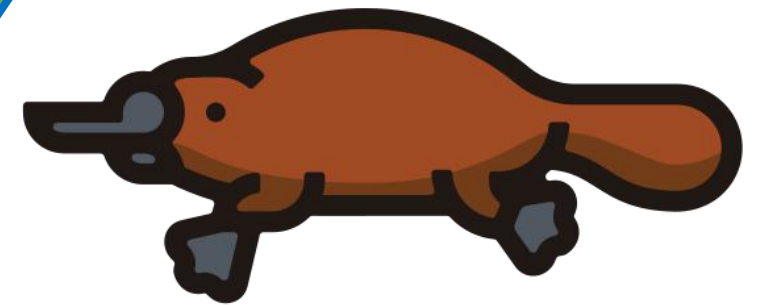


Kuryr Kubernetes

*Interconnecting Kubernetes
with Openstack*

Kubernetes Meetup #7
Google Indonesia, SCBD



KURYR

an OpenStack Community Project

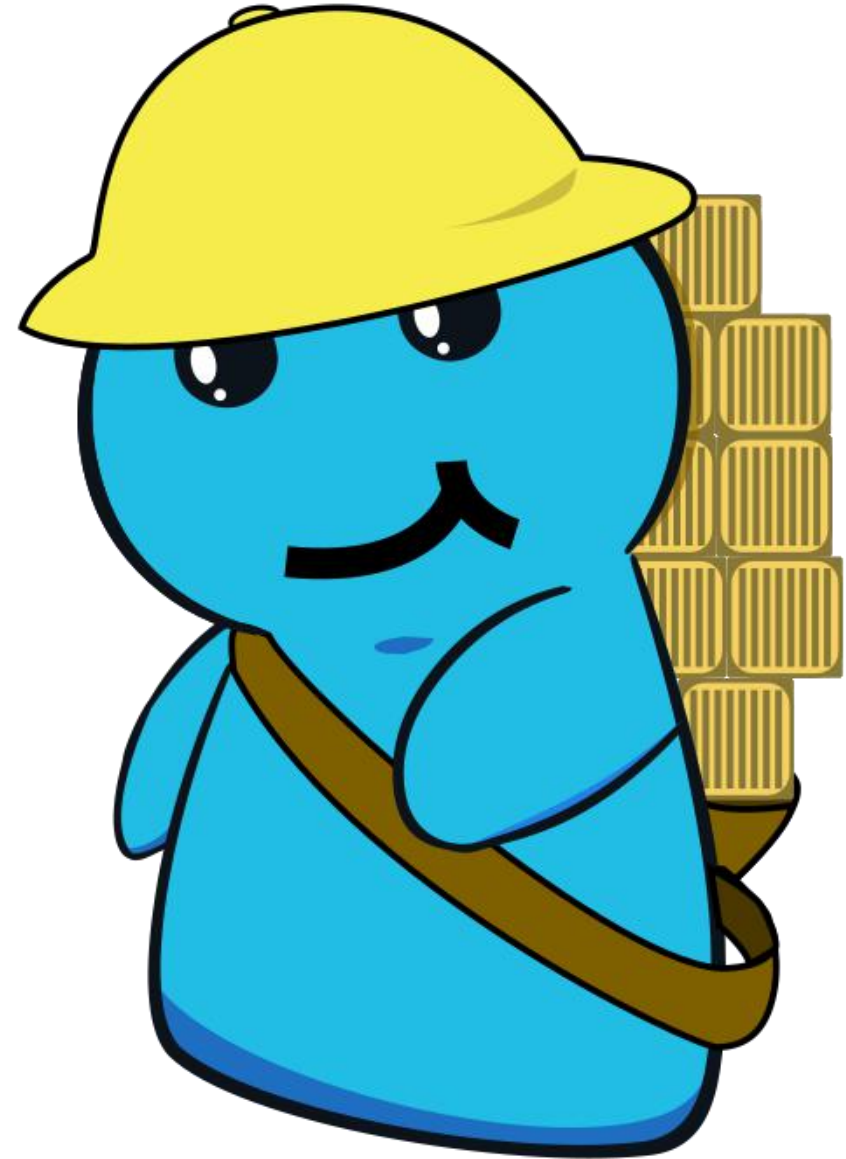
Introduction

- Zufar Dhiyaulhaq
- 6th-semester student,
Telecommunication Engineering,
- Open Networking Foundation
Ambassador (opennetworking.org),
- Cloud Engineer at Boer Technology
(btech.id).



Agenda

- Openstack Networking overview
- Kubernetes Networking overview
- Motivation
- Kuryr-kubernetes concept
- Demos



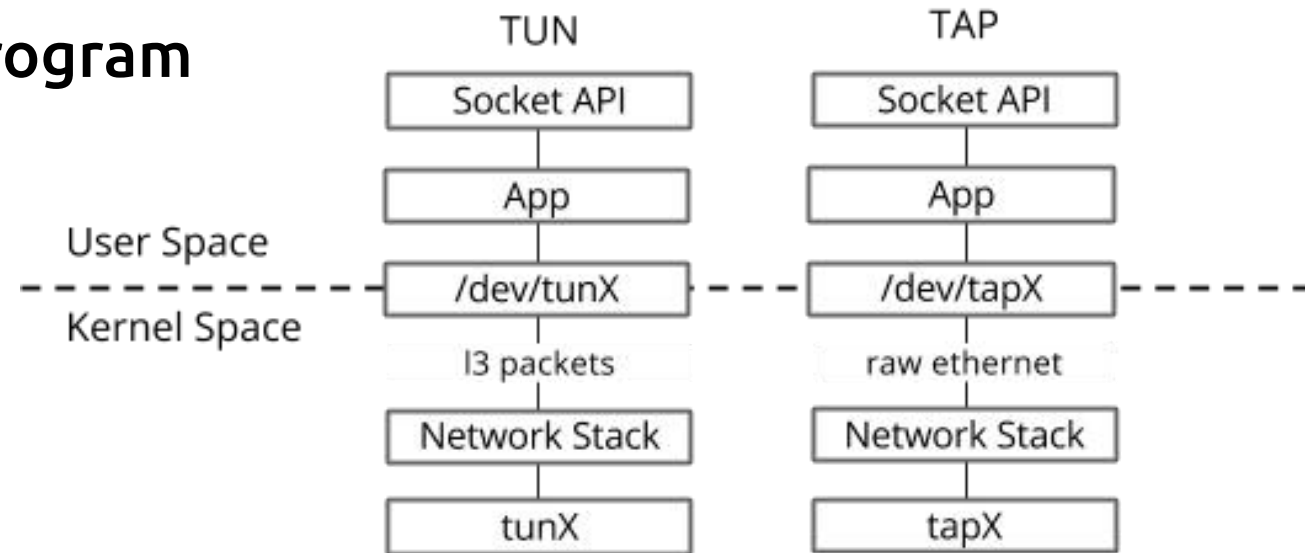
Before Start

Before we start, Five main concepts/Linux virtual network devices

- TAP device
- veth pair
- Linux bridge
- Openvswitch
- Network namespaces

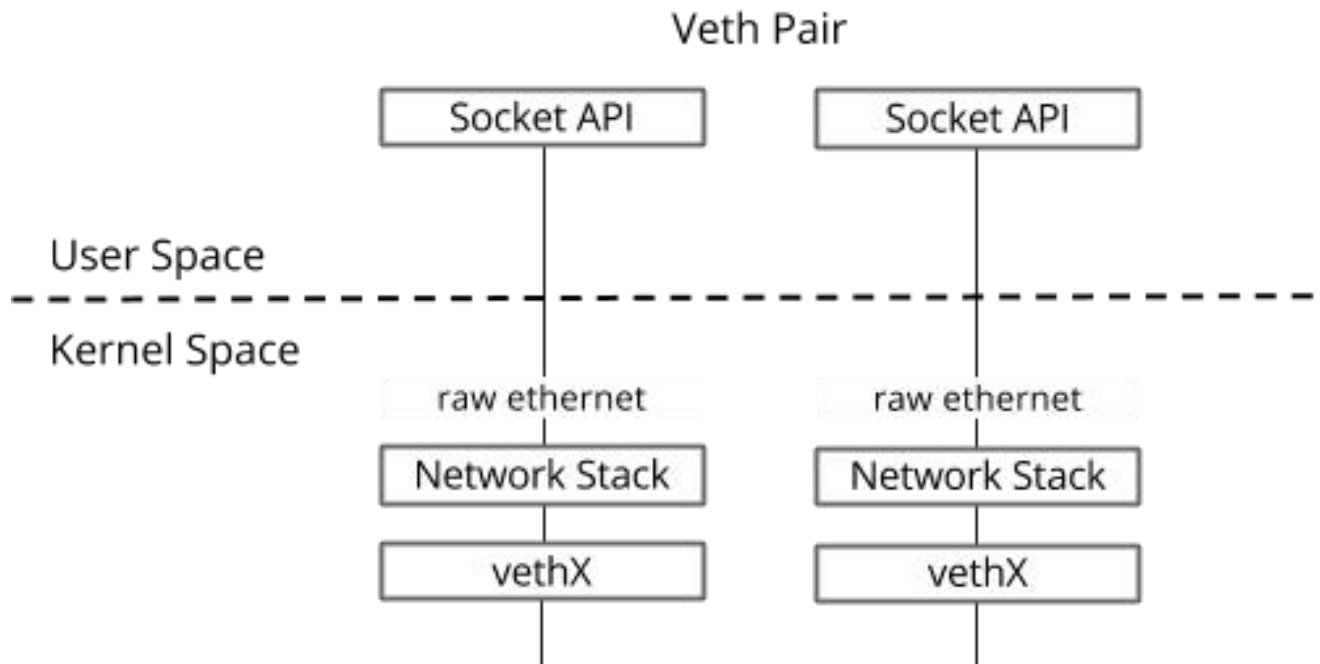
TAP Devices

- Software-only interface. i.e virtual
- Attached to a user-space program
- Accepts Ethernet frames



Virtual Ethernet Pair

- Two virtual NICs connected via a virtual wire
- Used to connect multiple virtual networking components
- Can connect network namespaces



Linux Bridge

- A virtual switch
- Virtual and physical interfaces
- Layer 2

OpenVSwitch

- More complicated virtual switch
- Virtual and physical interfaces
- Layer 2
- Openflow rules

Network Namespaces

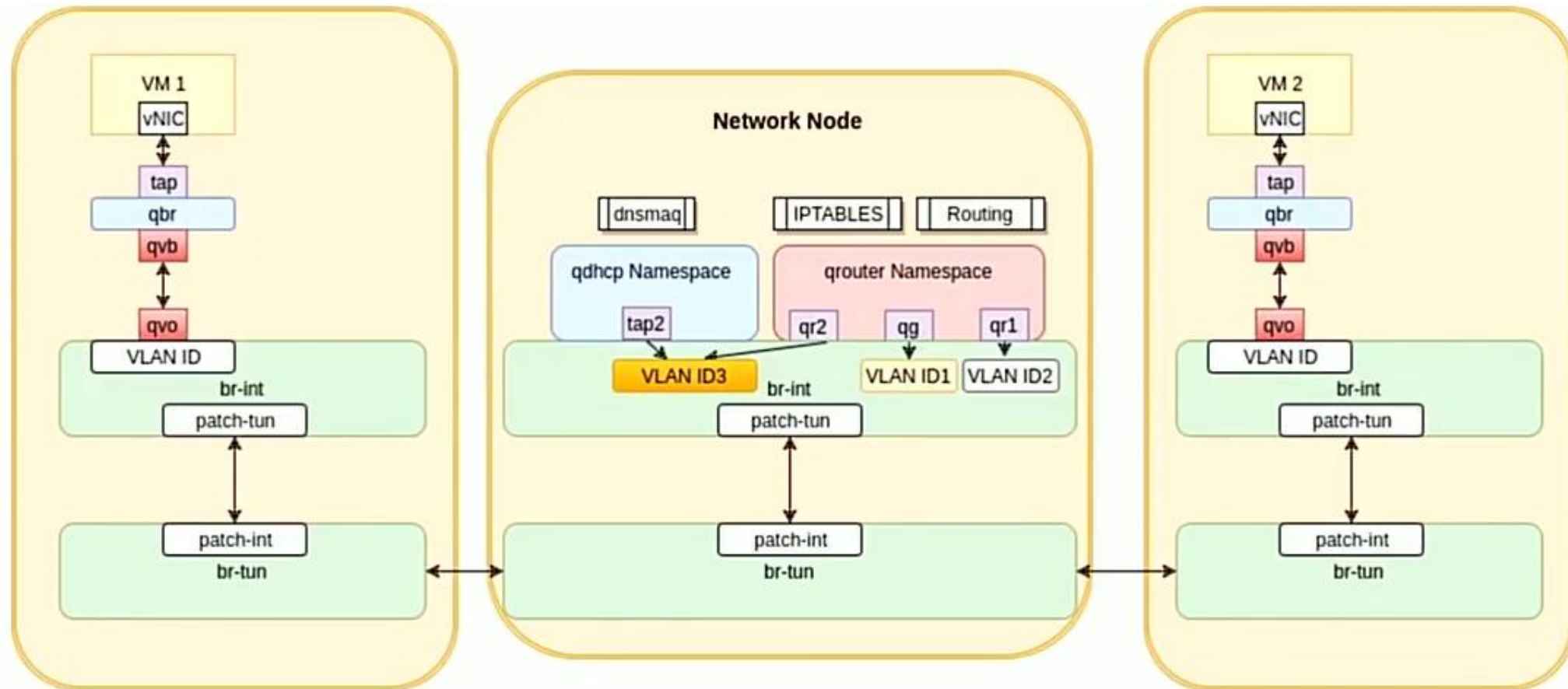
- Isolated network stack
- Interfaces can be assigned to namespaces
- Separate routing tables
- 2 interfaces assigned to 2 different namespaces can have the same IP

Openstack Networking

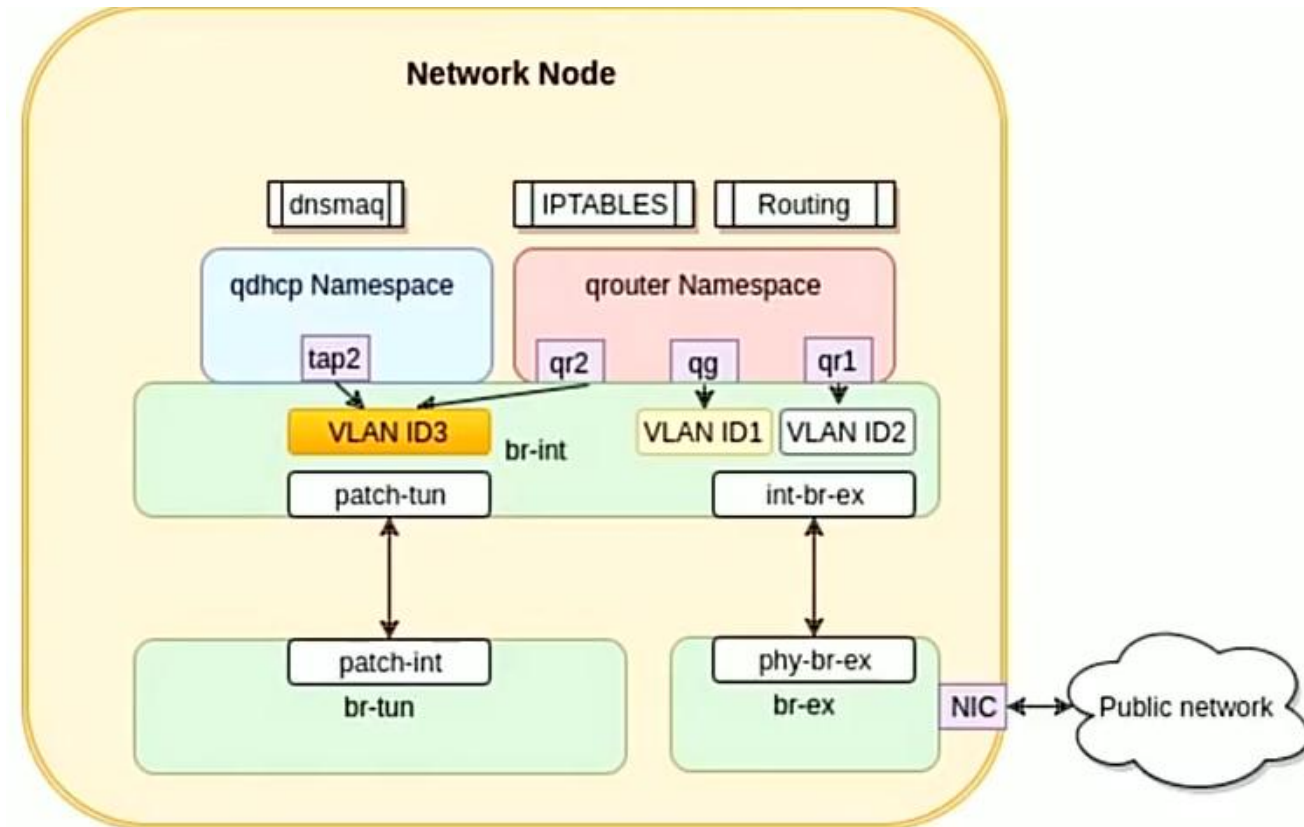
Openstack use neutron as networking component. Neutron use 5 agent to build the network:

- **neutron-server**, provide API, create networks and routers, manage the databases
- **l3-agent**, route L3 traffic, provide floating IP
- **ml2-agent**, switches L2 traffic
- **dhcp-agent**, provide DHCP into instances
- **metadata-agent**, provide metadata into instances

Openstack Networking



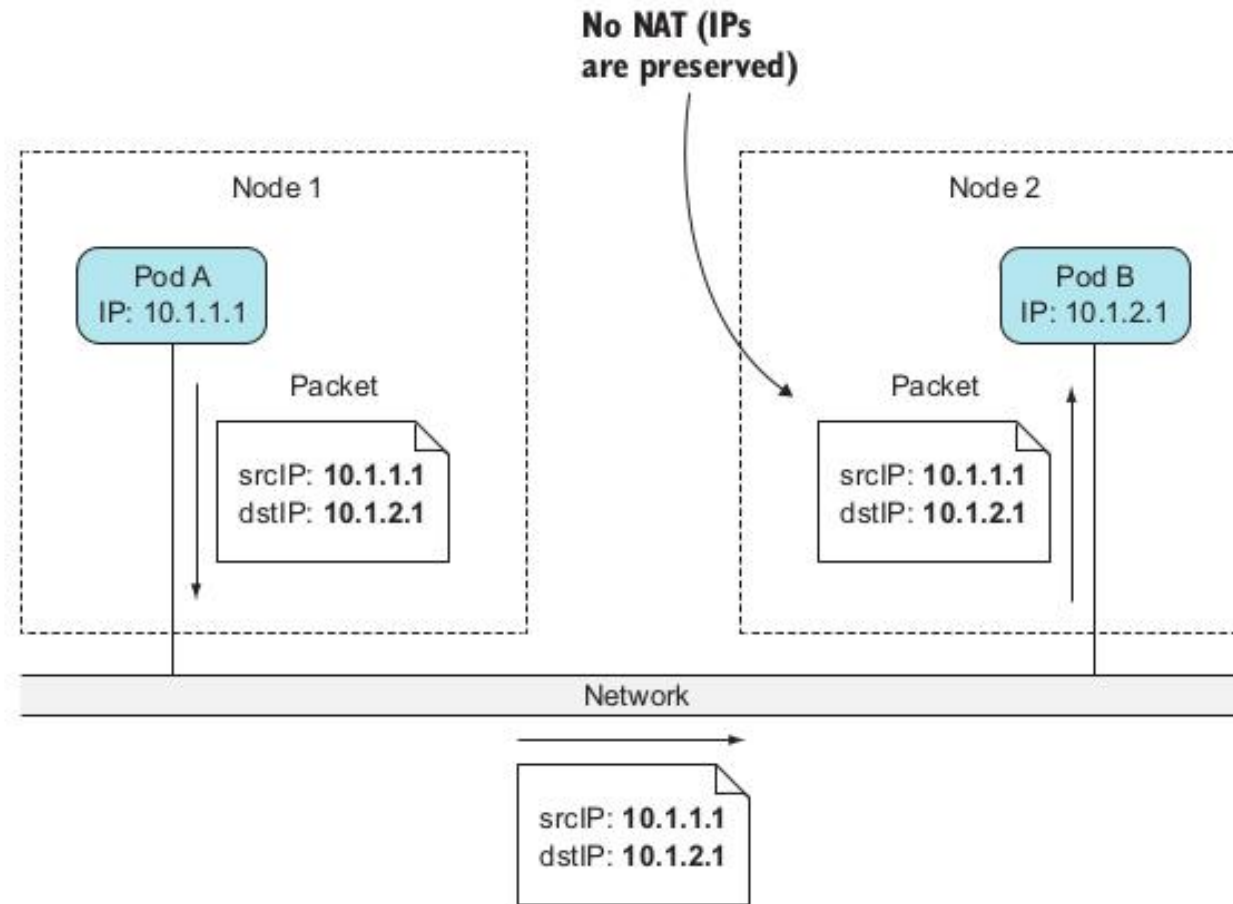
Openstack Networking



Kubernetes Networking

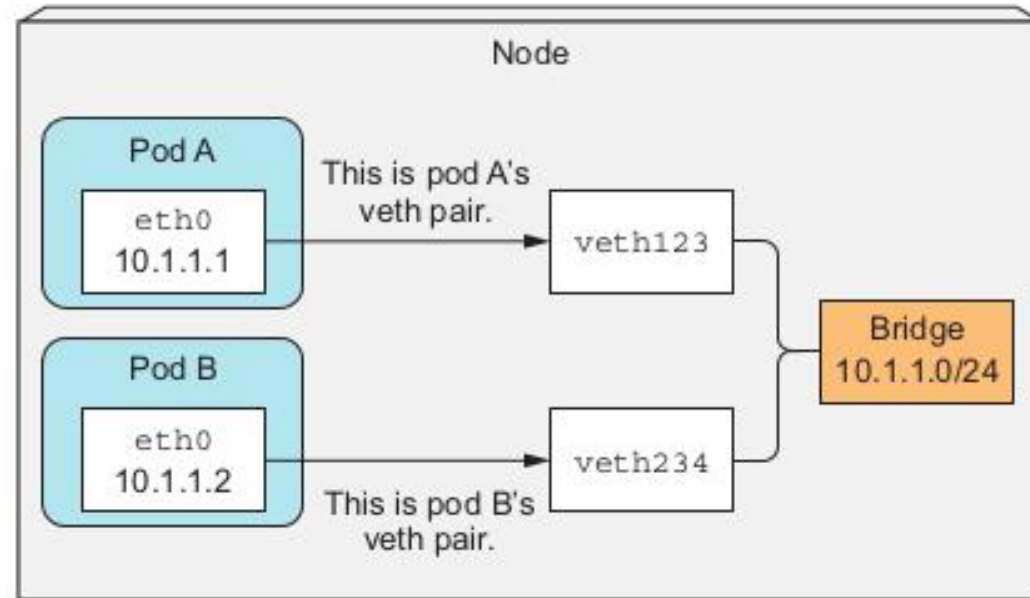
- The network is set up by a Container Network Interface (CNI) plugin, not by Kubernetes itself.
- Kubernetes doesn't require to use a specific networking technology, but it does mandate that the pods (or to be more precise, their containers) can communicate with each other, regardless if they're running on the same worker node or not
- There should be no network address translation (NAT) performed in between—the packet sent by pod-to-pod communication, both the source and destination address unchanged.

Kubernetes Networking

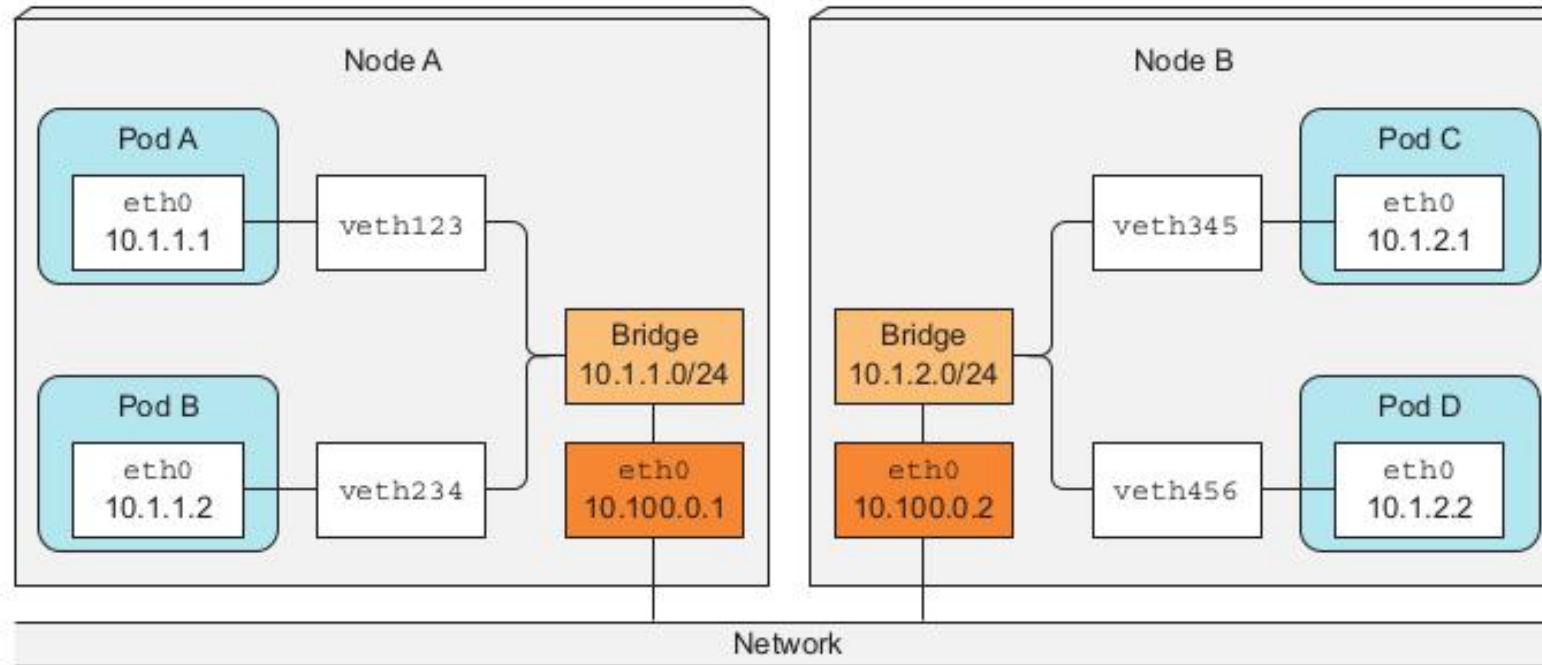


Kubernetes Networking

- a virtual Ethernet interface pair (a veth pair) is created for the container.
- If pod A sends a network packet to pod B, the packet first goes through pod A's veth pair to the bridge and then through pod B's veth pair.
- All containers on a node are connected to the same bridge, which means they can all communicate with each other



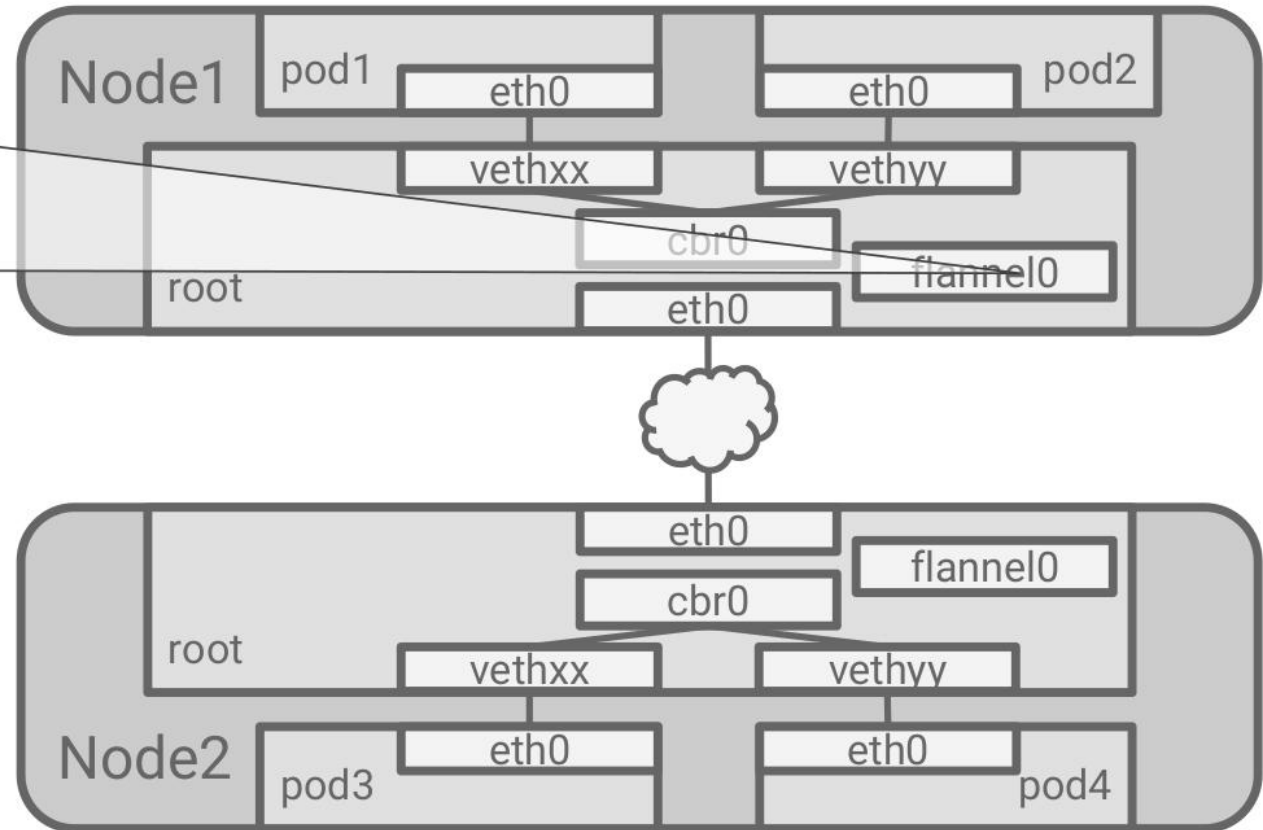
Kubernetes Networking



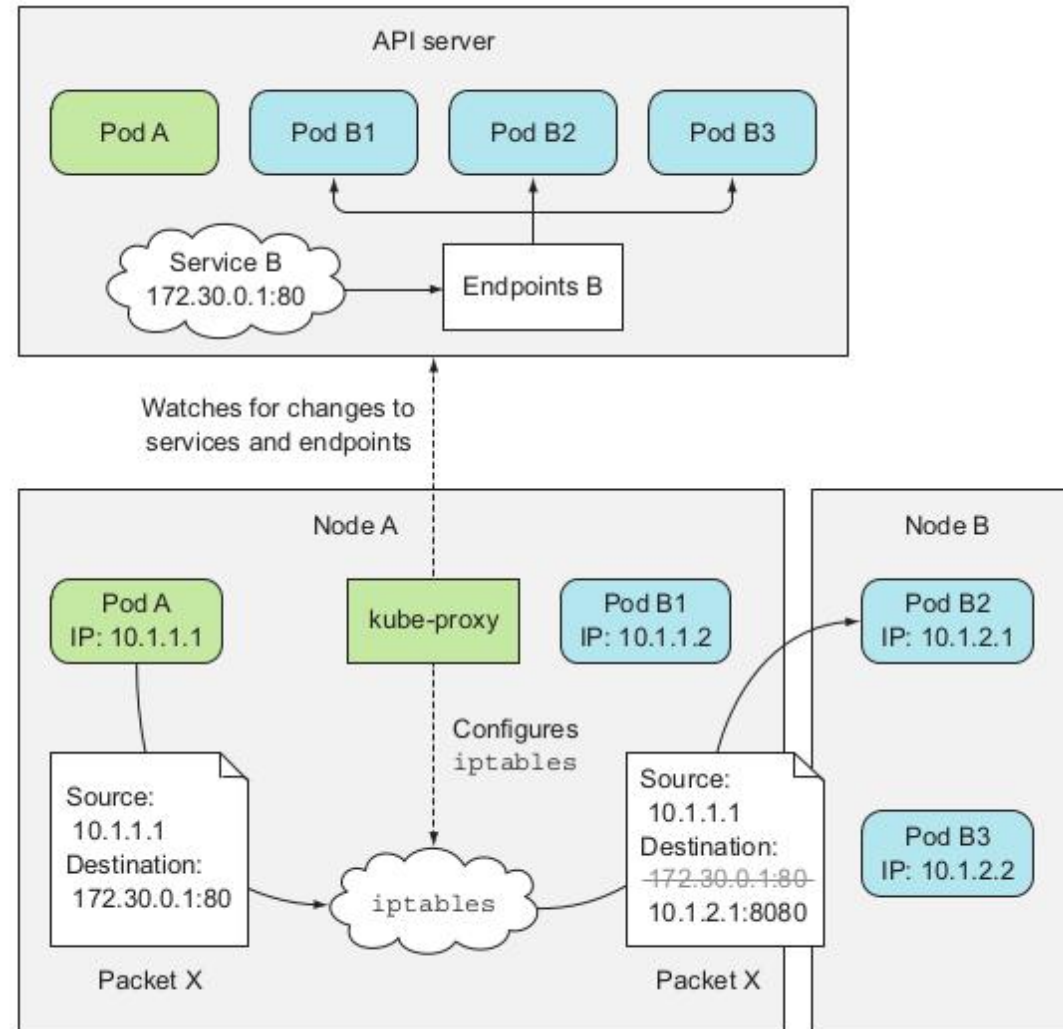
to enable communication between containers running on different nodes, the bridges on those nodes need to be connected somehow.

Kubernetes Networking

VXLAN interface
acts like any
other vNIC



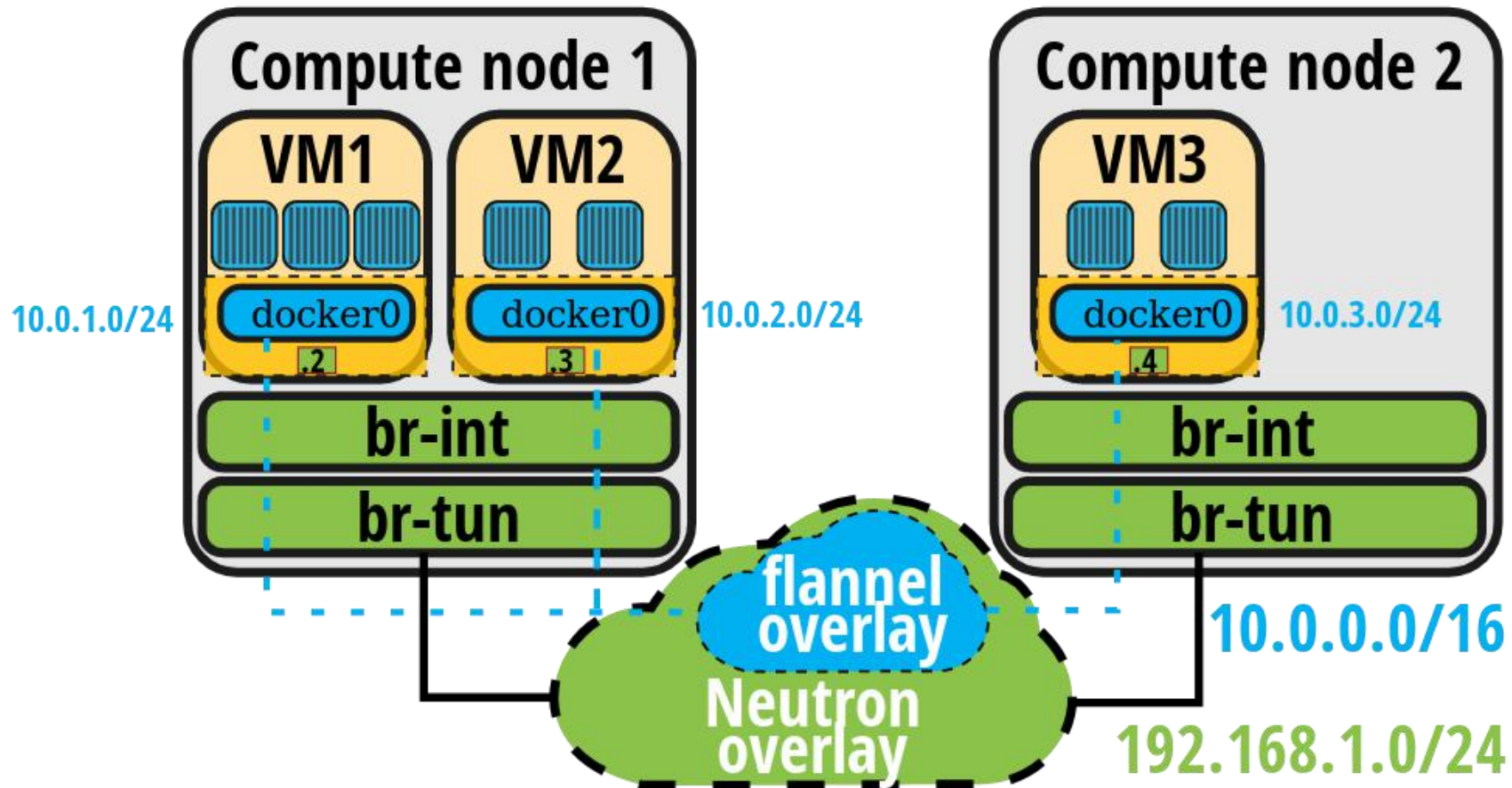
Kubernetes Networking



Motivation

- Use some of Neutron functionalities for Pods networking.
- Provide *interconnectivity* between OpenStack VMs and Kubernetes Pods and Services.
 - Pod->VM, VM->Pod
 - VM->Services
- Avoiding having multiple network overlays in Kubernetes on OpenStack deployments.
- Need for a smooth transition from VM into container based.

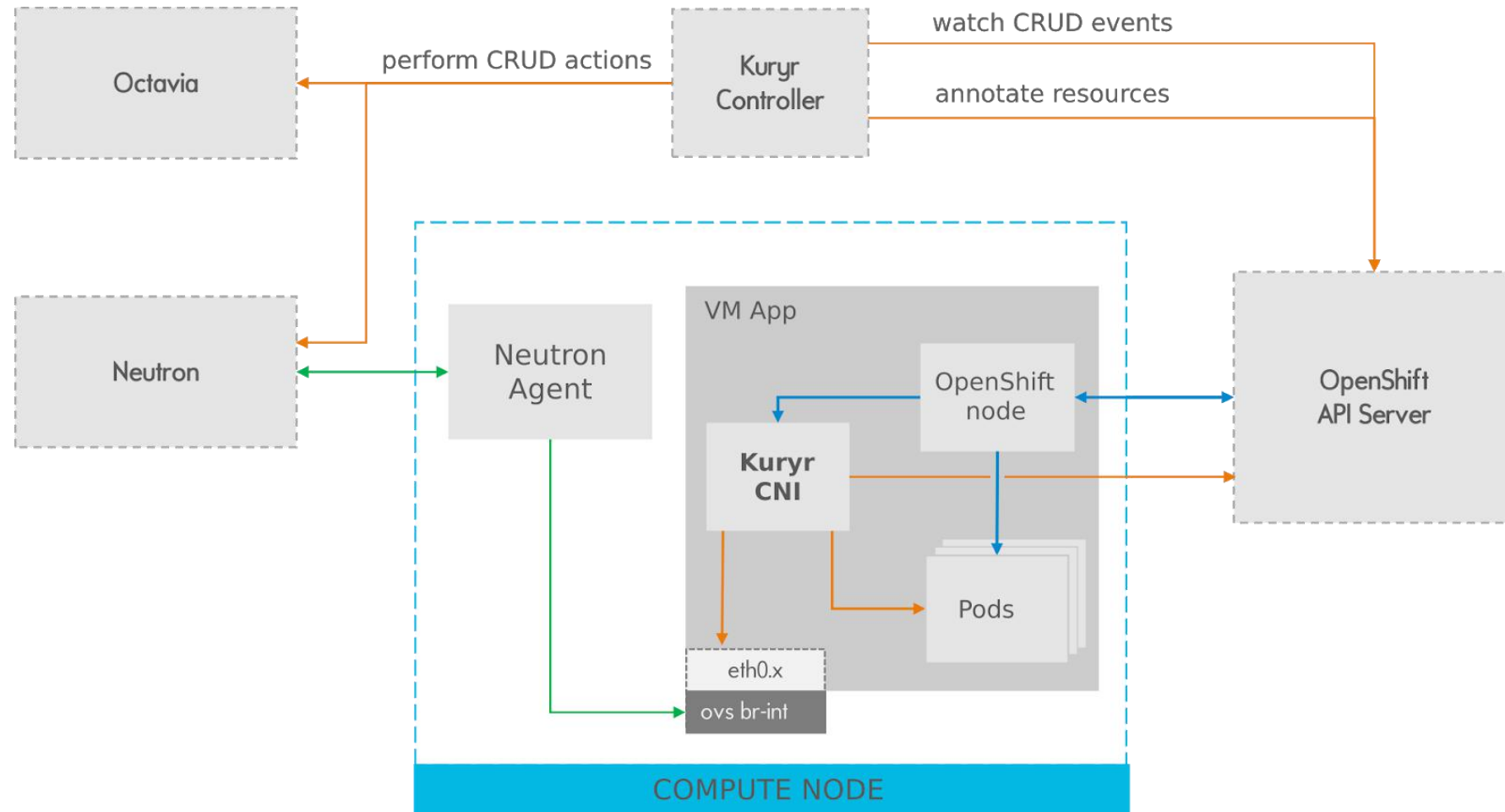
Motivation



Kuryr-component

- **kuryr-controller**, Responsible for OpenStack API operations.
- **kuryr-cni**, Executed by CNI, CNI driver is just a thin client that passes CNI ADD and DEL requests to kuryr-daemon instance via its HTTP API. It's simple Python executable that is supposed to be called by kublet's CNI.
- **kuryr-daemon**, is a service that should run on every Kubernetes node. It is responsible for watching pod events on the node it's running on, answering calls from CNI Driver and attaching VIFs when they are ready.

Kuryr-component

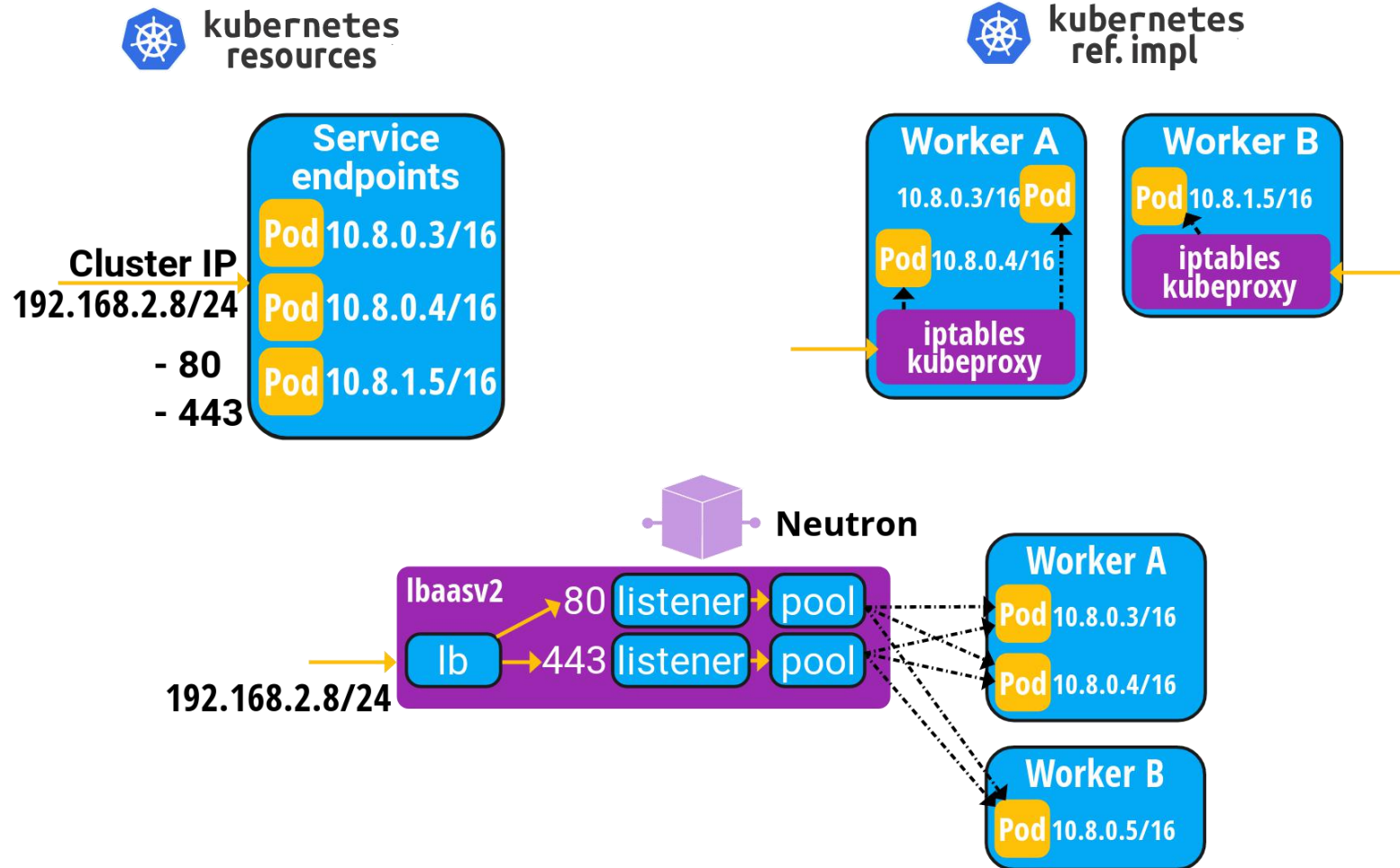


kuryr-kubernetes mapping

Kuryr-Kubernetes default handler for handling Kubernetes services and endpoints uses the OpenStack Neutron LBaaS API in order to have each service be implemented in the following way:

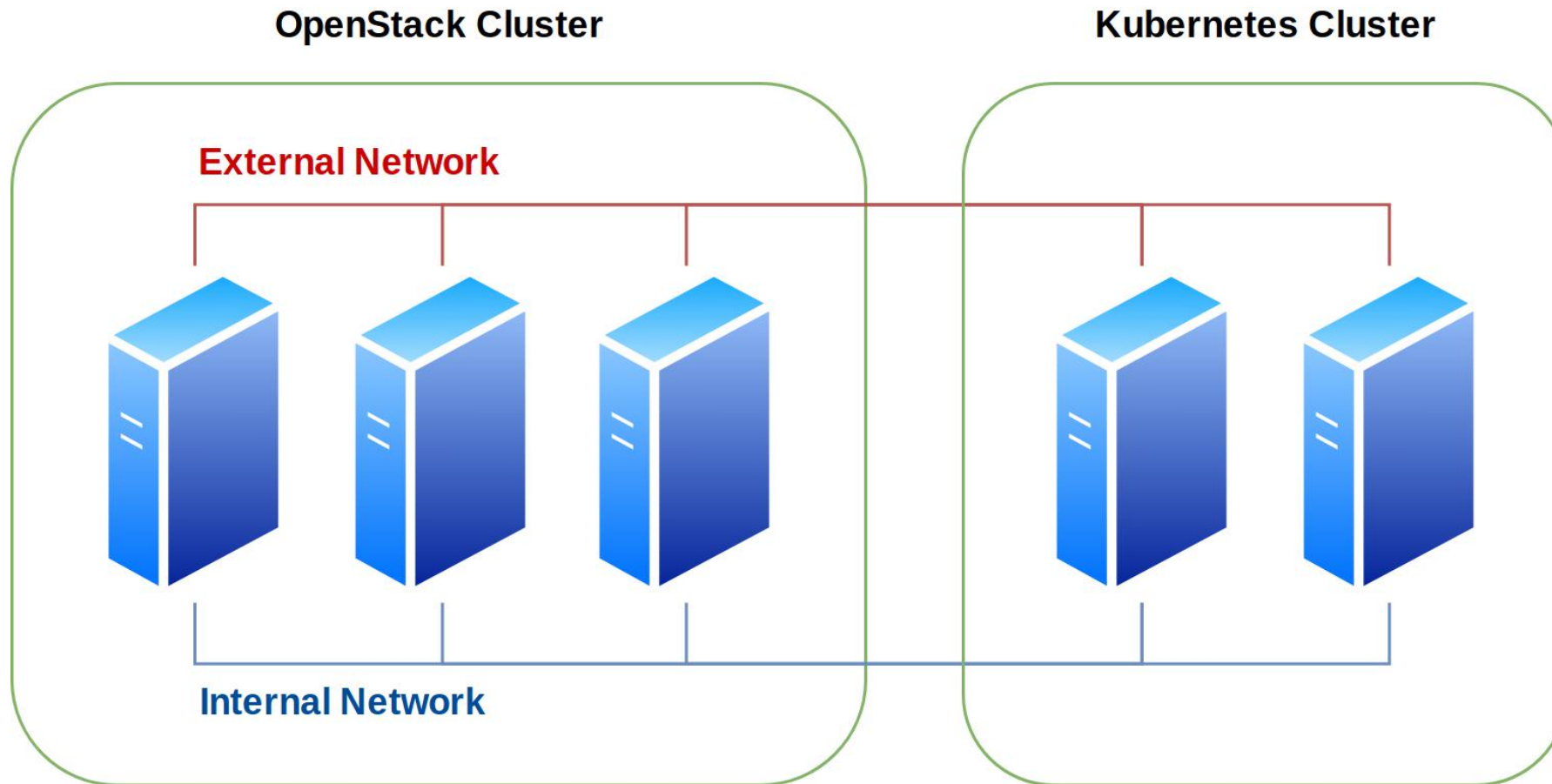
- **Service:** It is translated to a single LoadBalancer and as many Listeners and Pools as ports the Kubernetes Service spec defines.
- **ClusterIP:** It is translated to a LoadBalancer's VIP.
- **loadBalancerIP:** Translated to public IP associated with the LoadBalancer's VIP.
- **Endpoints:** The Endpoint object is translated to a LoadBalancer's VIP.

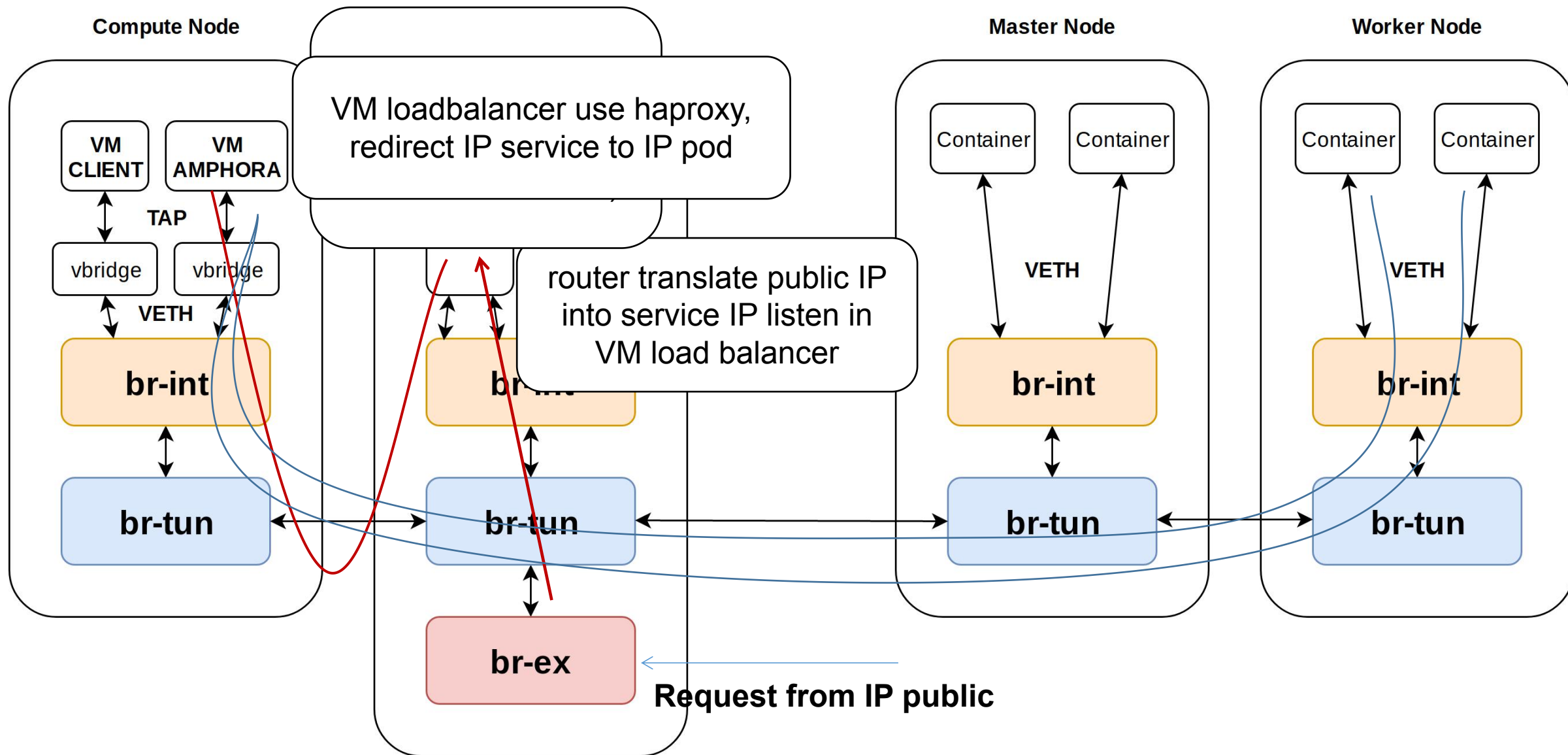
kuryr-kubernetes mapping



Kuryr-networking

for simplicity and trial error lab, I set up all node with two interfaces.





Any question?

