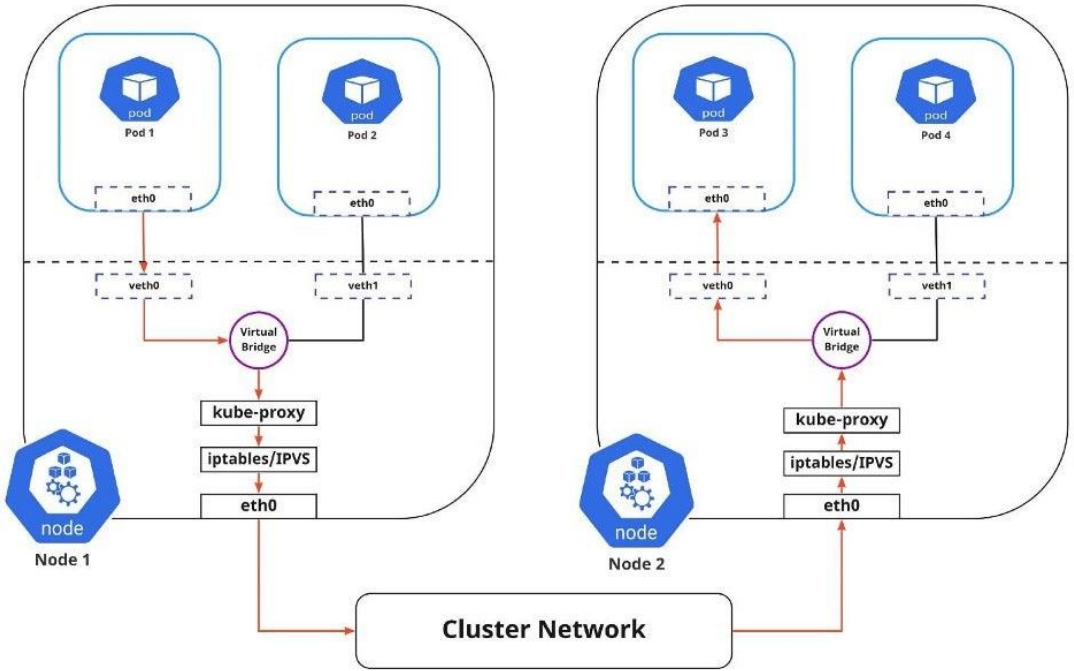
k8s networking model

- Every Pod gets its own IP address: There should be no need to create links between Pods and no need to map container ports to host ports.
- NAT is not required: Pods on a node should be able to communicate with all Pods on all nodes without NAT.
- Agents get all-access passes: Agents on a node (system daemons, Kubelet) can communicate with all the Pods in that node.

Network Topology

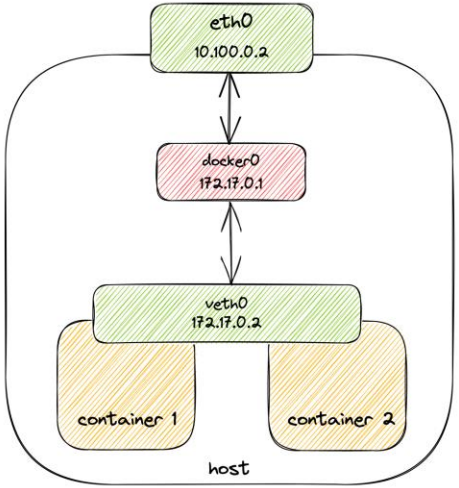


Networking cluster



Container-to-container networking

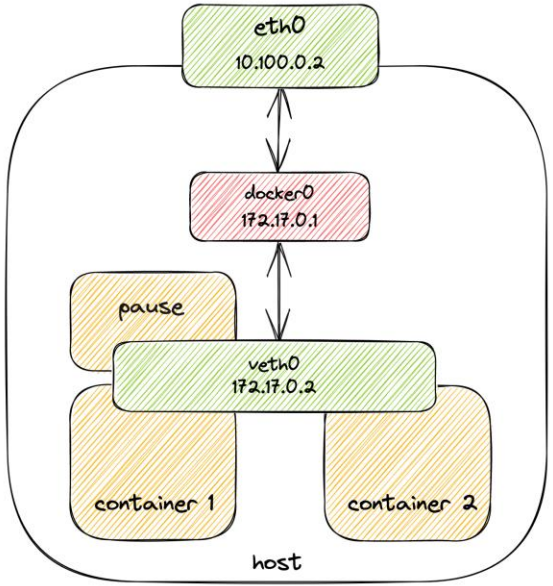
- Both containers will have the same IP address of 172.17.0.2, so the applications inside it can listen to the port on the same IP address and can connect to each other through localhost.
- This also means that two containers cannot listen to the same port, which is a limitation but it is no different from running multiple processes on the same host machine.



Container-to-container networking

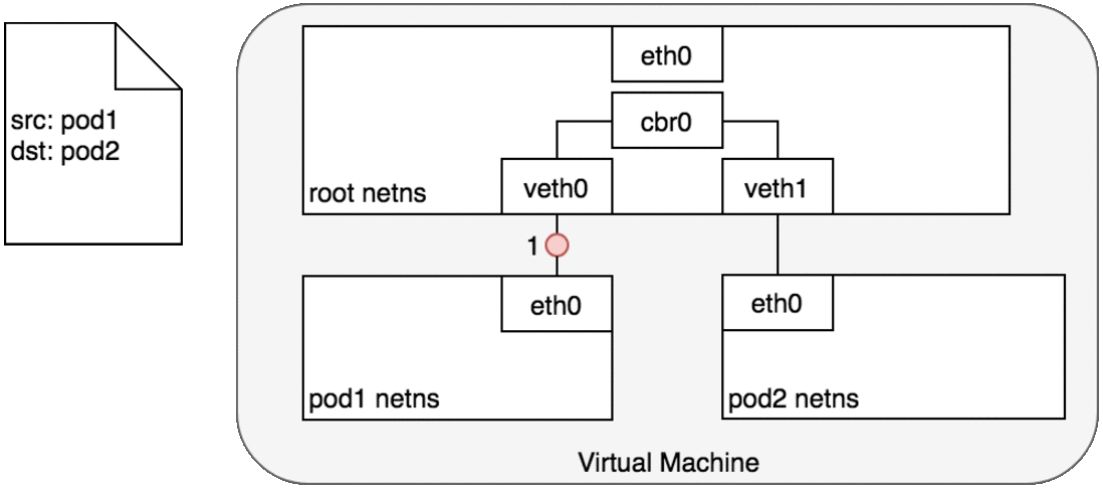
Kubernetes has implemented the above pattern by implementing a special container for each Pod and whose sole purpose is to provide a network interface for the containers.
We can test using the **docker ps** command on the Kubernetes nodes (running Pods):

```
➜  docker ps
CONTAINER ID        IMAGE                  COMMAND ...
...
3b45e983c859        k8s.gcr.io/pause:3.2   "/pause" ...
```

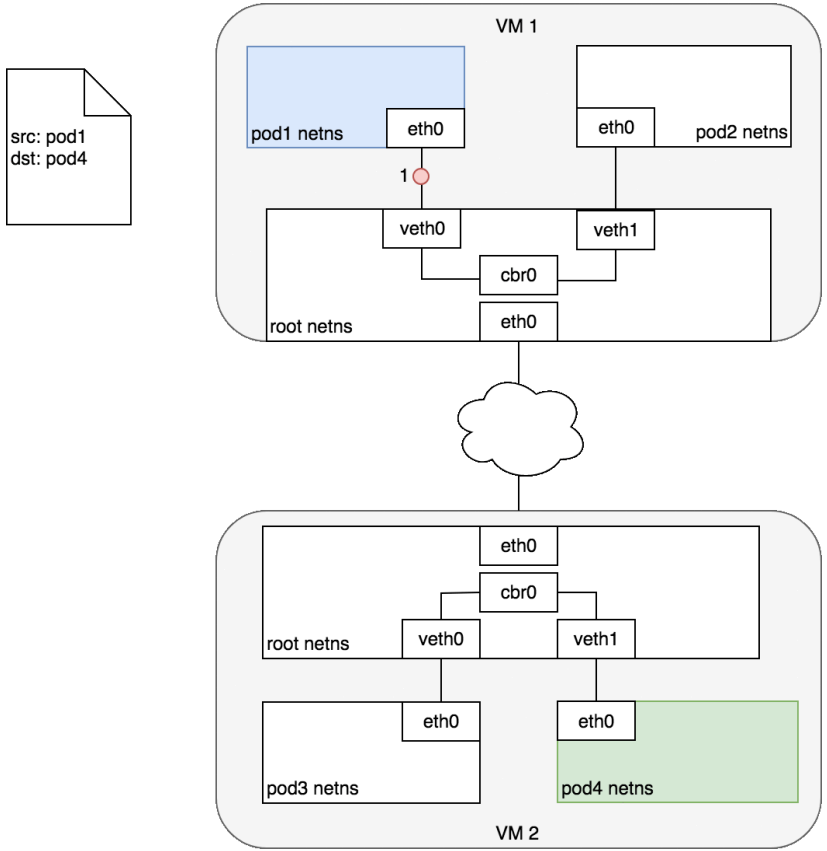


Pod Networking (Pod-to-Pod)

- With Kubernetes, every node has a designated CIDR range of IPs for Pods. This ensures that every Pod receives a unique IP address that other Pods in the cluster can see. When a new Pod is created, the IP addresses never overlap
- Pod-to-Pod communication happens using real IPs



Pod Networking (Pod-to-Pod across Nodes)



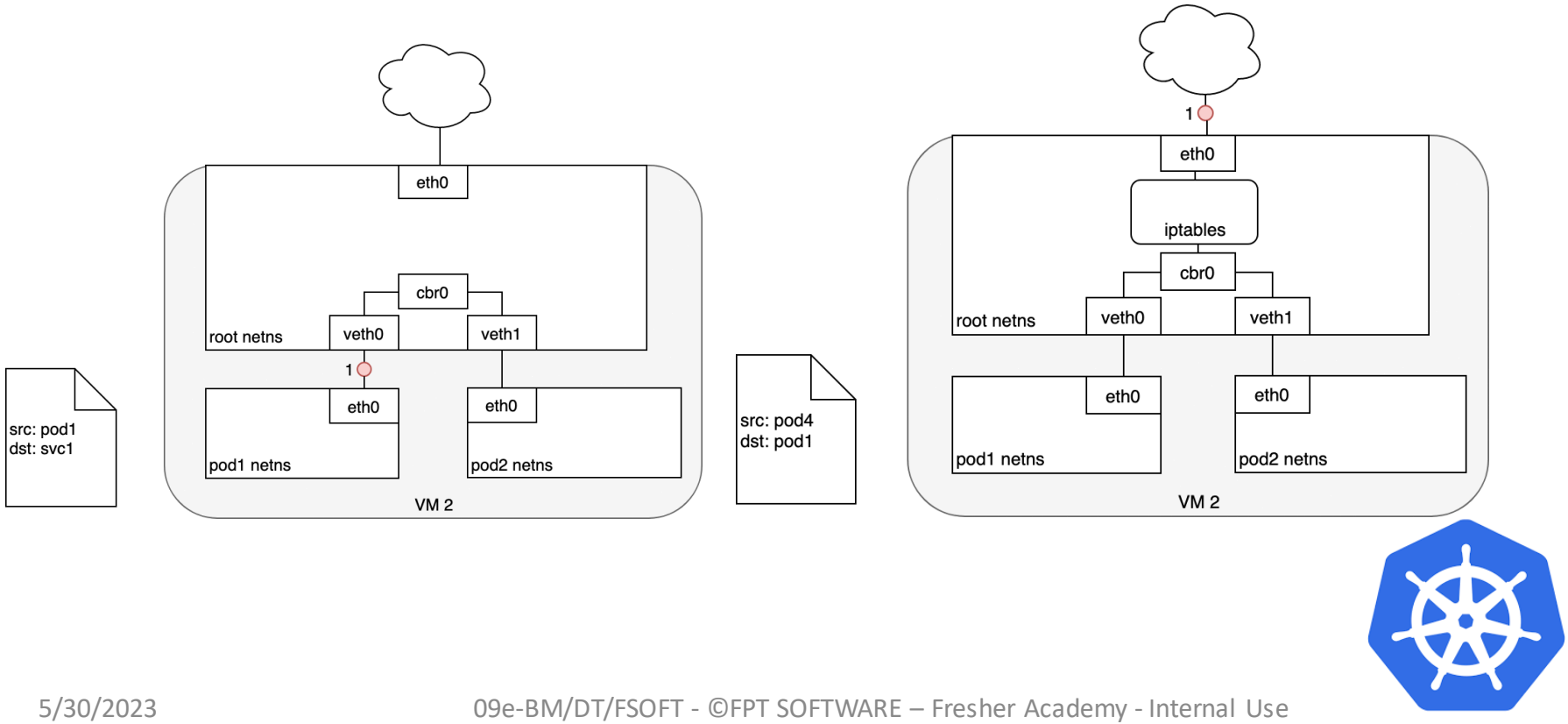
Pod Networking (Pod-to-Service)

- Pod may need to scale up or down based on demand
- Pod may be created again in case of an application crash or a node failure
- These events cause a Pod's IP address to change

Kubernetes solves this problem by using the Service function, which does the following:

- Assigns a static virtual IP address in the frontend to connect any backend Pods associated with the Service.
- Load-balances any traffic addressed to this virtual IP to the set of backend Pods.
- Keeps track of the IP address of a Pod, such that even if the Pod IP address changes

Pod Networking (Pod-to-Service)



Discovering Services (DNS)

Kubernetes can optionally use DNS to avoid having to hard-code a Service's cluster IP address into your application

Q&A