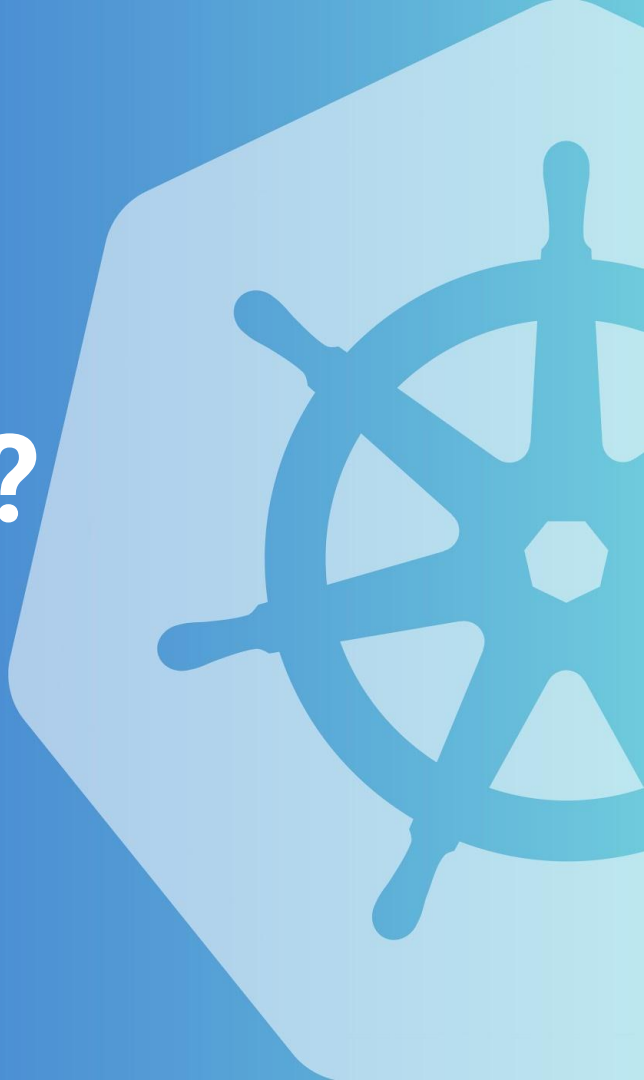
Fast Packet Processing with KubeVirt

Petr Horáček
Senior Software Engineer
Red Hat

David Vossel
Principal Software Engineer
Red Hat

kubernetes

What is KubeVirt?



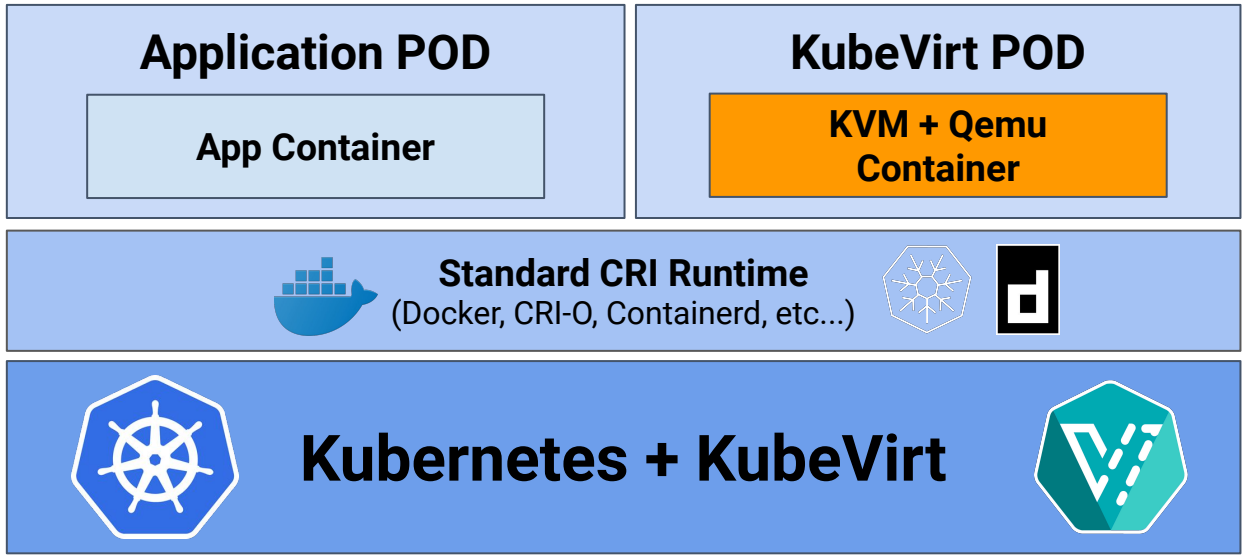
What is KubeVirt?

"KubeVirt is a Kubernetes extension that allows running traditional VM workloads natively side by side with Container workloads."

kubernetes

KubeVirt Basics

KubeVirt VM is a KVM+qemu process running inside a pod



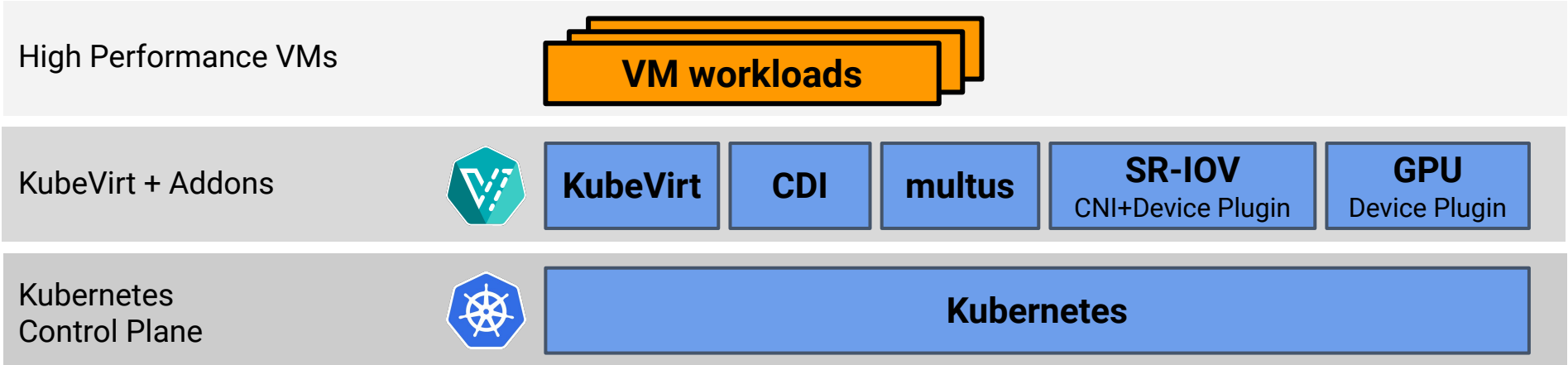
kubernetes

Scope of KubeVirt Project?

- Limited in scope to managing virtual machine lifecycle
  - Starting VMs
  - Stopping VMs
  - Pausing/suspending VMs
  - Live Migrating VMs
  - Monitoring VMs
  - Replicating VMs.
  - Etc…
- Utilizes existing cluster resources for everything else
  - PVCs for persistent virtual machine disks
  - CNI for network devices
  - CPU Manager for cpu affinity
  - Device Plugins for access to host devices (GPU, SR-IOV)

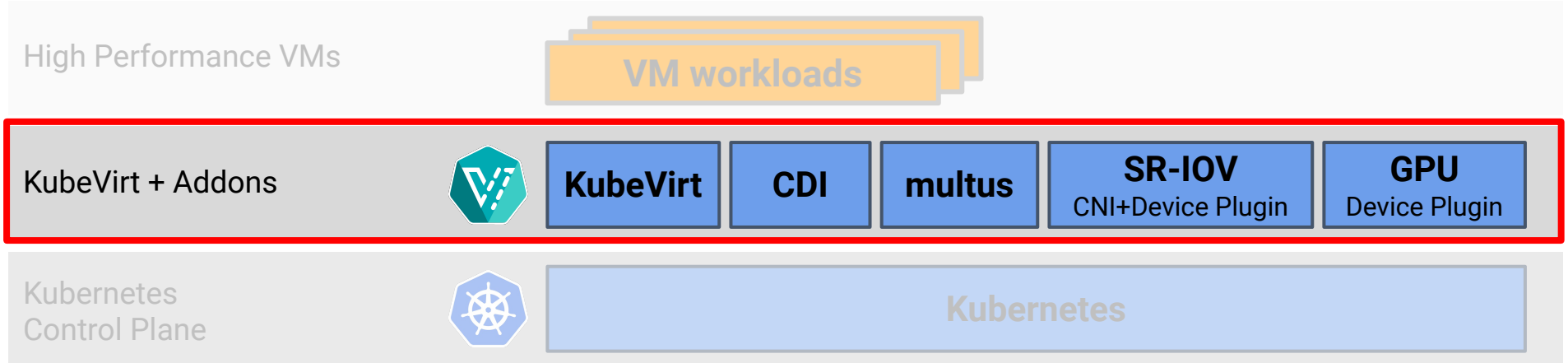
Addon Functionality

- Enable common workflows and functionality with addons
  - **CDI:** For building VM Image repositories using PVCs
  - **Multus + CNI Plugins**: For tieing VMs into multiple networks.
  - **SR-IOV CNI/Device Plugin:** for passing making SR-IOV devices available to VMs
  - **Nvidia GPU Device Plugin:** For passing GPU devices to VMs.



What is an Opinionated Install?

Opinionated Install: KubeVirt + optional addon components + configuration data



Opinionated Install for Traditional VMs

- Opinionated Install Addons for…
  - Importing VMs from legacy VM management Platforms (Like VMWare and oVirt)
  - Providing feature parity with legacy VM management platforms
- Problem…
  - Complex collection of [Kubevirt + Addons + Config data] is difficult to manage.
- Solution?

Hyper Converged Operator (HCO)

- HCO is an **operator of operators** or **meta operator**
  - Coordinates installing KubeVirt + addons
  - Allows installing/updating entire opinionated install as a **single cohesive unit.**

kubernetes

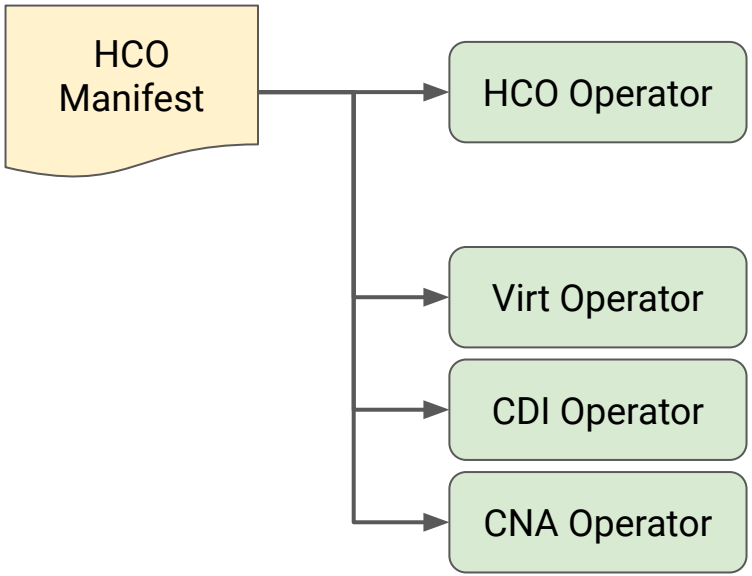
HCO Architecture

HCO Manifest Deploys HCO operator + Subcomponent operators

kubernetes

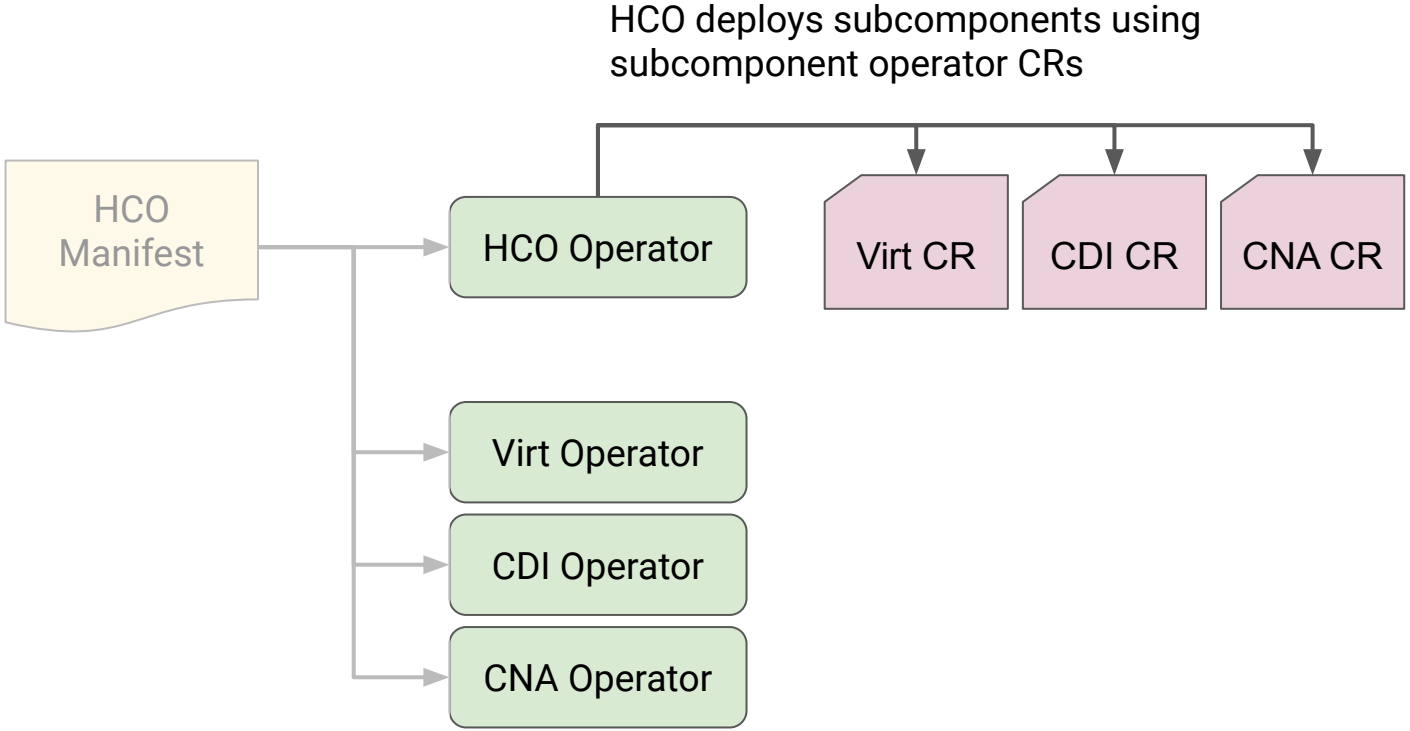
HCO Architecture

HCO Manifest Deploys HCO operator + Subcomponent operators



kubernetes

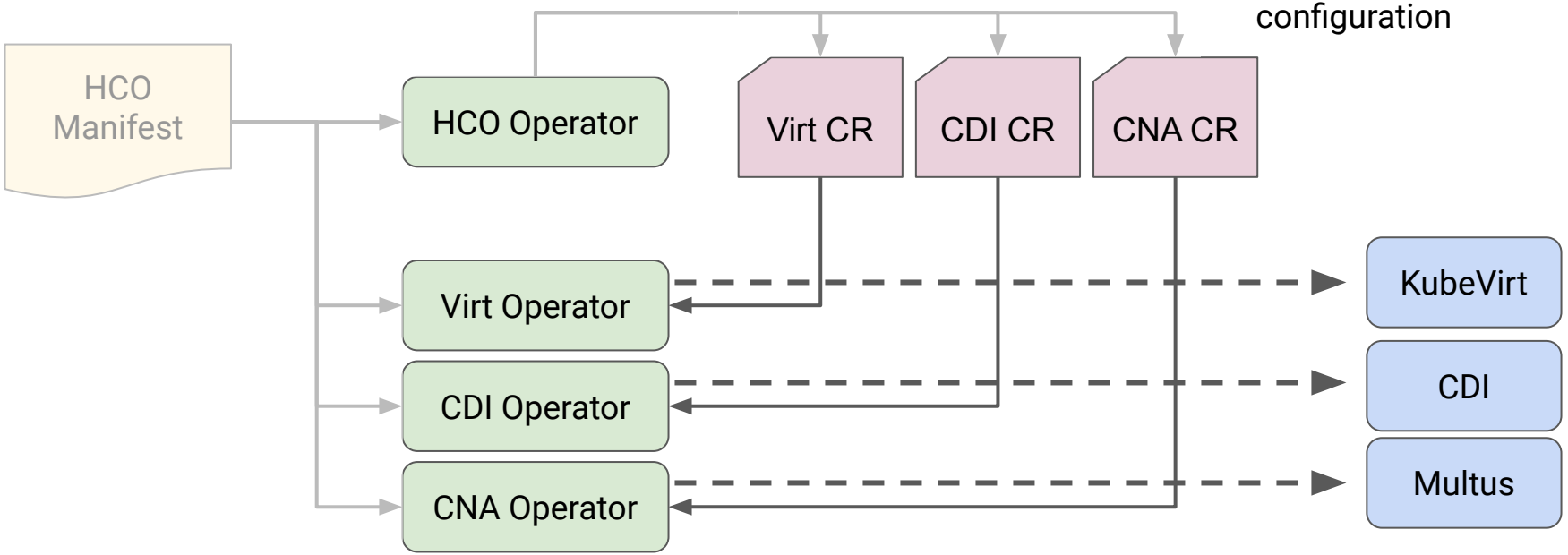
HCO Architecture



kubernetes

HCO Architecture

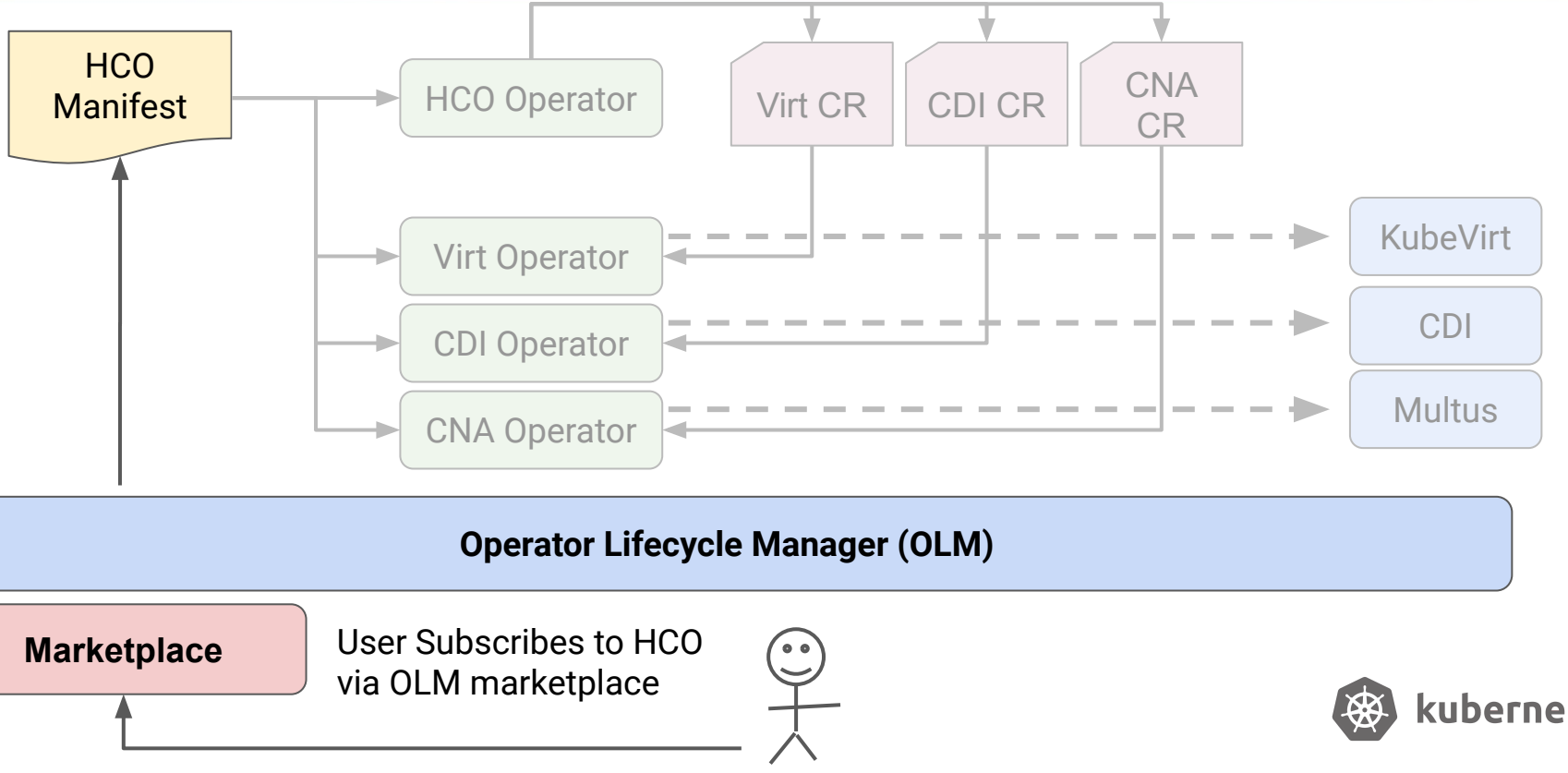
Subcomponent Operators deploy the subcomponent using the HCO's supplied configuration



Current HCO Subcomponents

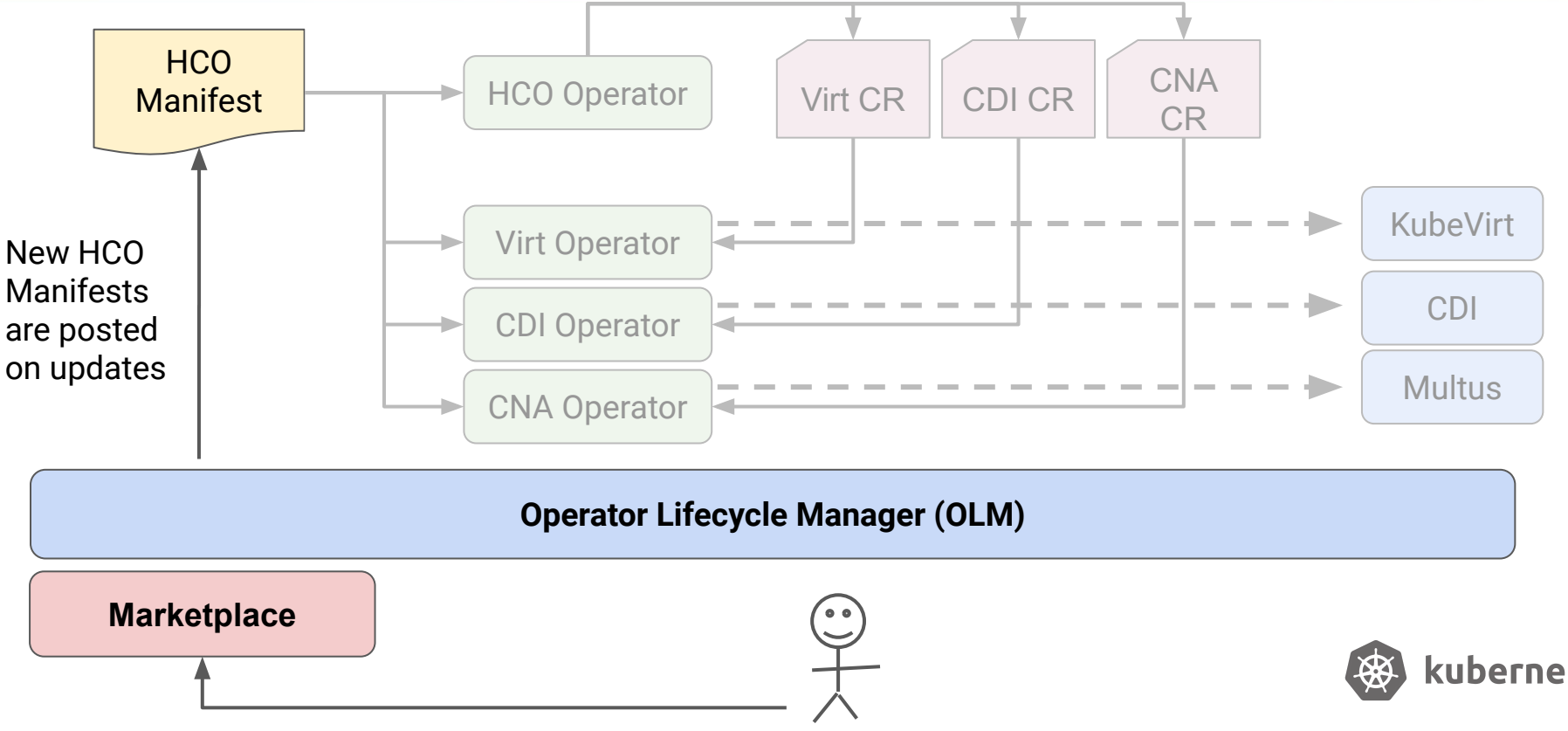
- **Virt-operator** - KubeVirt Core
- **Cluster Network Addons (CNA)** - multus, mac pool, CNI plugins.
- **SSP operator** - node feature labeller, common vm templates
- **CDI operator** - VM image importing and cloning
- **Node maintenance operator** - server side node maintenance (similar to kubectl drain but server side)
- **Hostpath Provisioner Operator** - local storage storage class provisioner
- **VM Import Operator** - workflows for importing rhv/ovirt vm to kubevirt

Installing HCO from OLM Marketplace



kubernetes

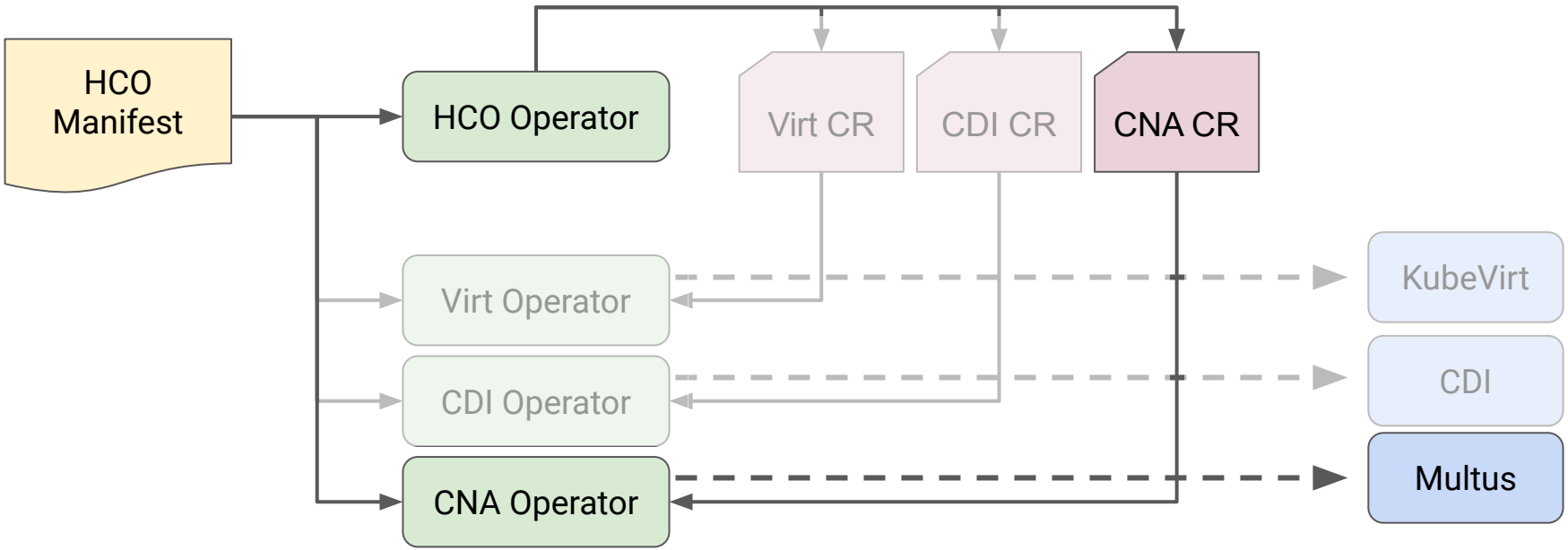
Updating from OLM Marketplace



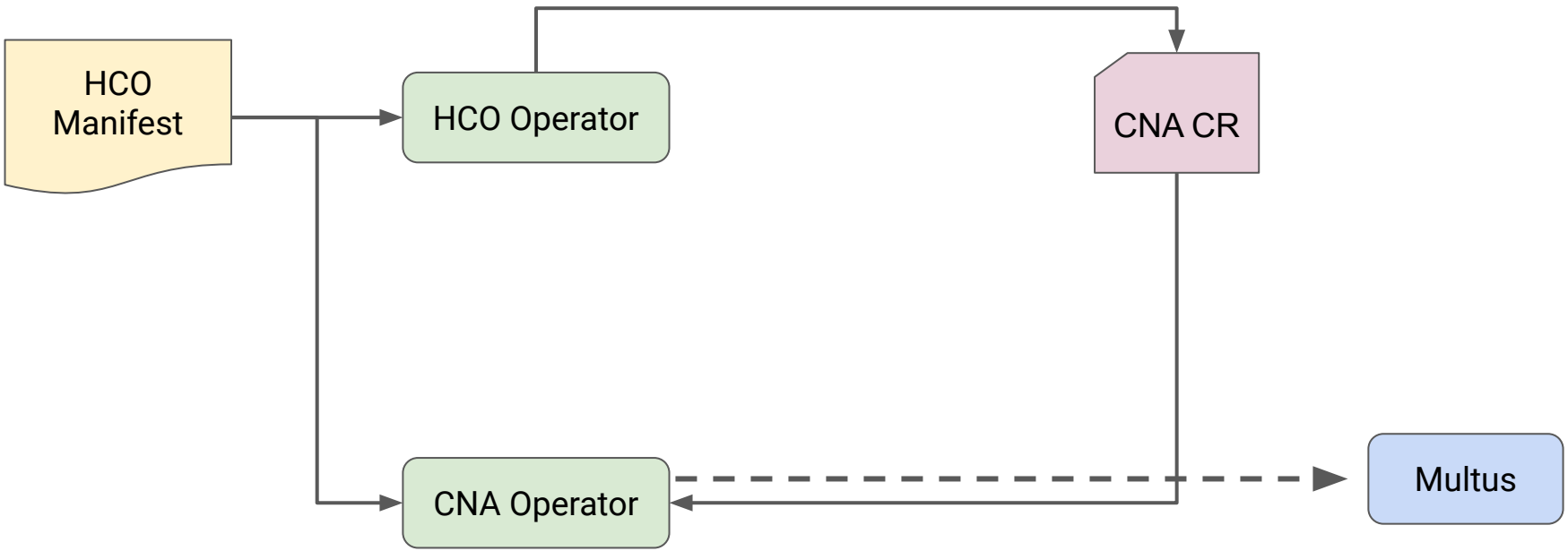
kubernetes

Network Addons

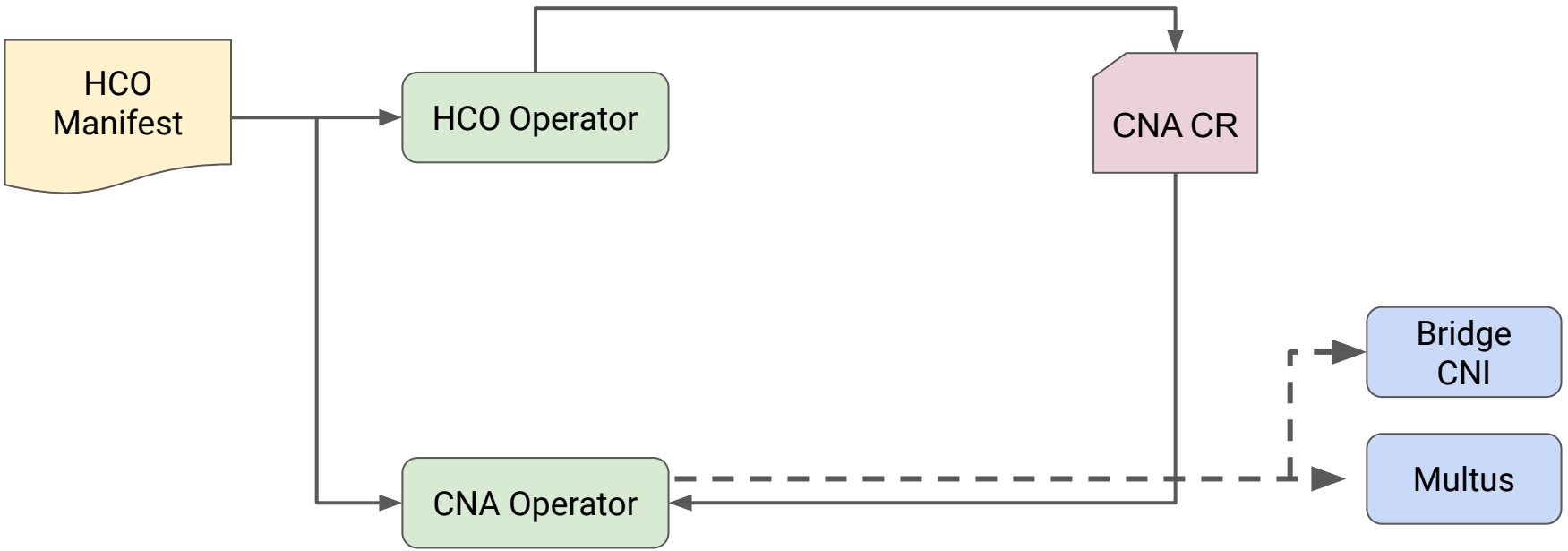
Network Addons



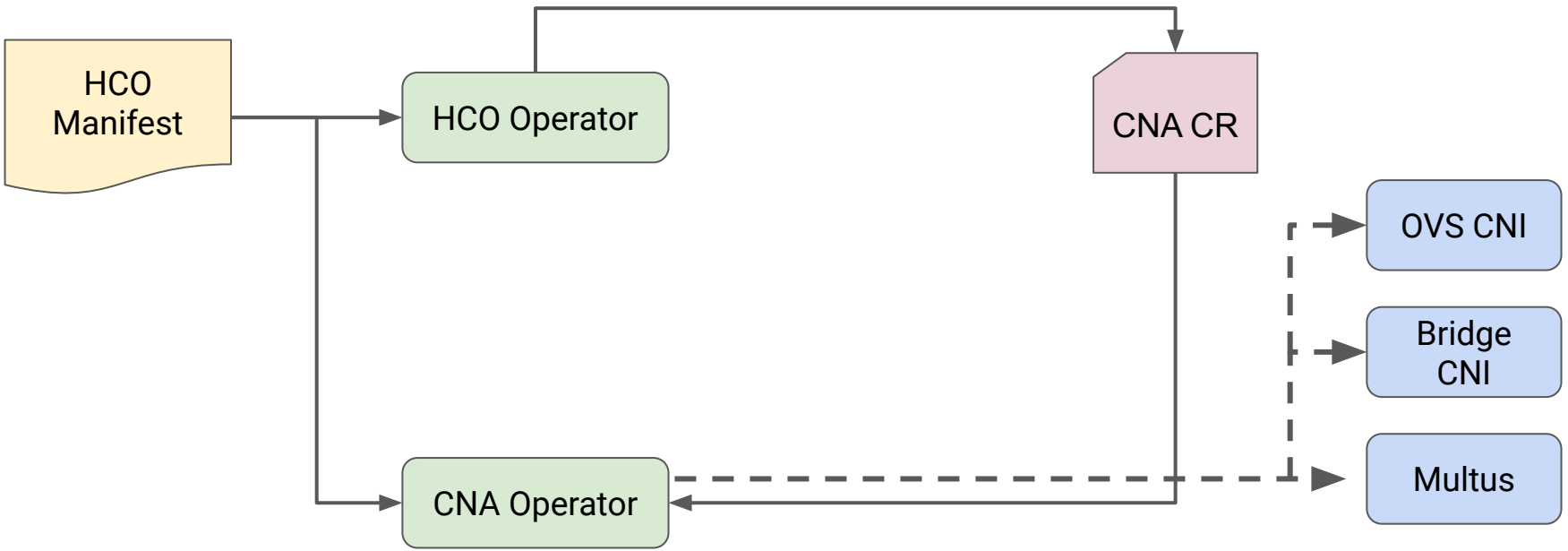
Network Addons



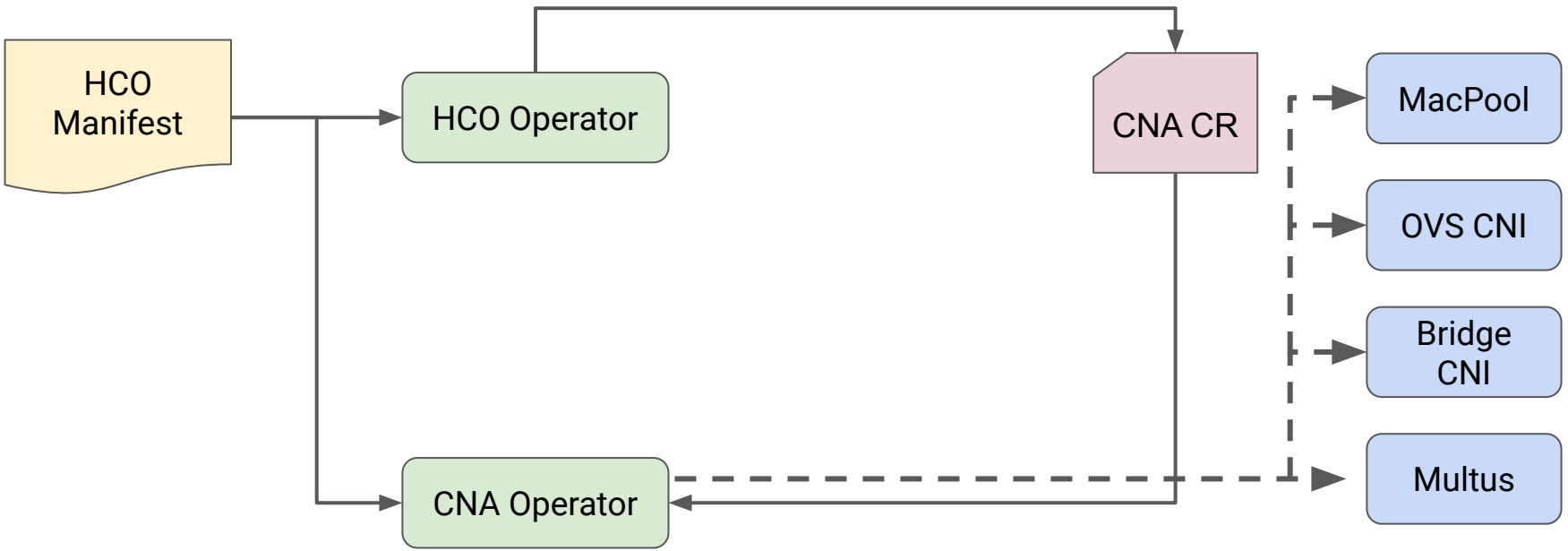
Network Addons



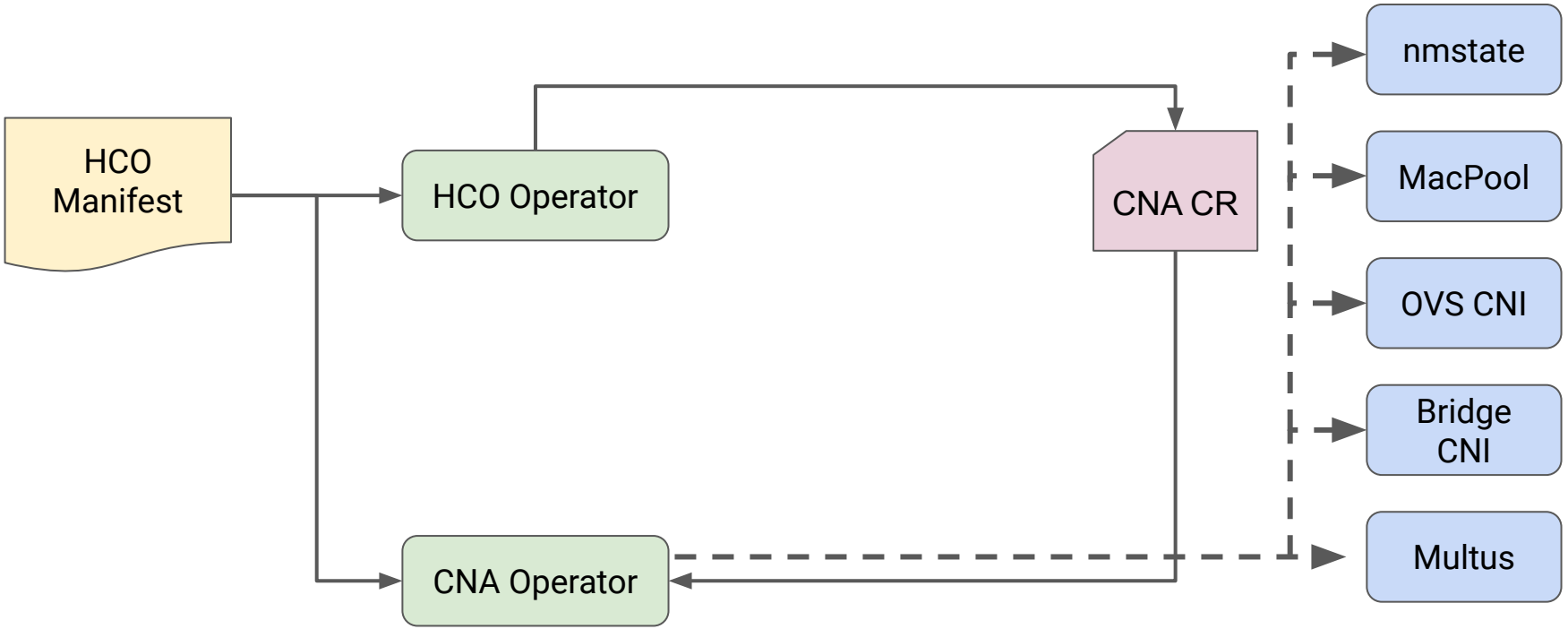
Network Addons



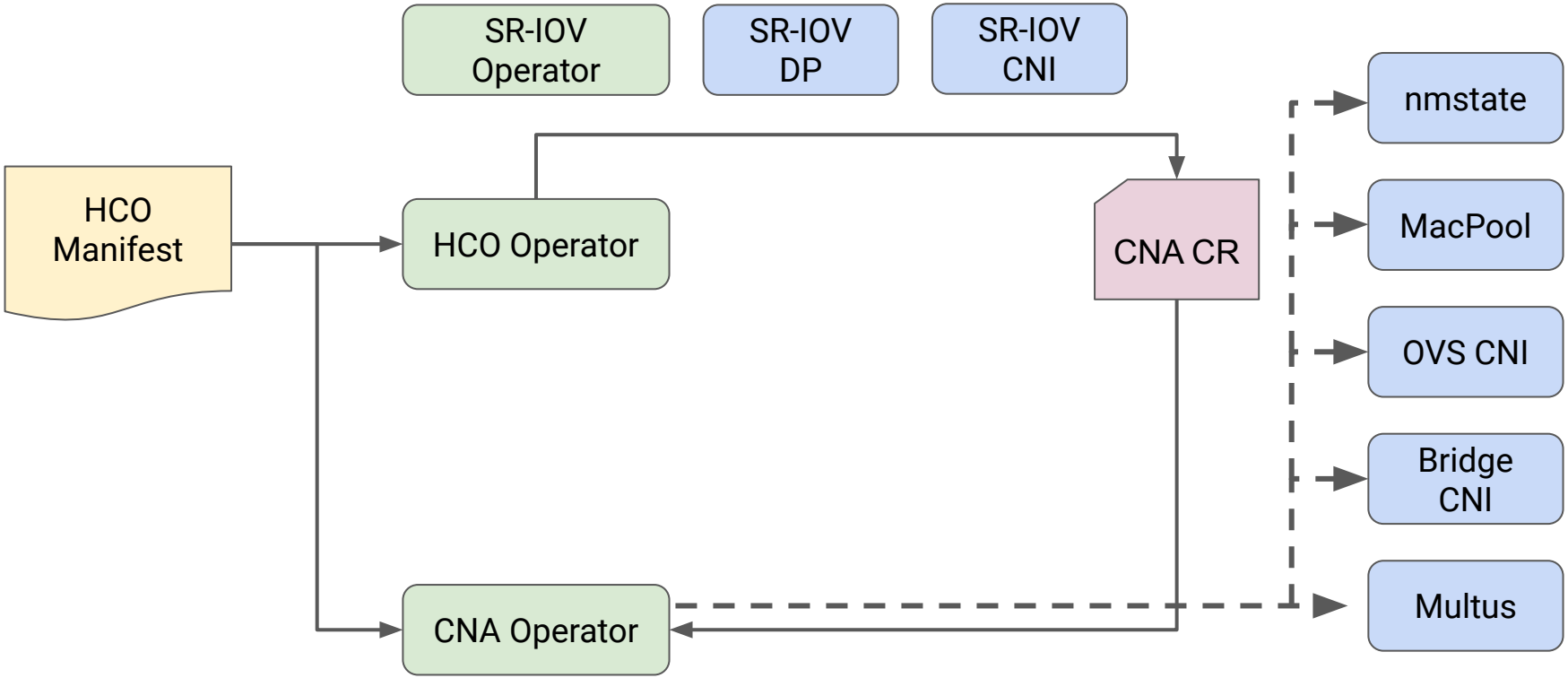
Network Addons



Network Addons



Network Addons



Regular Packet Processing

Regular Packet Processing

- The default CNI plugin of the cluster,
- often based on an overlay network,
- connected to the VM container,
- forwarded to the VM process through NAT or Linux bridge.

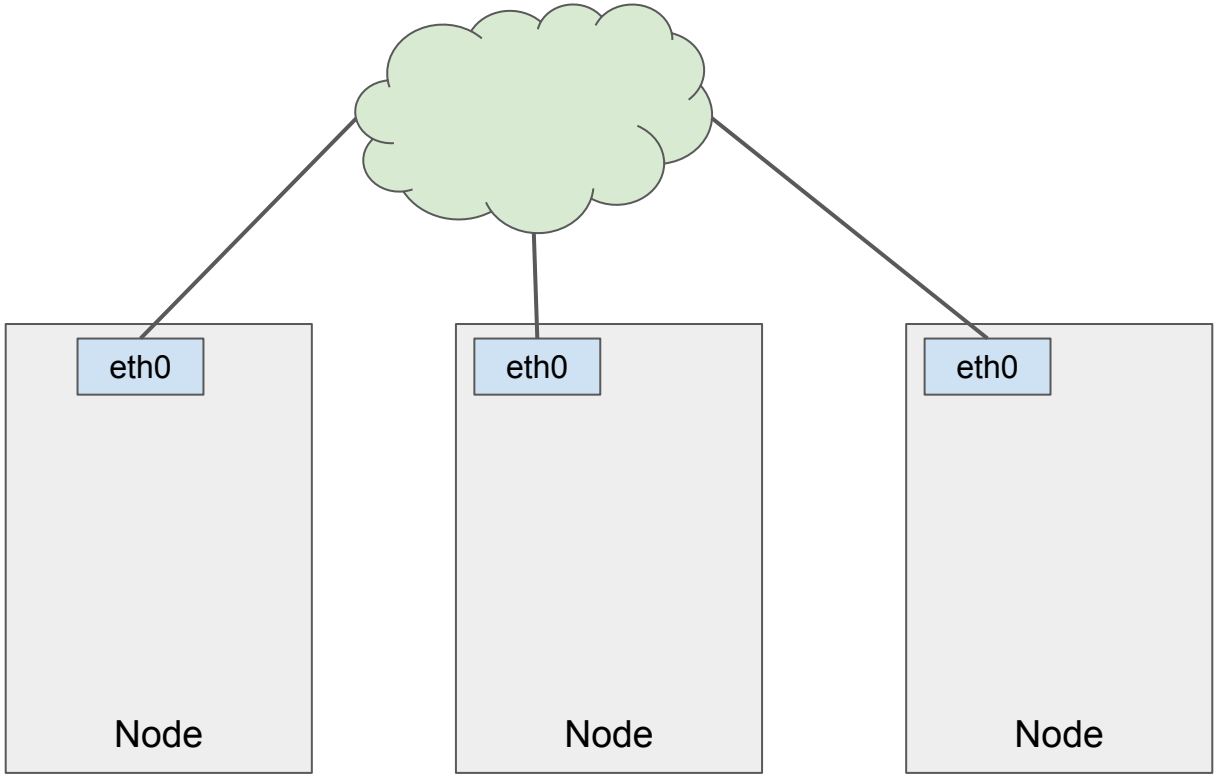
Regular Packet Processing

- The default CNI plugin of the cluster,
- often based on an overlay network,
- connected to the VM container,
- forwarded to the VM process through NAT or Linux bridge.

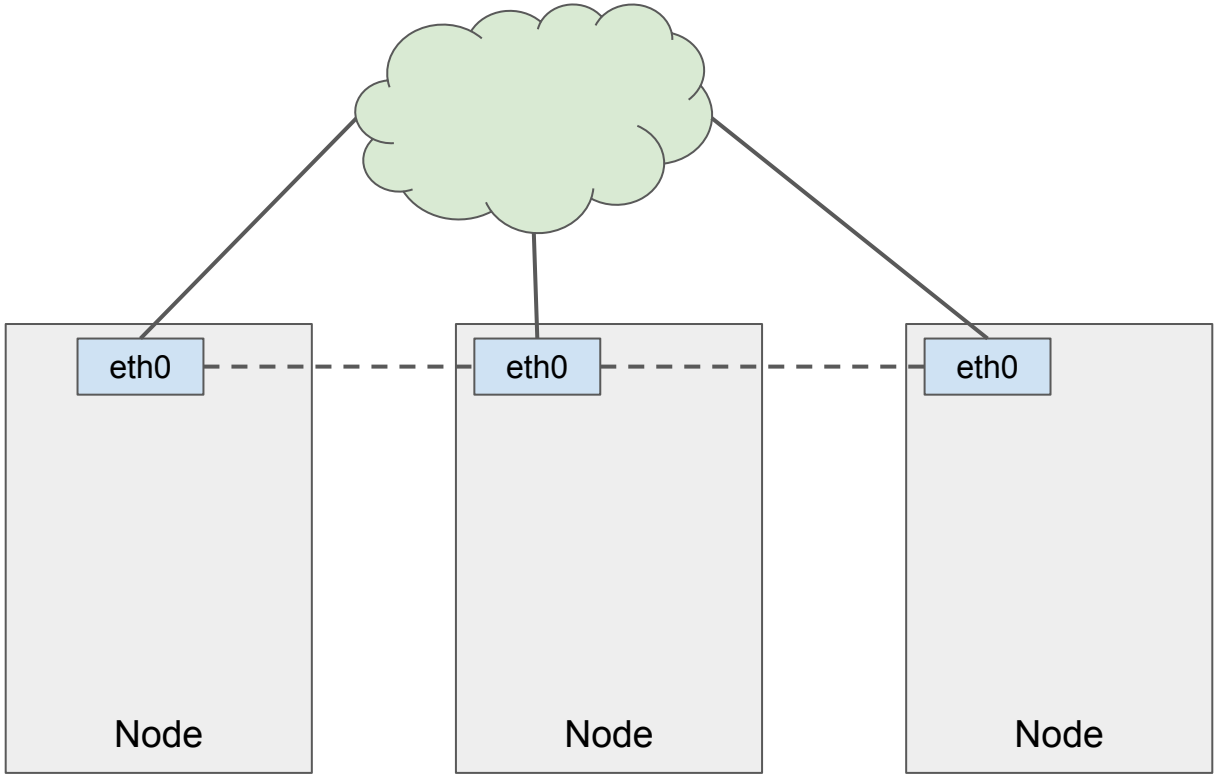
- Offers all the benefits of Kubernetes infrastructure,
- is not as fast as other solutions.

Regular Packet Processing (Cluster Network)

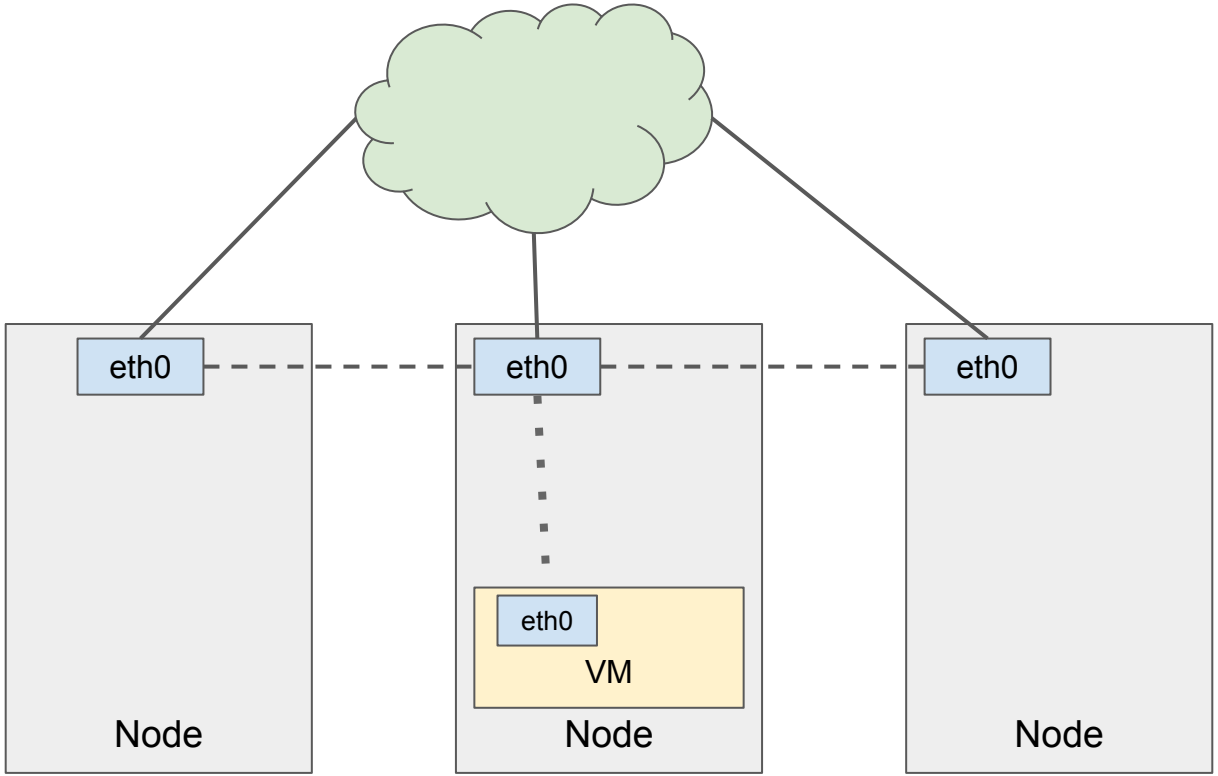
Regular Packet Processing (Cluster Network)



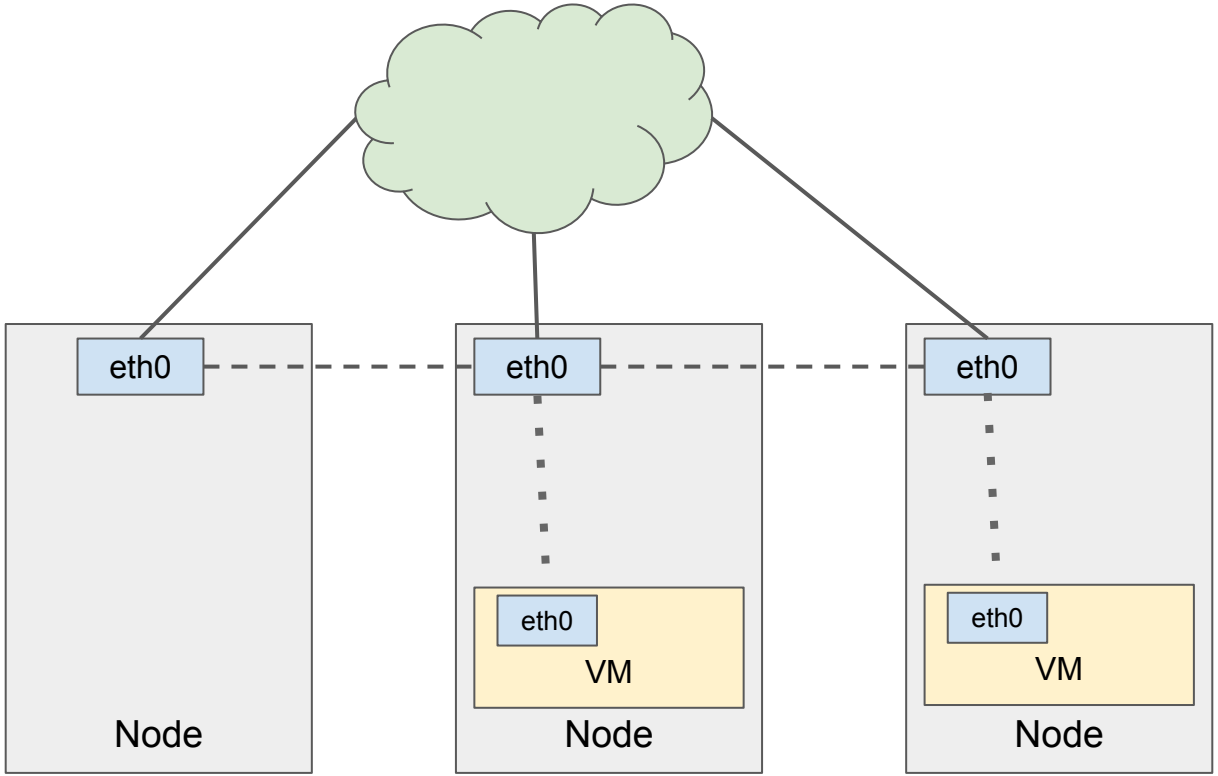
Regular Packet Processing (Cluster Network)



Regular Packet Processing (Cluster Network)



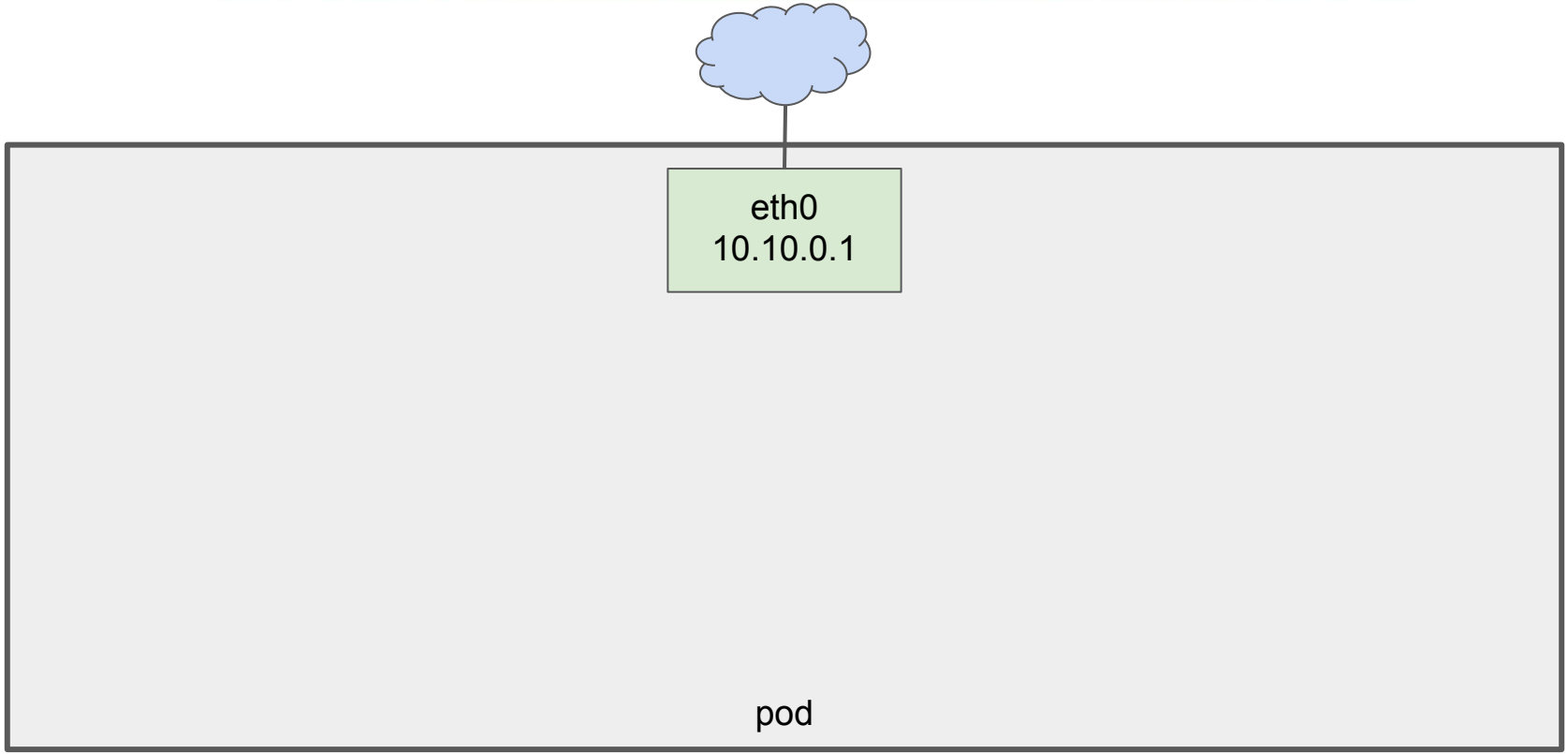
Regular Packet Processing (Cluster Network)



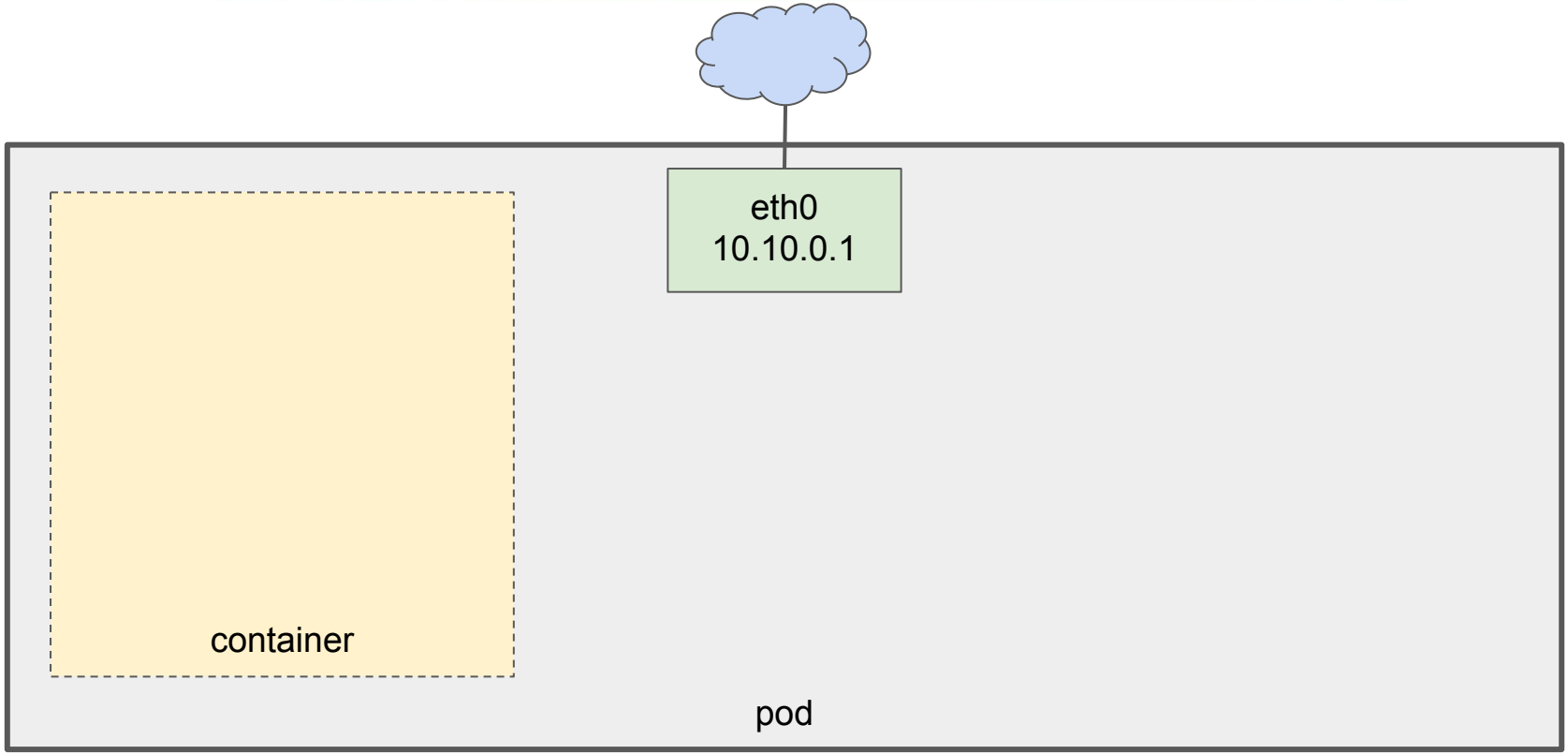
Regular Packet Processing (Binding)

Regular Packet Processing (Binding)

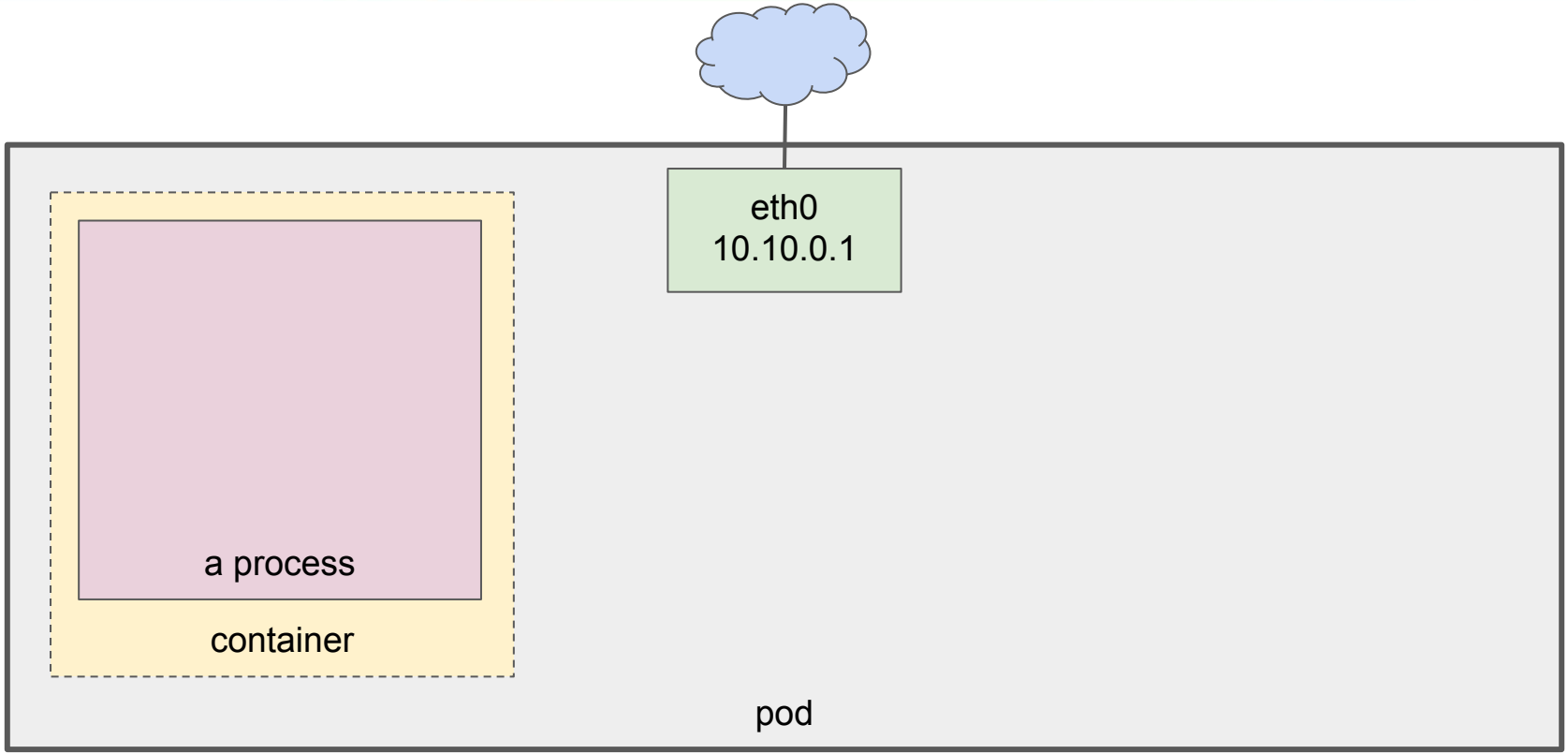
Regular Packet Processing (Binding)



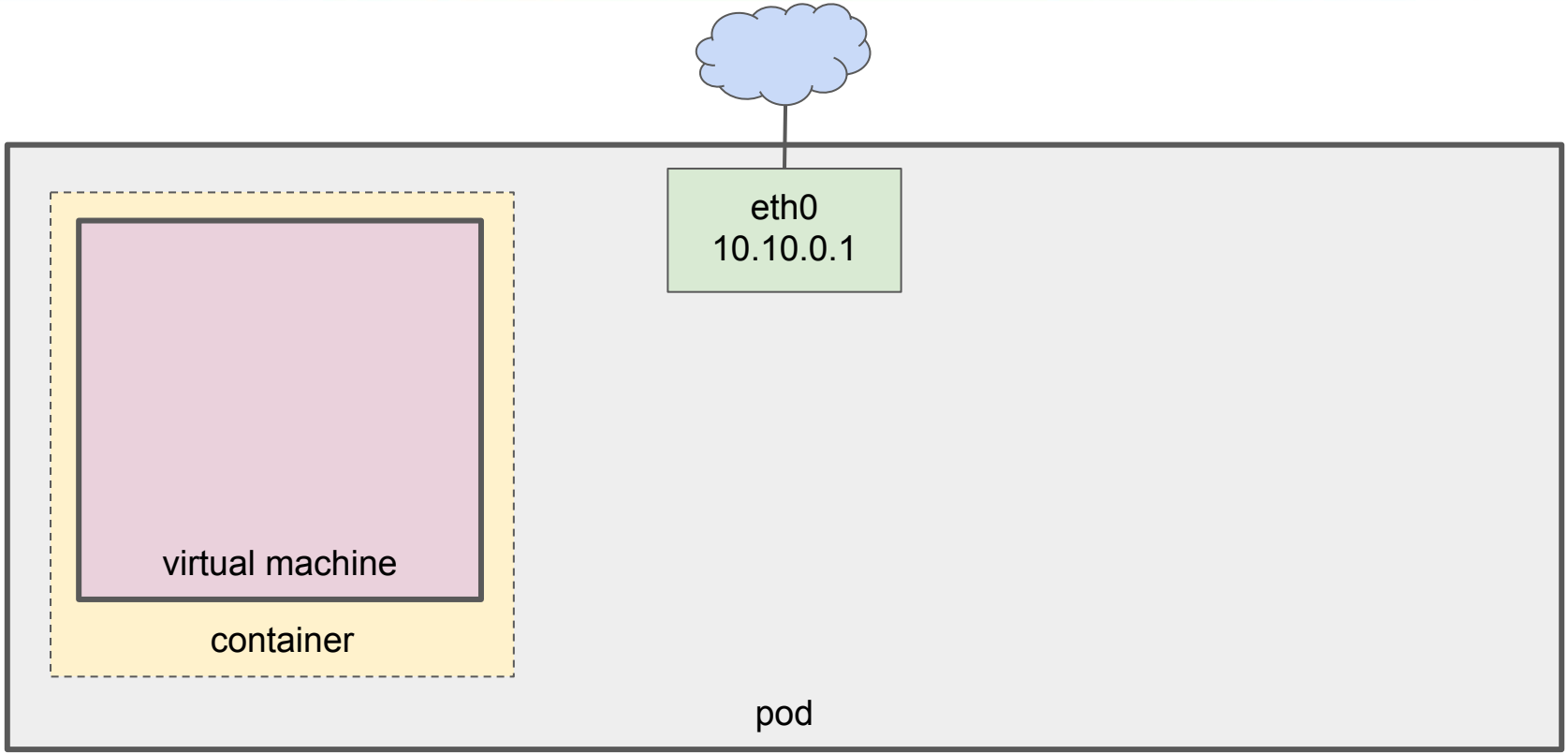
Regular Packet Processing (Binding)



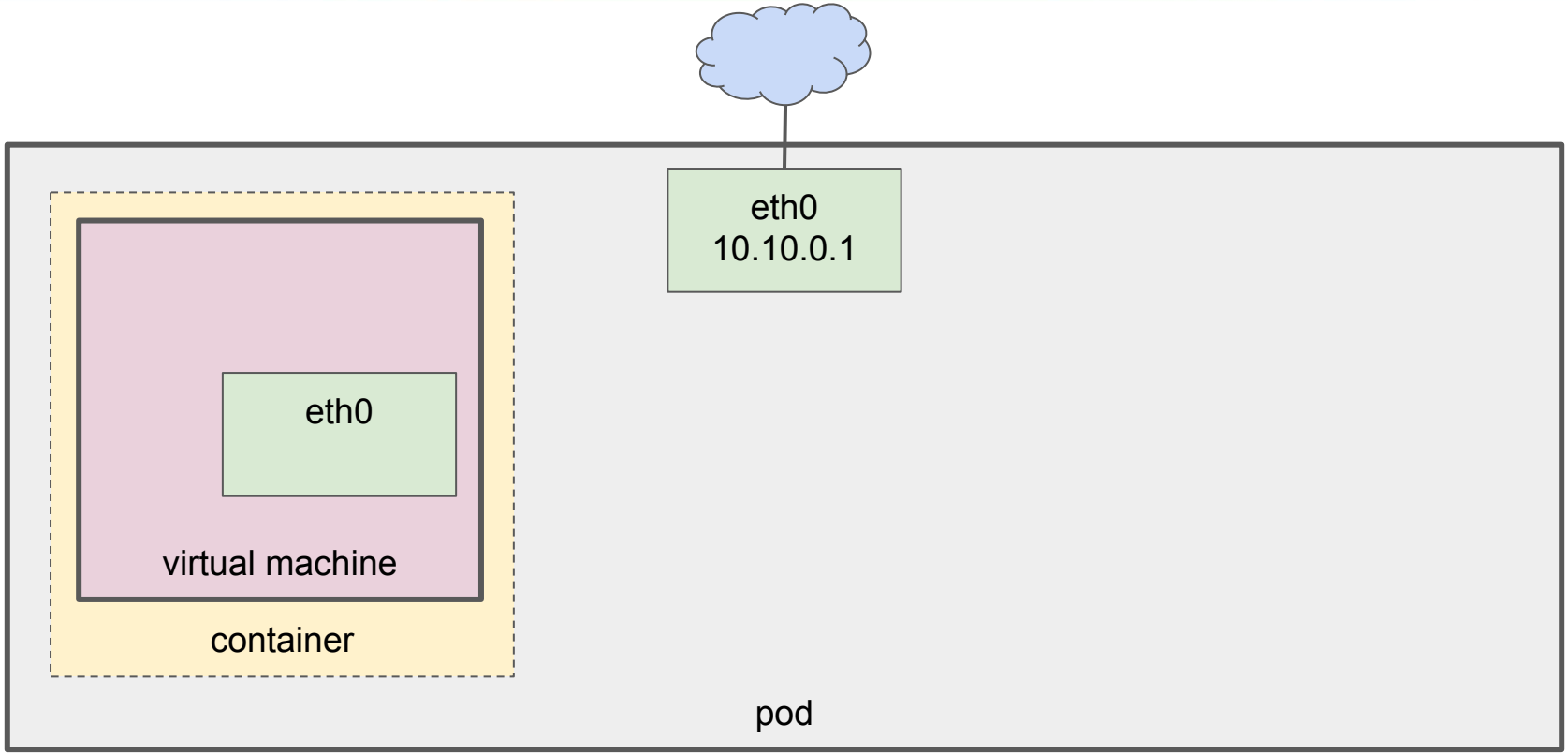
Regular Packet Processing (Binding)



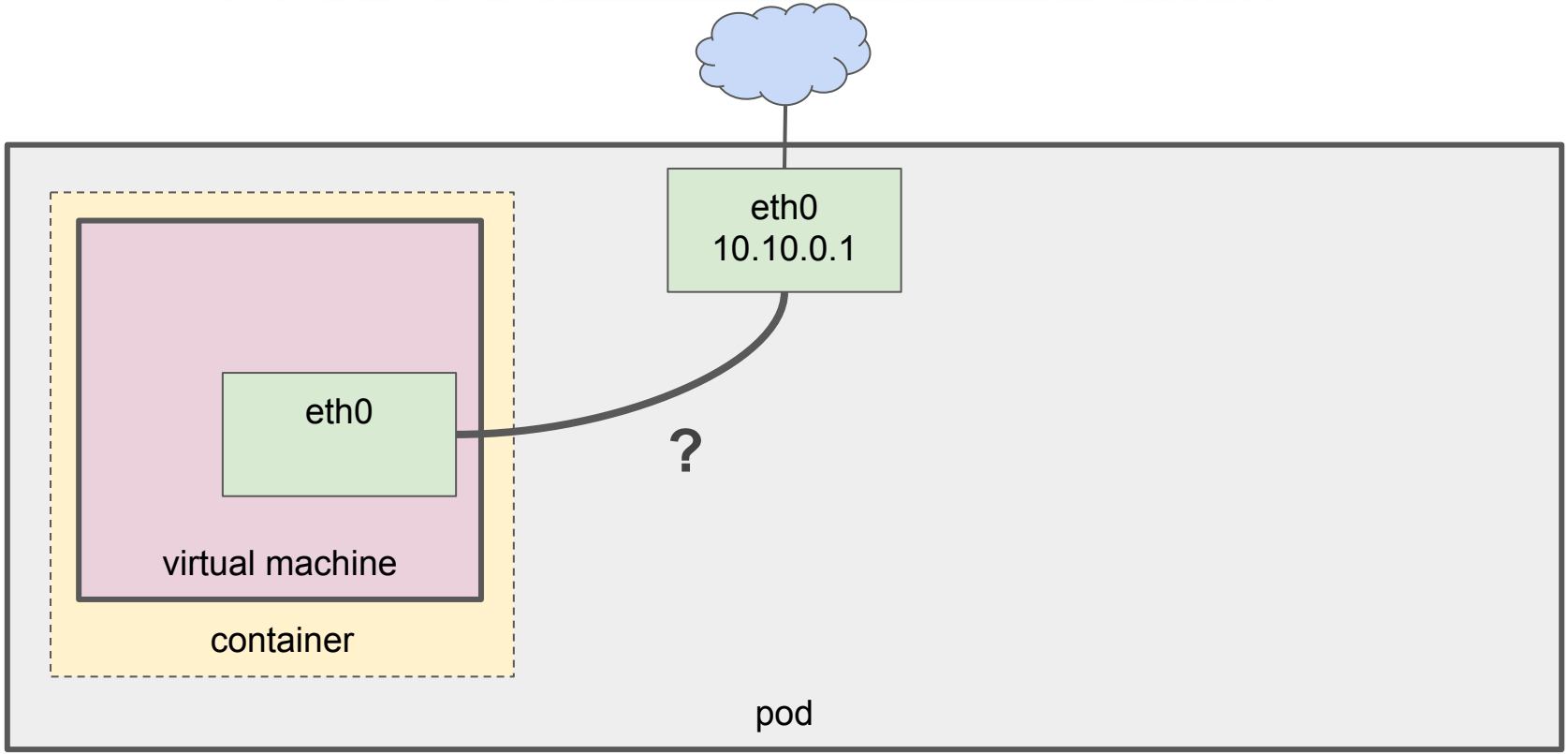
Regular Packet Processing (Binding)



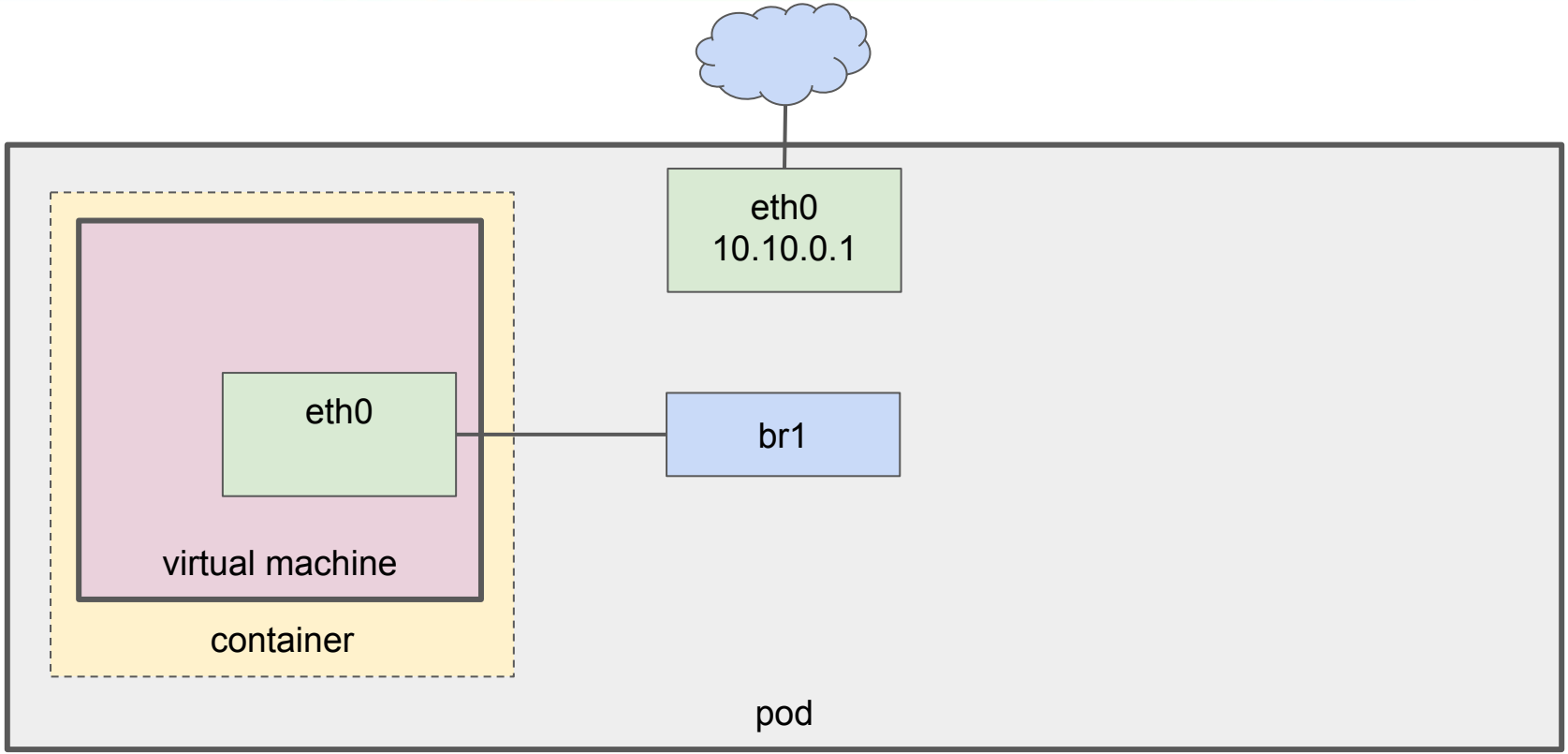
Regular Packet Processing (Binding)



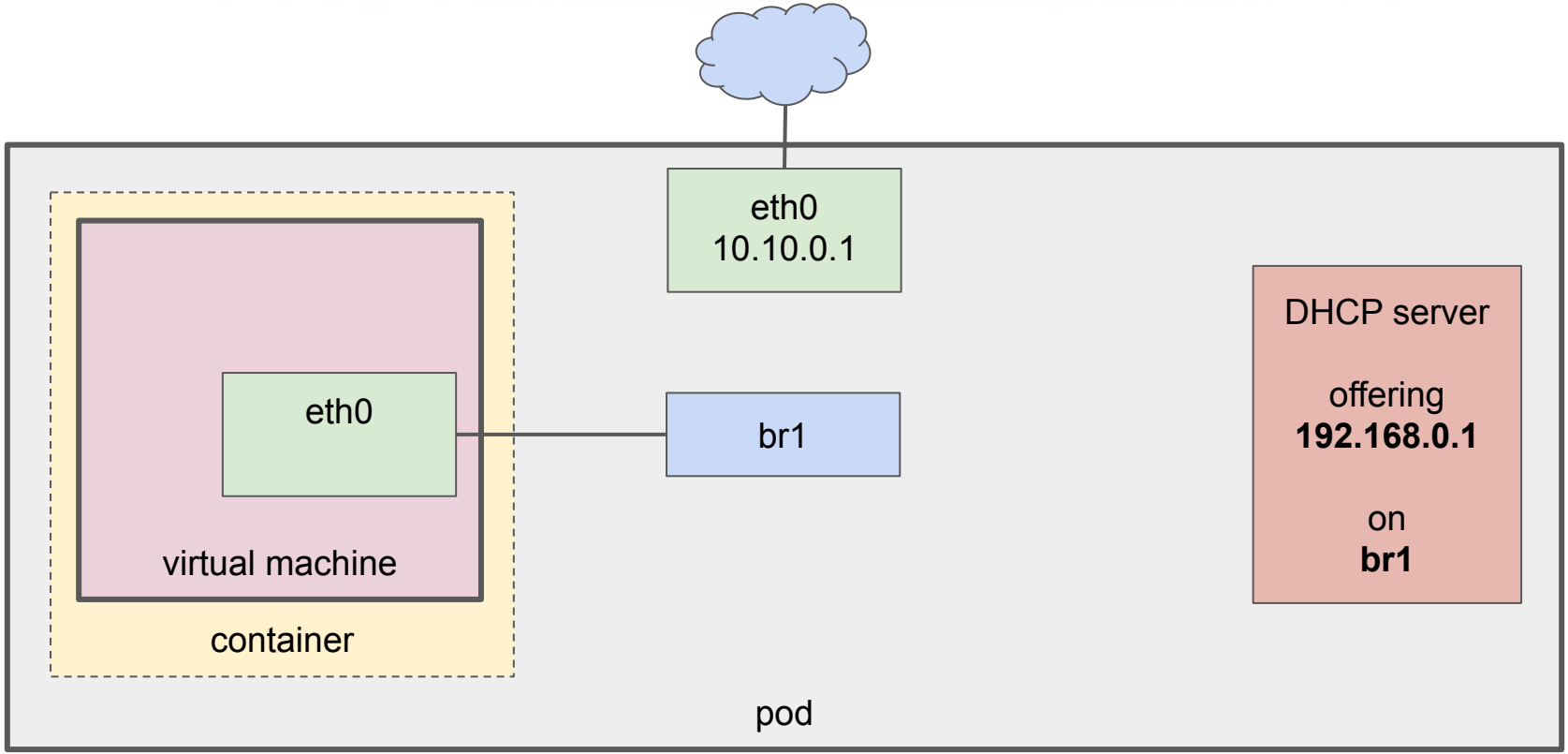
Regular Packet Processing (Binding)



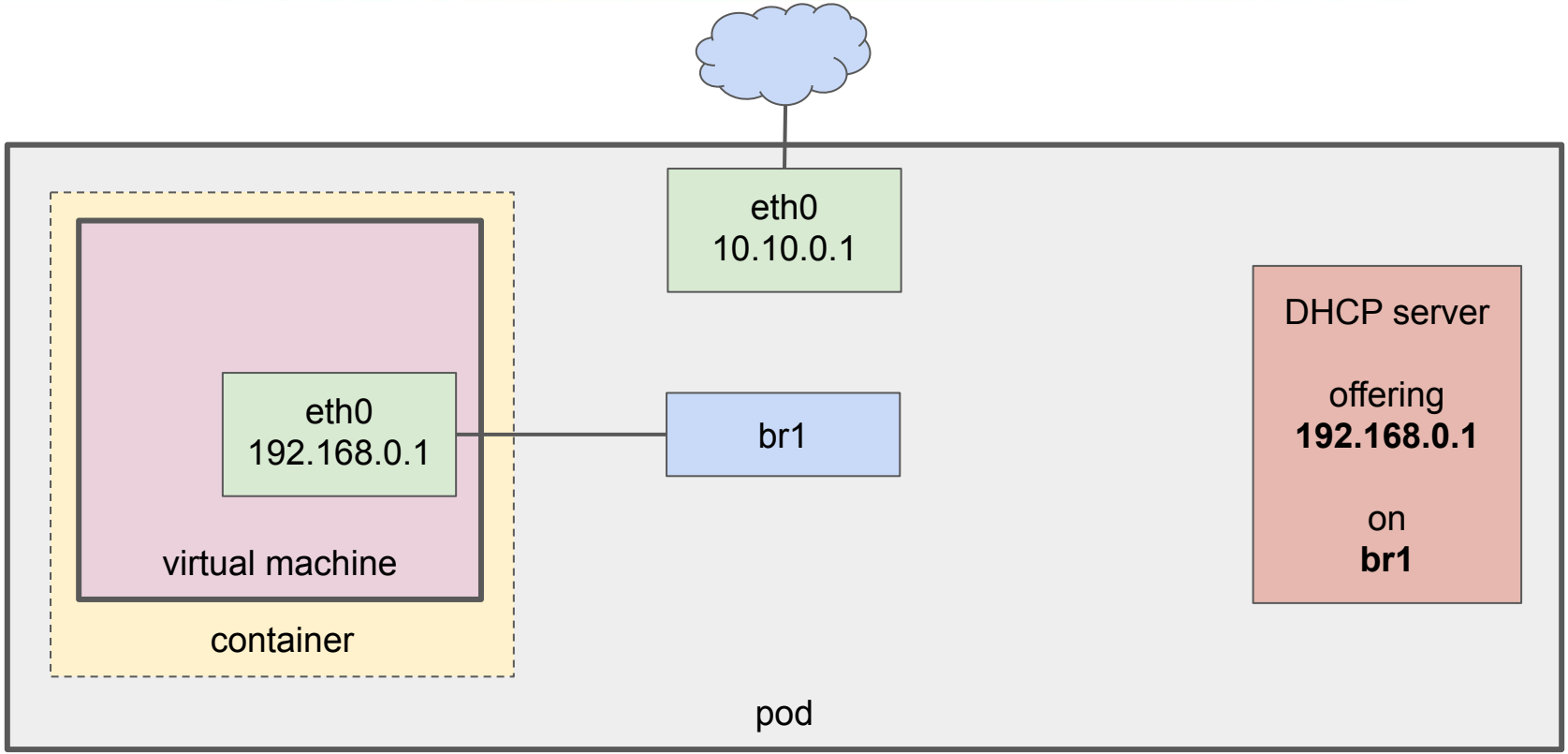
Regular Packet Processing (Binding)



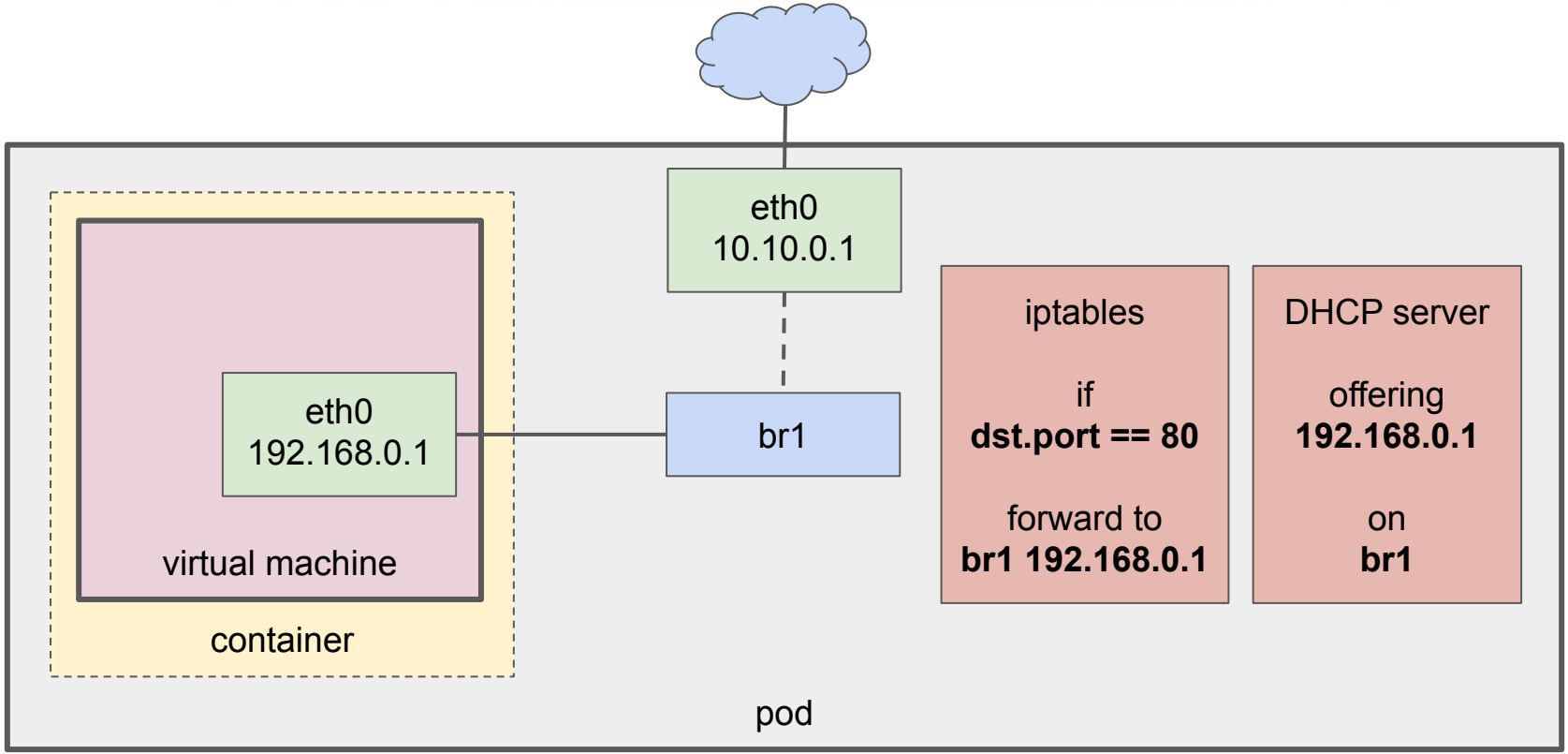
Regular Packet Processing (Binding)



Regular Packet Processing (Binding)



Regular Packet Processing (Binding)



Regular Packet Processing (Example)

```
apiVersion: kubevirt.io/v1alpha3
kind: VirtualMachineInstance
metadata:
  name: vmi-test
  ...
```

```
spec:
  networks:
  - name: default
    pod: {}
  domain:
    devices:
      ...
      interfaces:
      - name: default
        masquerade: {}
    ...
...
```

Faster Packet Processing

Faster Packet Processing

- Bridge configured on the host,
- with host NIC as its port,
- extending L2 access to the network to containers,
- as an additional network,
- forwarded to the VM through another bridge.

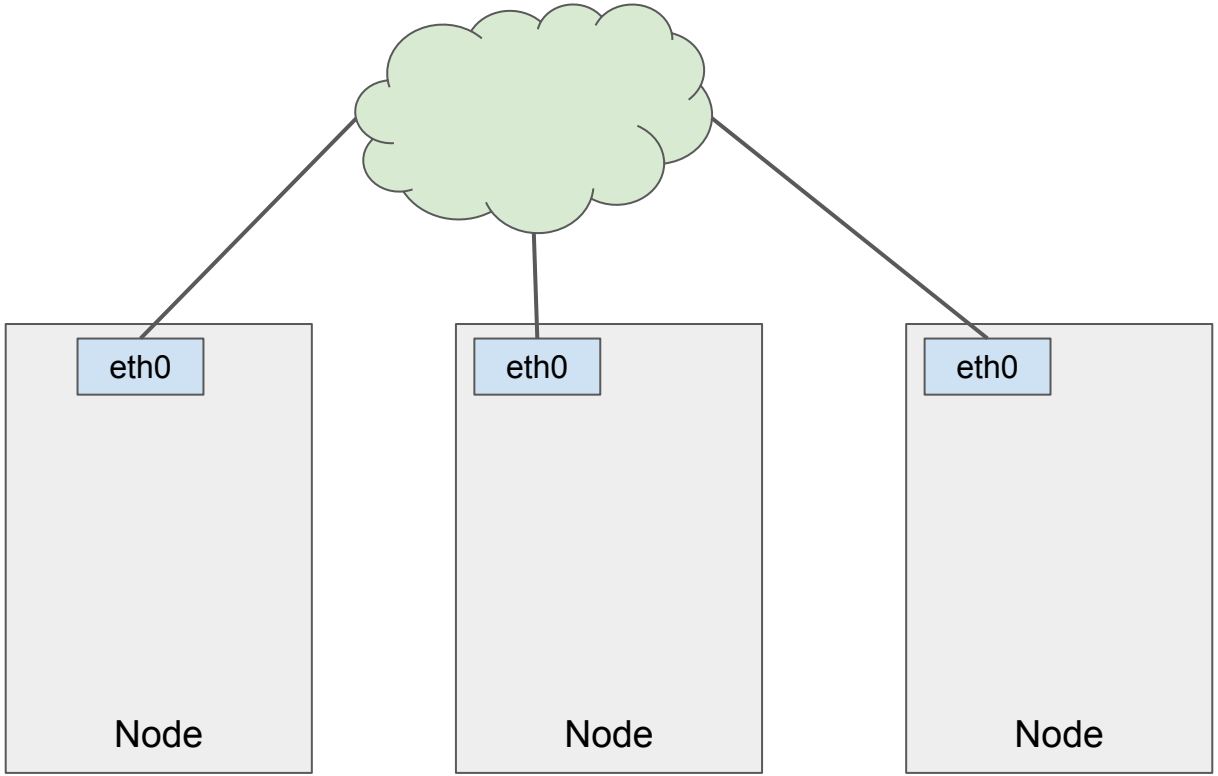
Faster Packet Processing

- Bridge configured on the host,
- with host NIC as its port,
- extending L2 access to the network to containers,
- as an additional network,
- forwarded to the VM through another bridge.

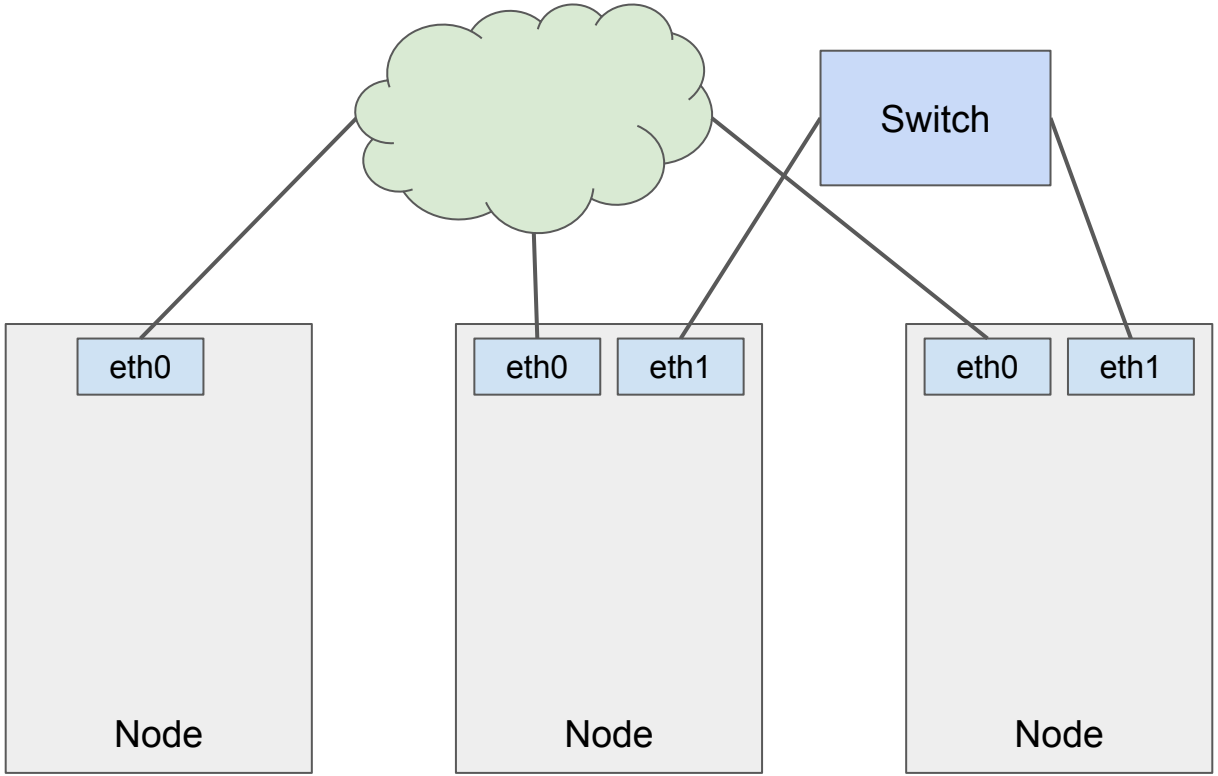
- May be faster thanks to direct L2 access,
- is not 100% integrated to Kubernetes networking components.

Faster Packet Processing (Cluster Network)

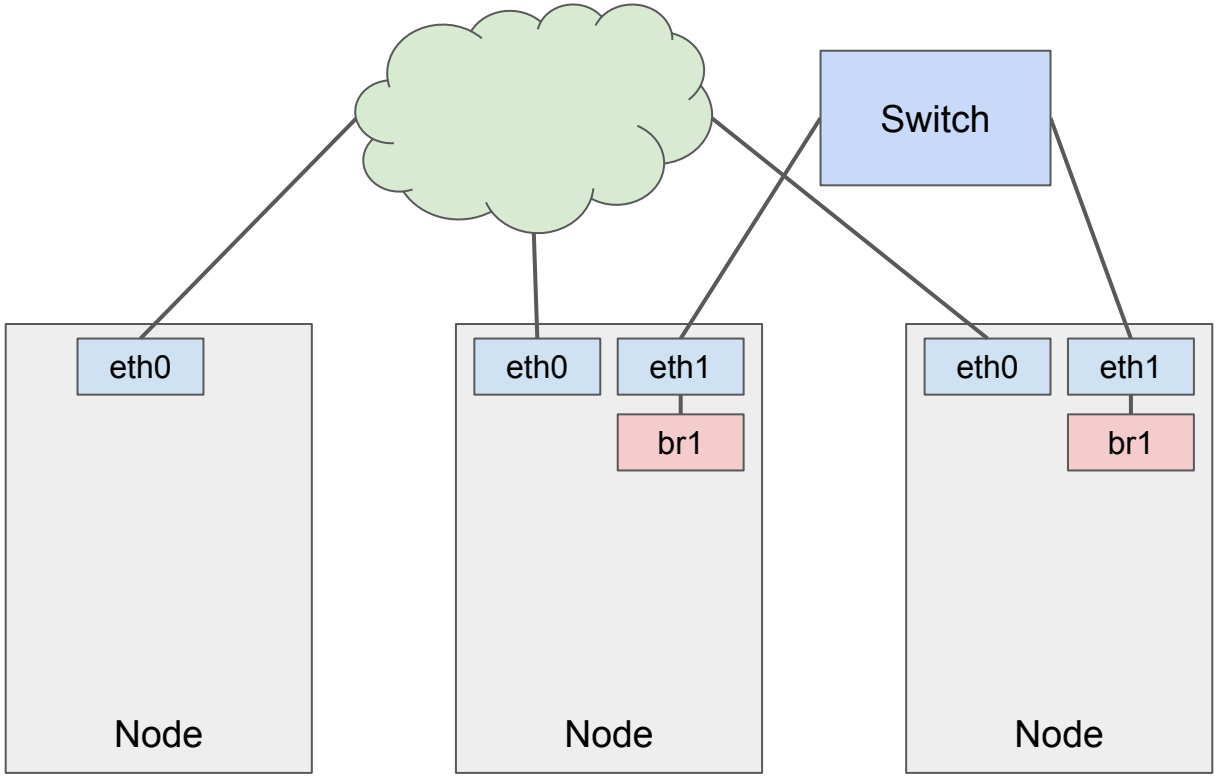
Faster Packet Processing (Cluster Network)



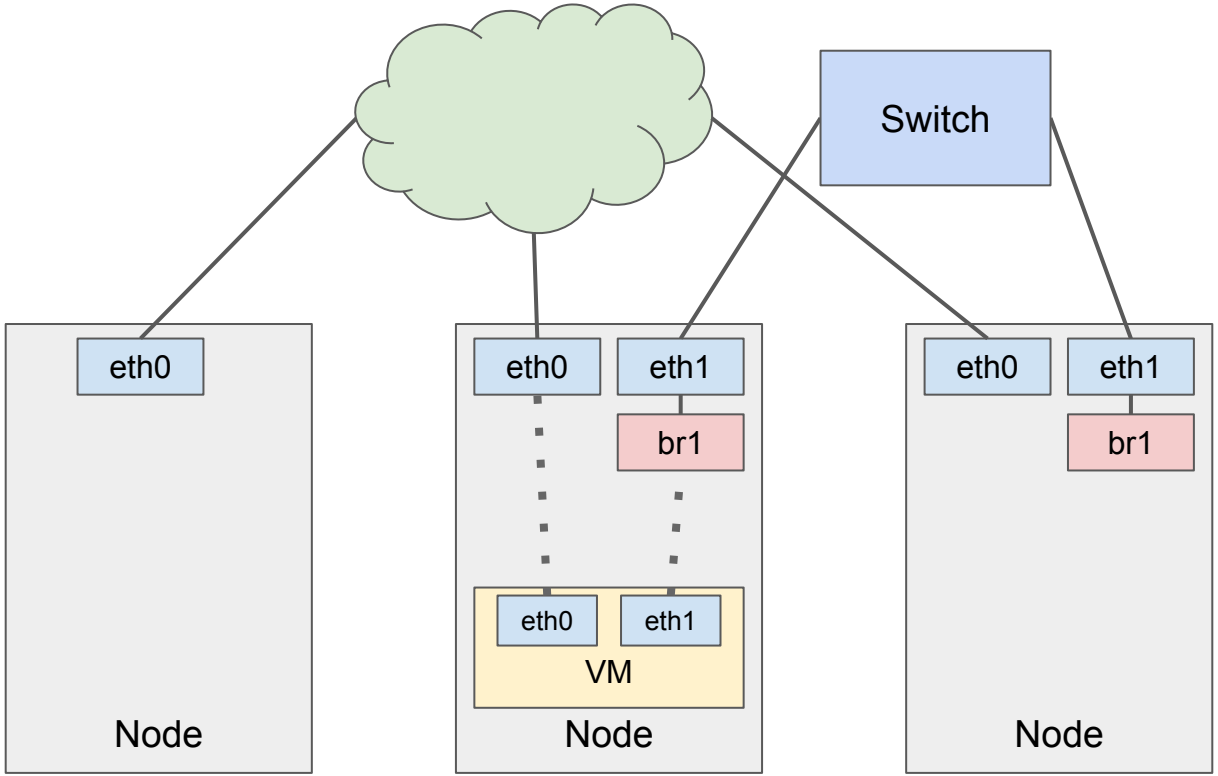
Faster Packet Processing (Cluster Network)



Faster Packet Processing (Cluster Network)

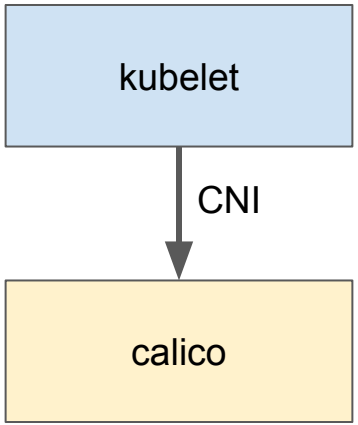


Faster Packet Processing (Cluster Network)

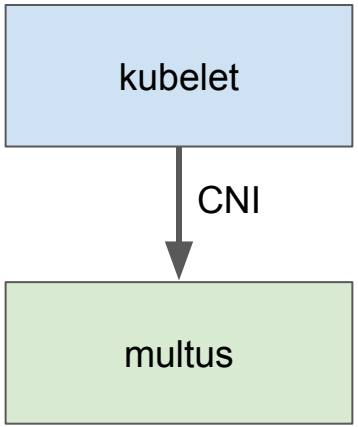


Faster Packet Processing (Multus)

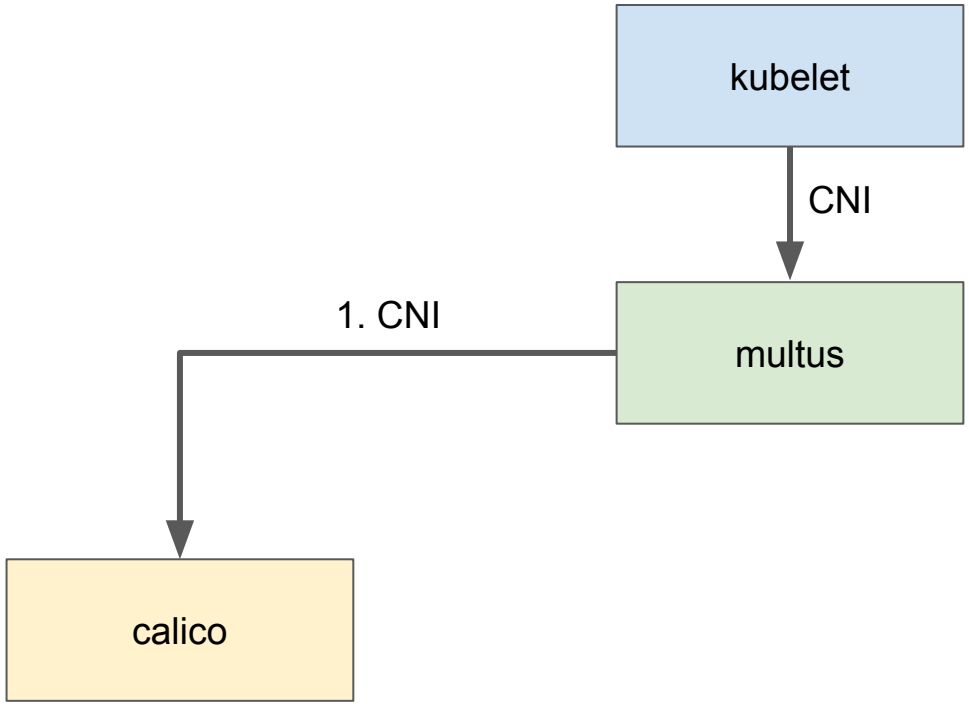
Faster Packet Processing (Multus)



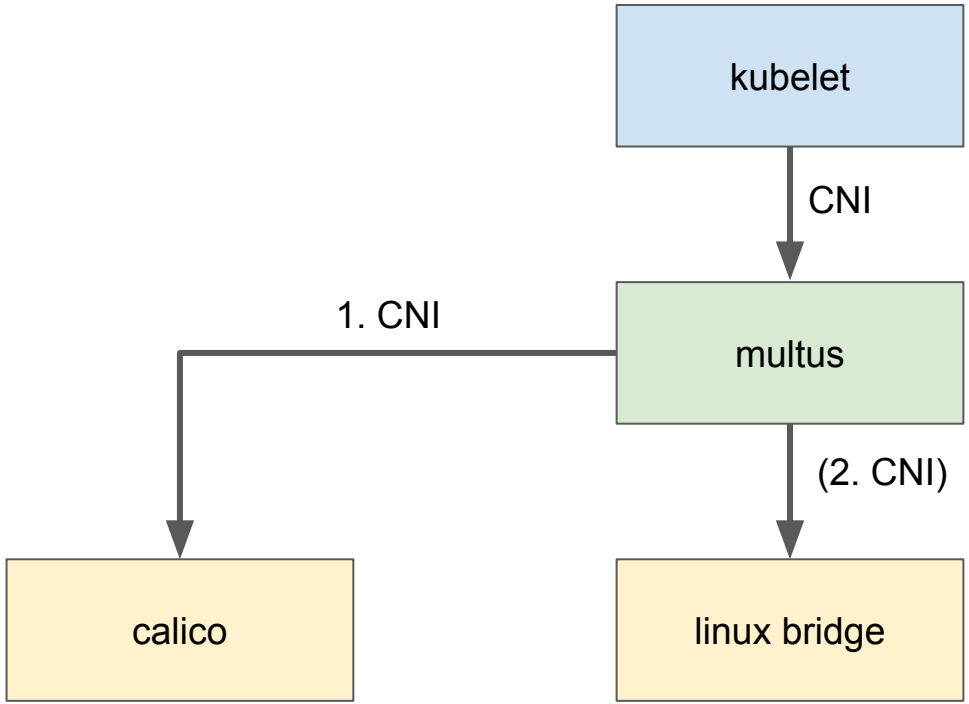
Faster Packet Processing (Multus)



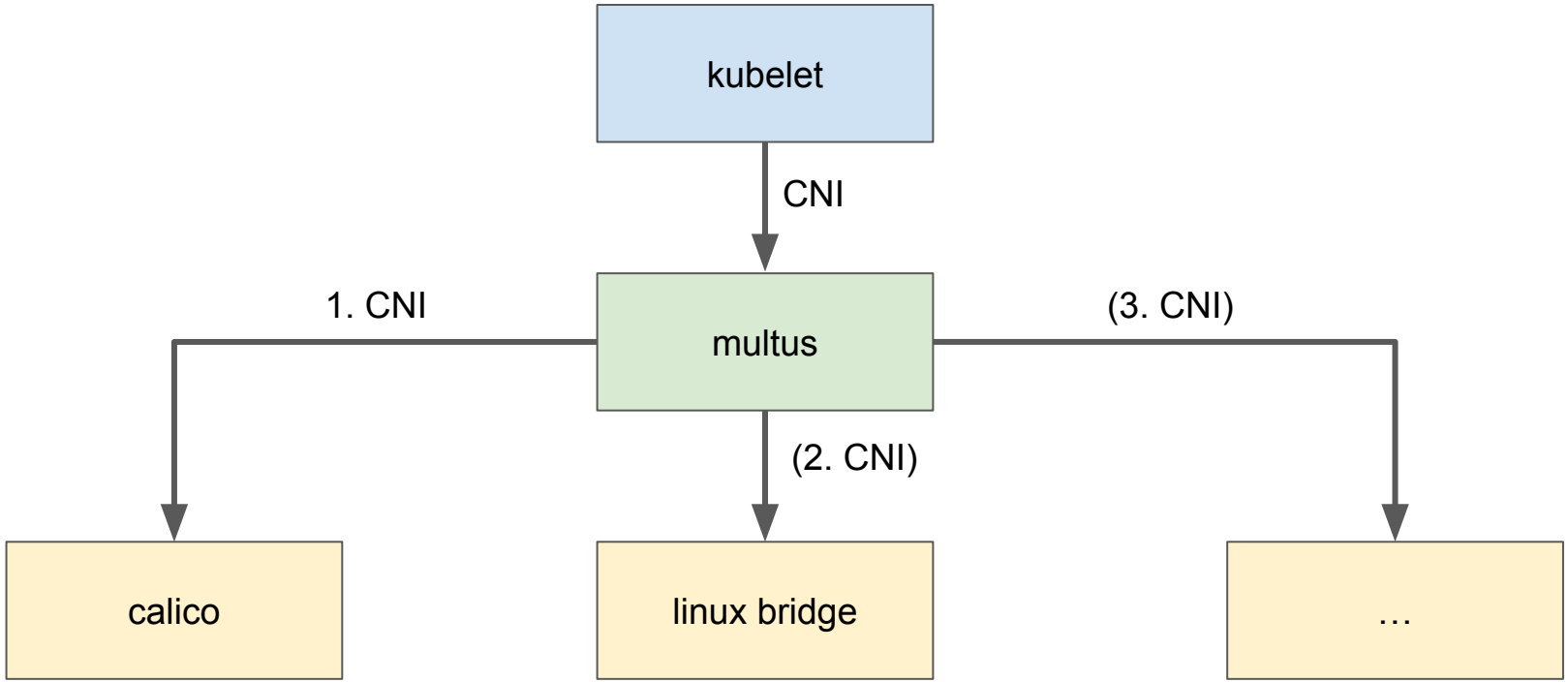
Faster Packet Processing (Multus)



Faster Packet Processing (Multus)

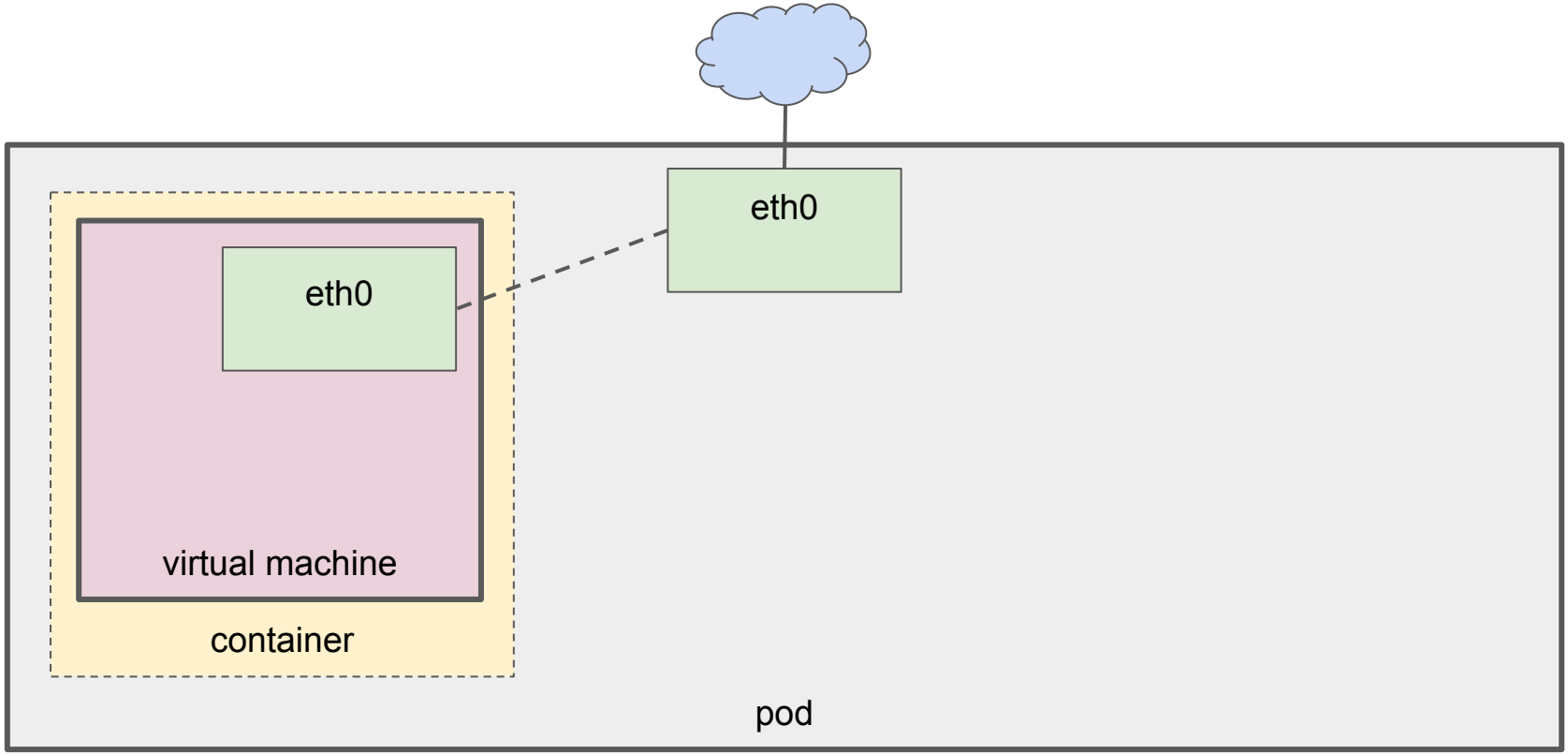


Faster Packet Processing (Multus)

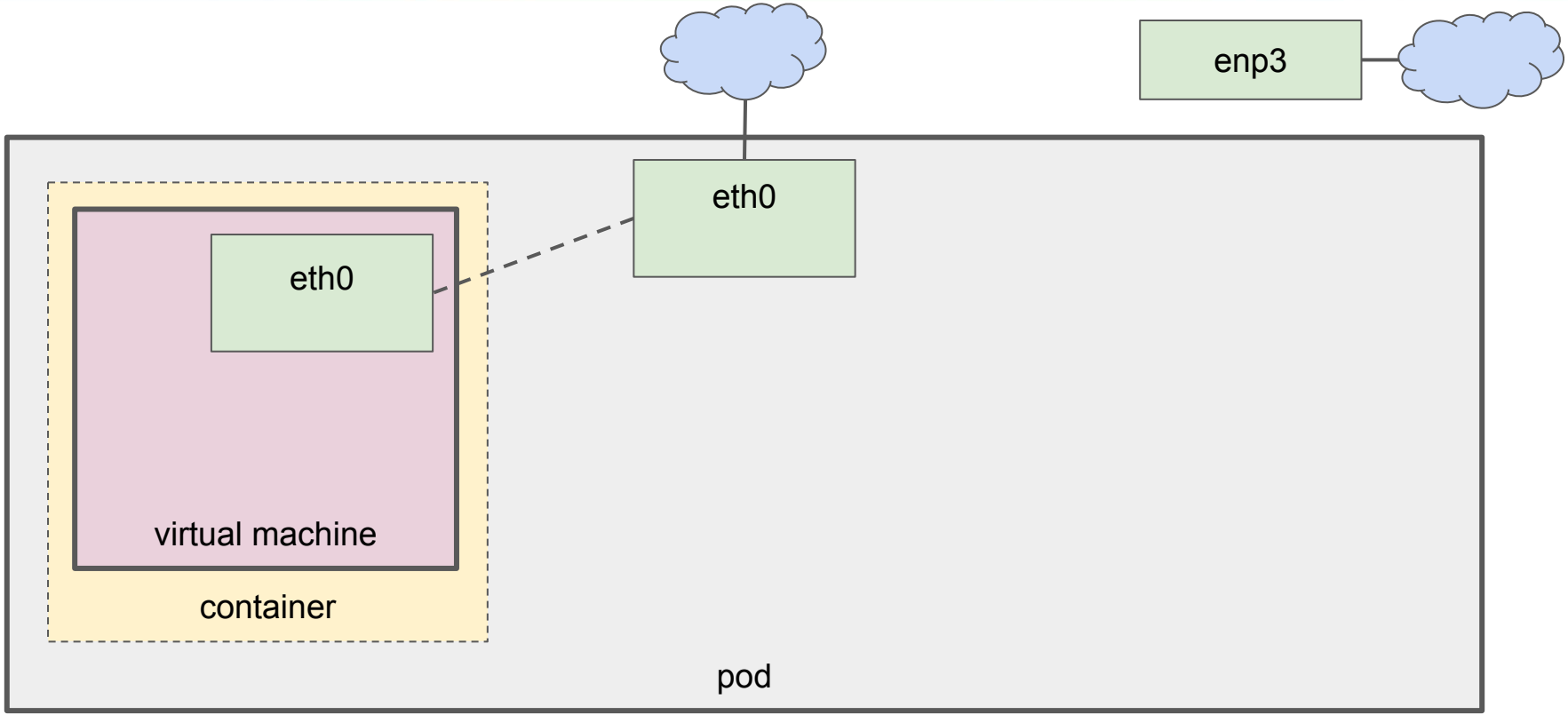


Faster Packet Processing (Binding)

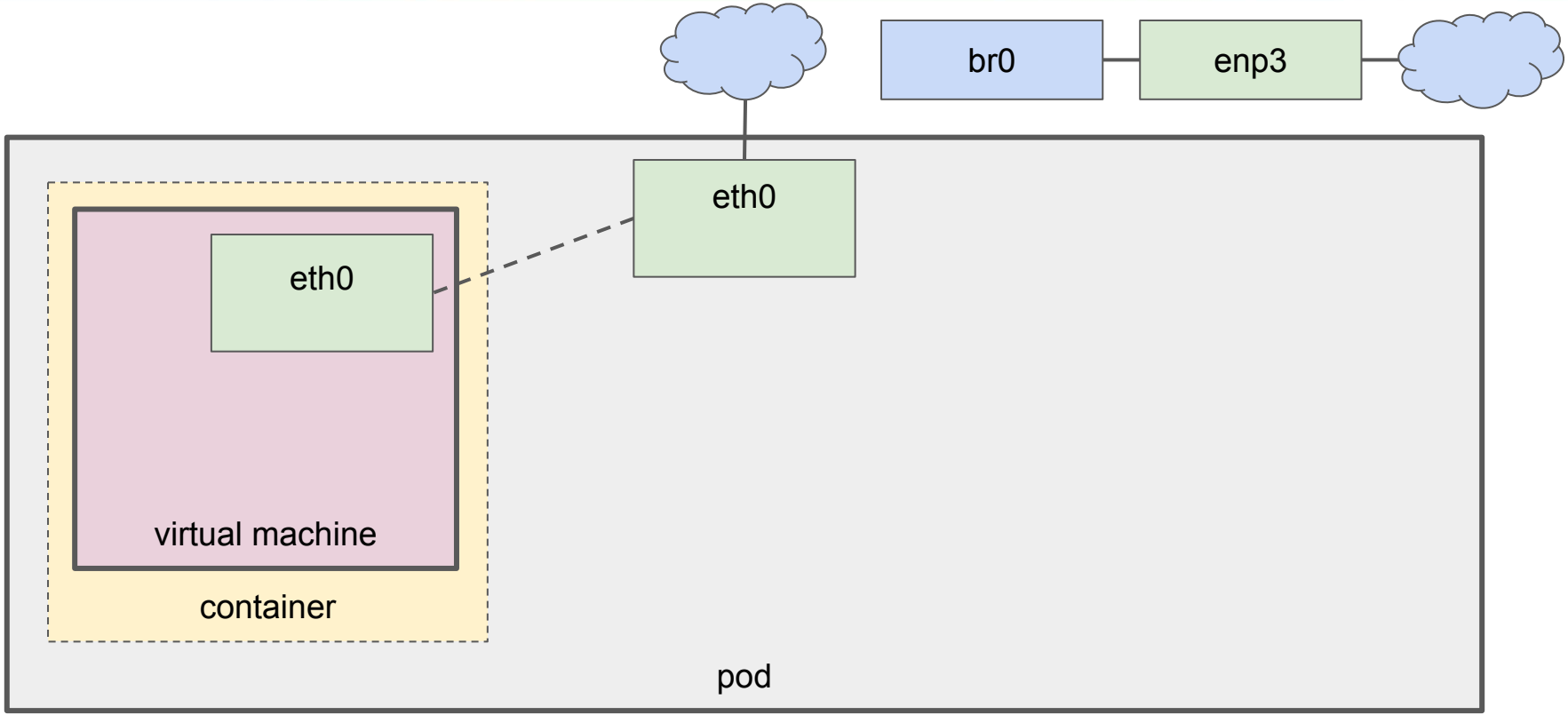
Faster Packet Processing (Binding)



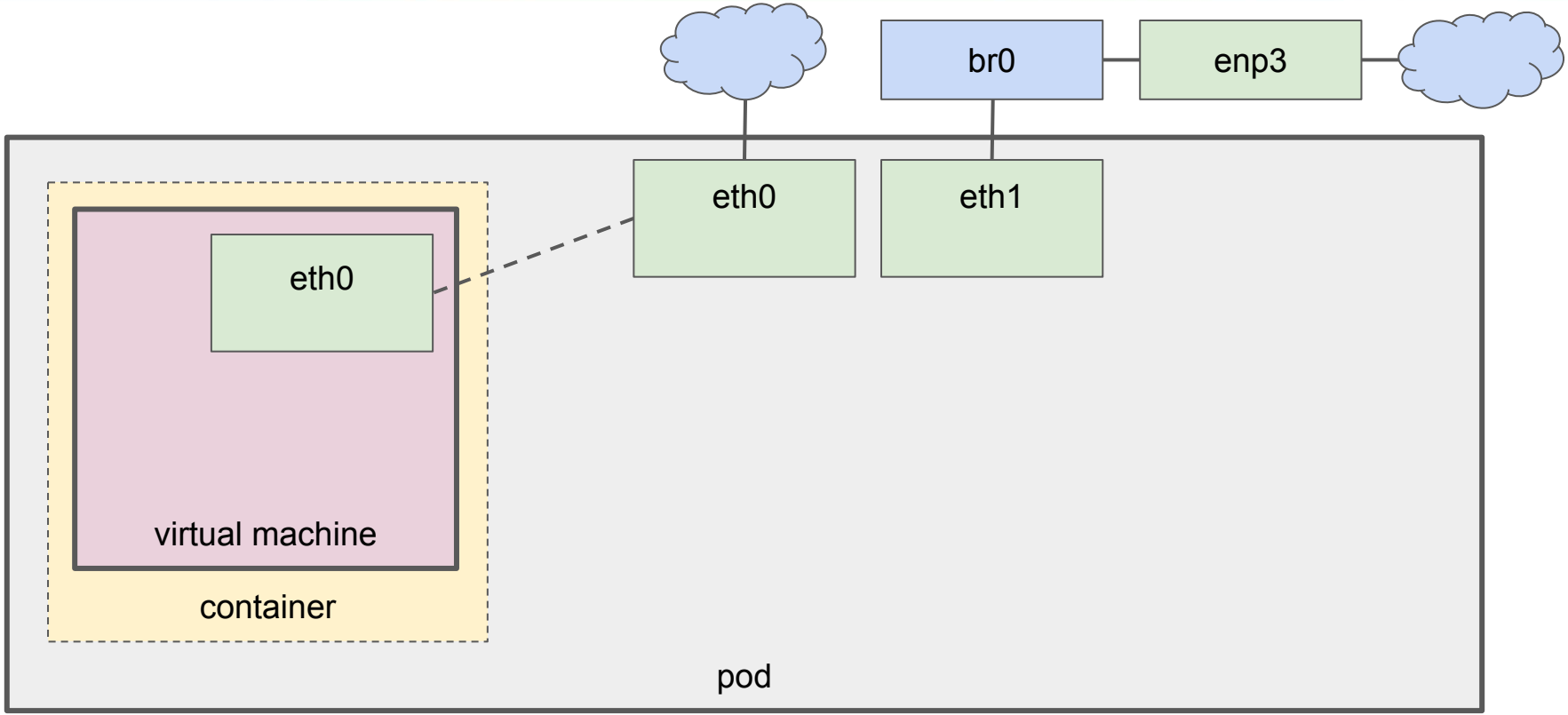
Faster Packet Processing (Binding)



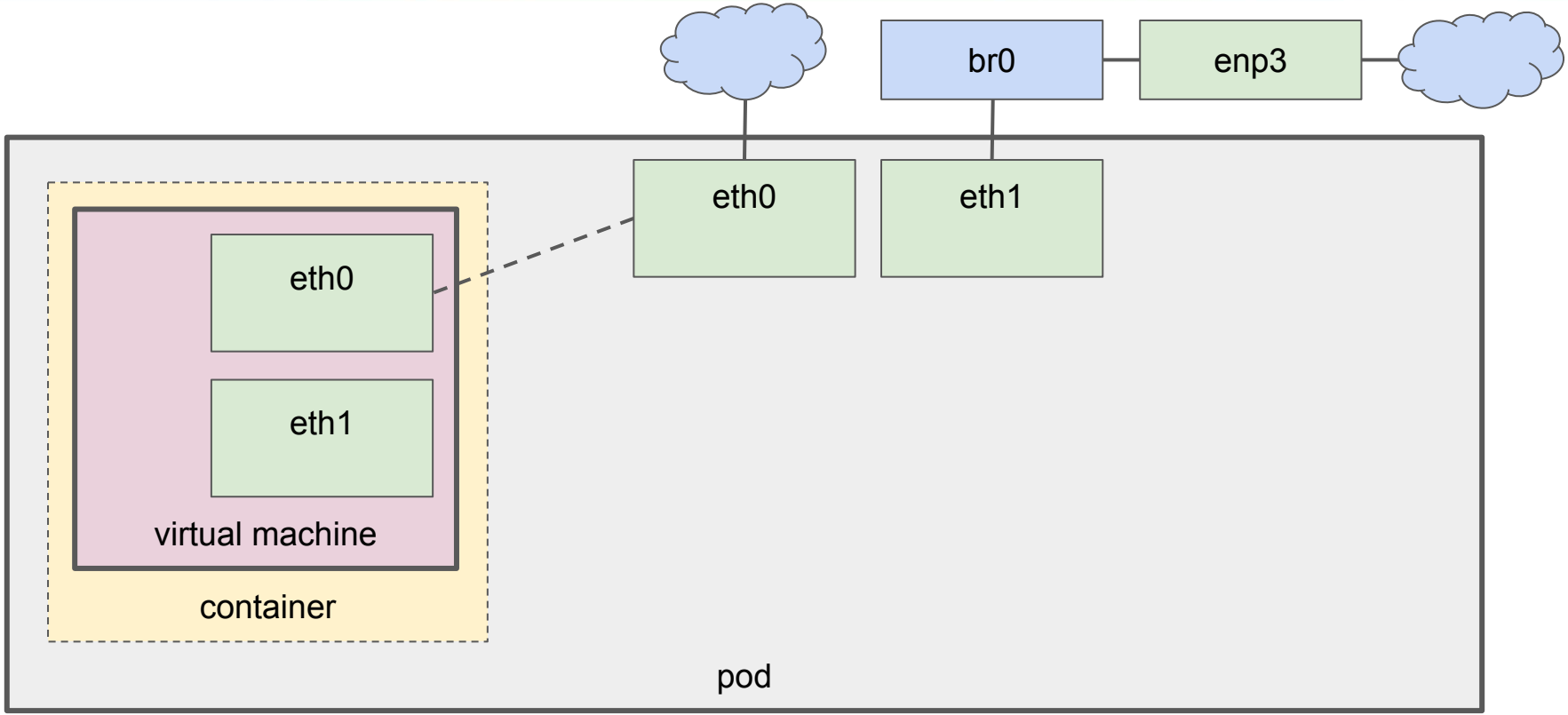
Faster Packet Processing (Binding)



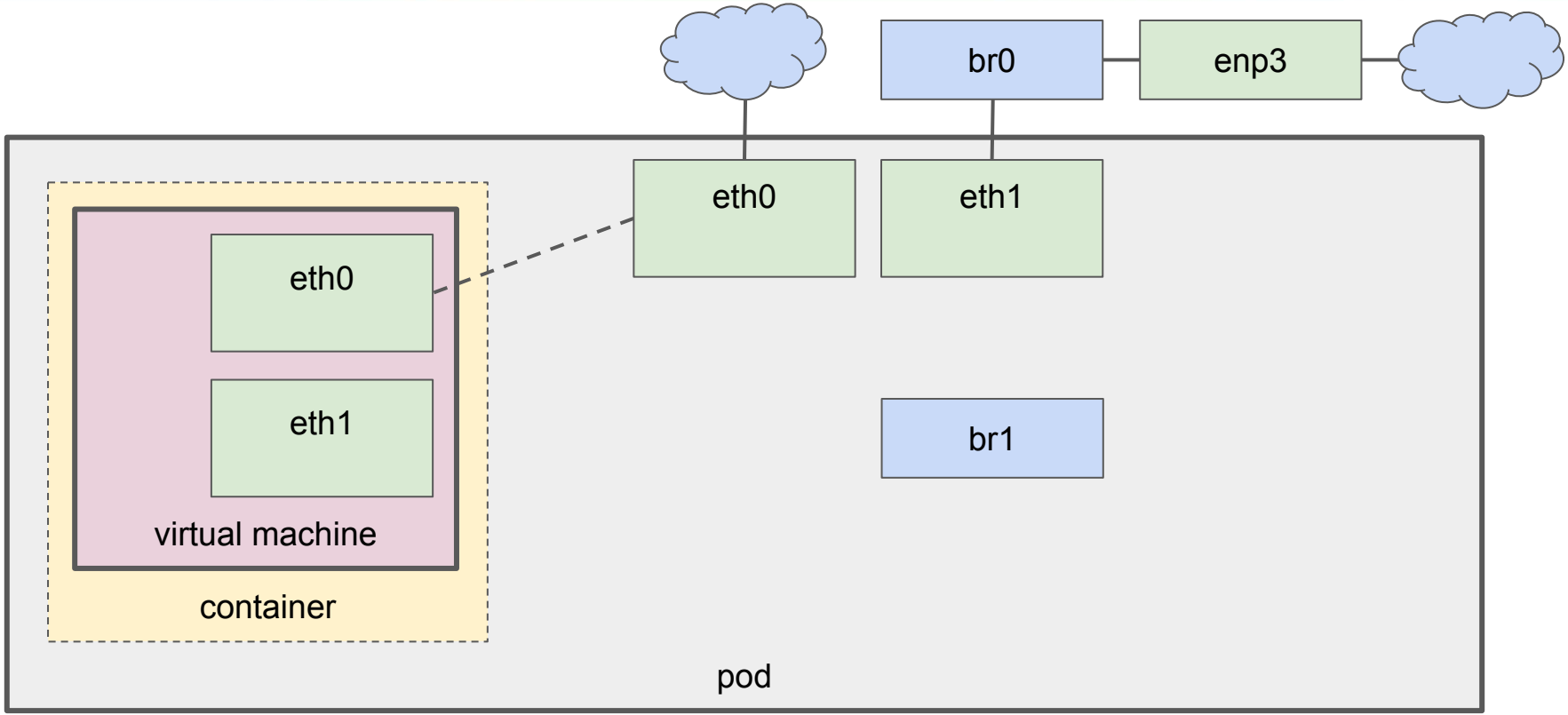
Faster Packet Processing (Binding)



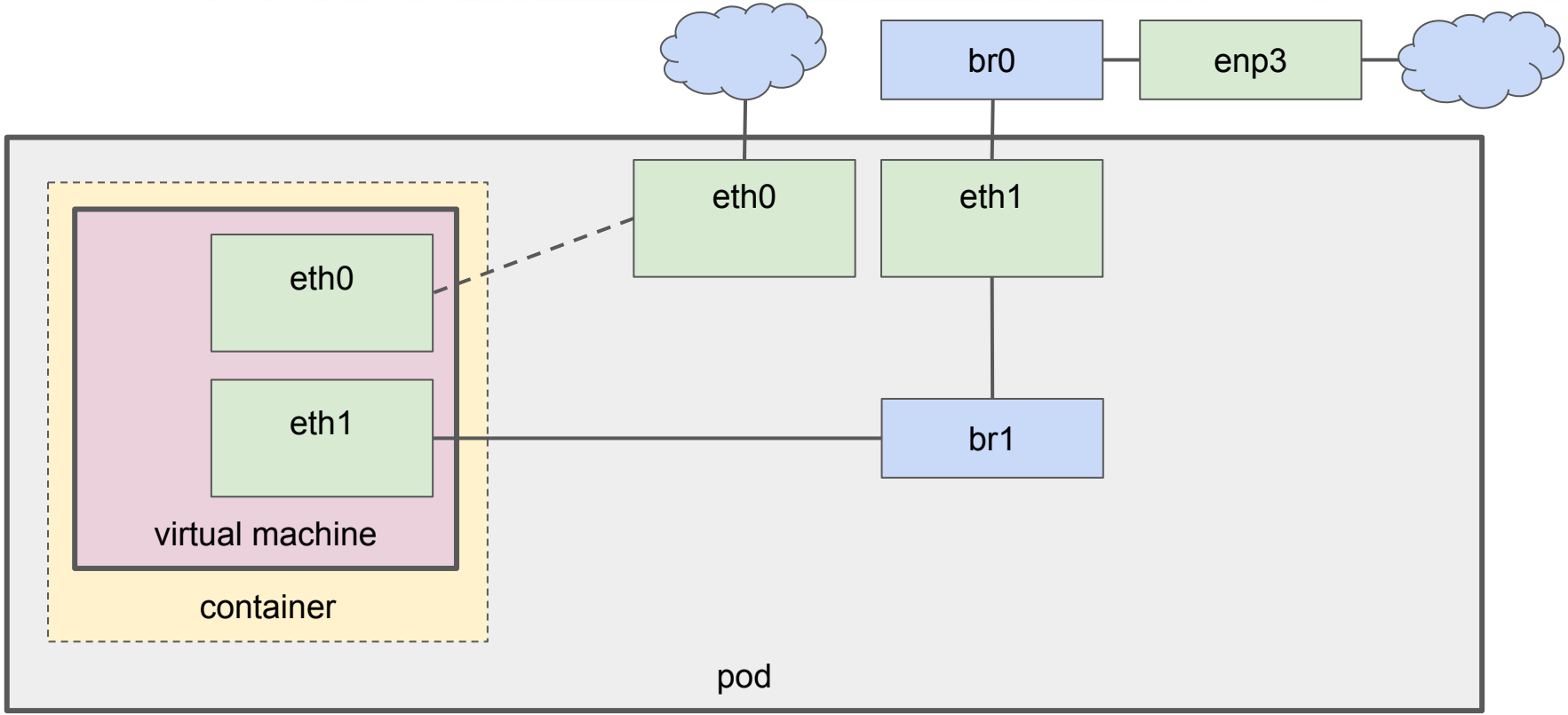
Faster Packet Processing (Binding)



Faster Packet Processing (Binding)



Faster Packet Processing (Binding)



Faster Packet Processing (Example)

```
apiVersion: nmstate.io/v1alpha1
kind: NodeNetworkConfigurationPolicy
metadata:
  name: br1-eth0
```

```
spec:
  desiredState:
    interfaces:
    - name: br1
      type: linux-bridge
      state: up
      ipv4:
        dhcp: true
        enabled: true
      bridge:
        port:
        - name: eth0
```

Faster Packet Processing (Example)

```
apiVersion: k8s.cni.cncf.io/v1
kind: NetworkAttachmentDefinition
metadata:
  name: blue-network
  annotations:
    k8s.v1.cni.cncf.io/resourceName: ovs-cni.network.kubevirt.io/br1
spec:
  config: '{
      "cniVersion": "0.3.1",
      "type": "bridge",
      "bridge": "br1",
      "vlan": 100
    }'
```

Faster Packet Processing (Example)

```
apiVersion: kubevirt.io/v1alpha3
kind: VirtualMachineInstance
metadata:
  name: vmi-test
  ...
```

```
spec:
  networks:
  - name: default
    pod: {}
  - name: blue
    multus:
      networkName: blue-network
  domain:
    devices:
      ...
      interfaces:
      - name: default
        masquerade: {}
      - name: blue
        bridge: {}
    ...
...
```

Really Fast Packet Processing

Fast Packet Processing

- SR-IOV NIC configured on the host,
- exposed as a node resource,
- plugged as a device into a container,
- mounted into the virtual machine through as a passthrough.

Fast Packet Processing

- SR-IOV NIC configured on the host,
- exposed as a node resource,
- plugged as a device into a container,
- mounted into the virtual machine through as a passthrough.

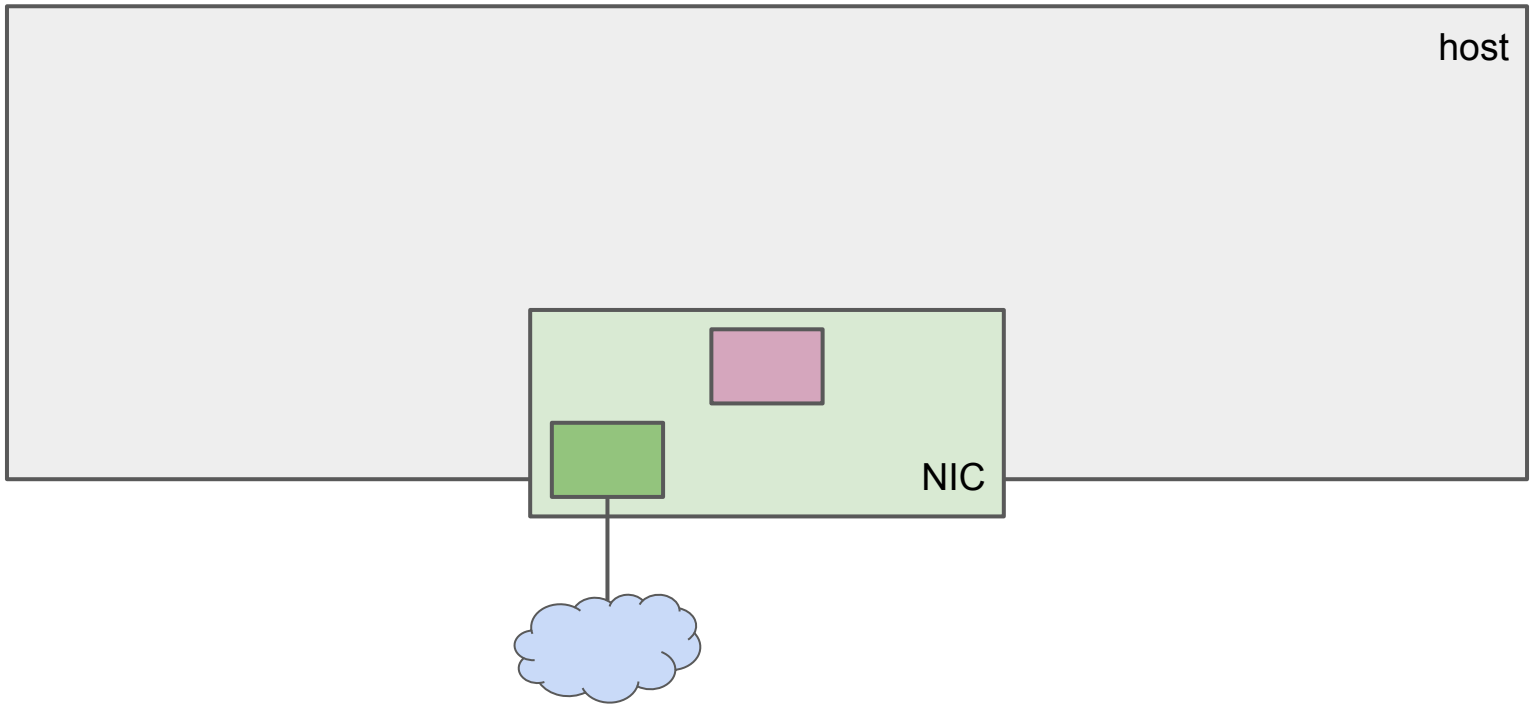
- As fast as it can get,
- requires special hardware and can service only so many containers.

Fast Packet Processing (SR-IOV)

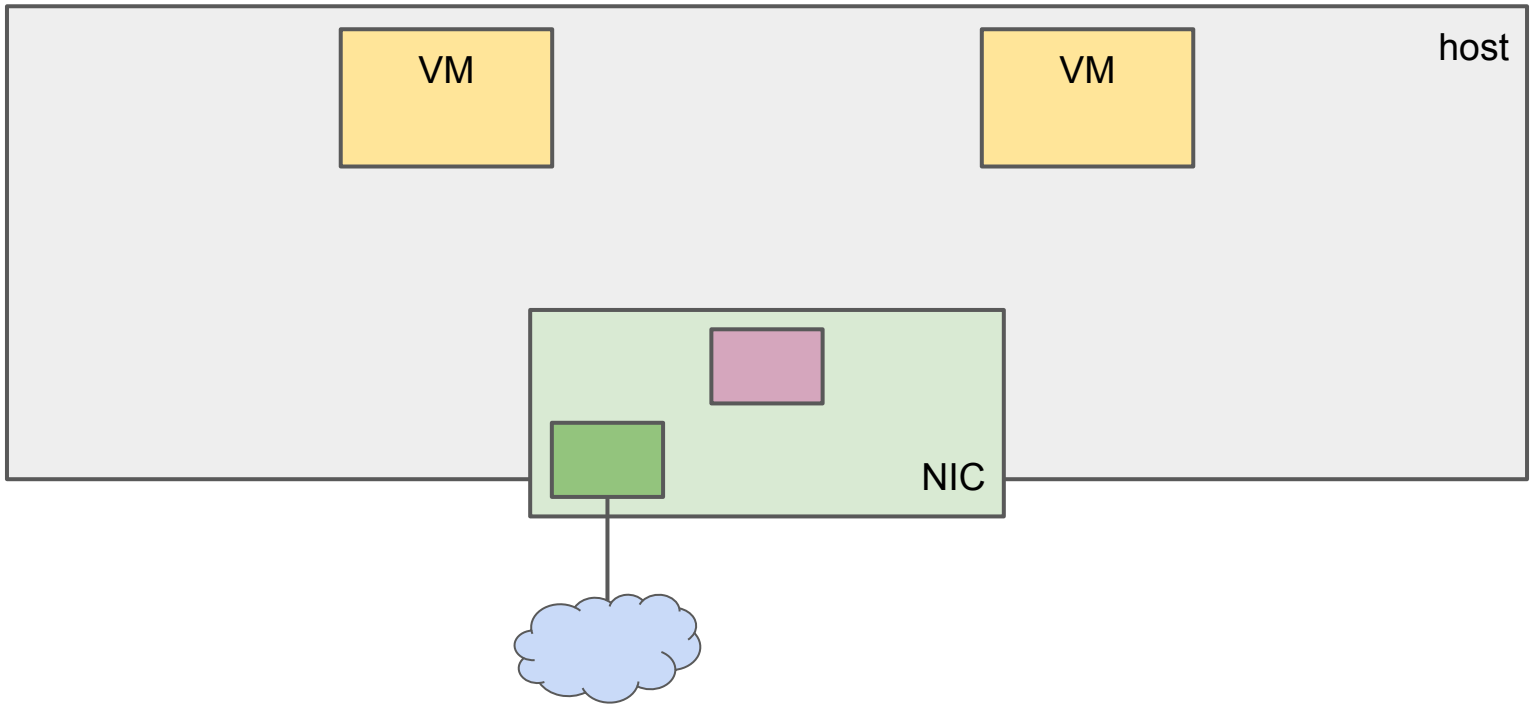
- Dedicated NICs for each VM would be a clunky solution.

- SR-IOV exposes a single NIC as many,
- isolation of PCIe resources,
- Physical Functions (PFs) and Virtual Functions (VFs),
- multiple containers and VMs can utilize the hardware NIC at the same time.

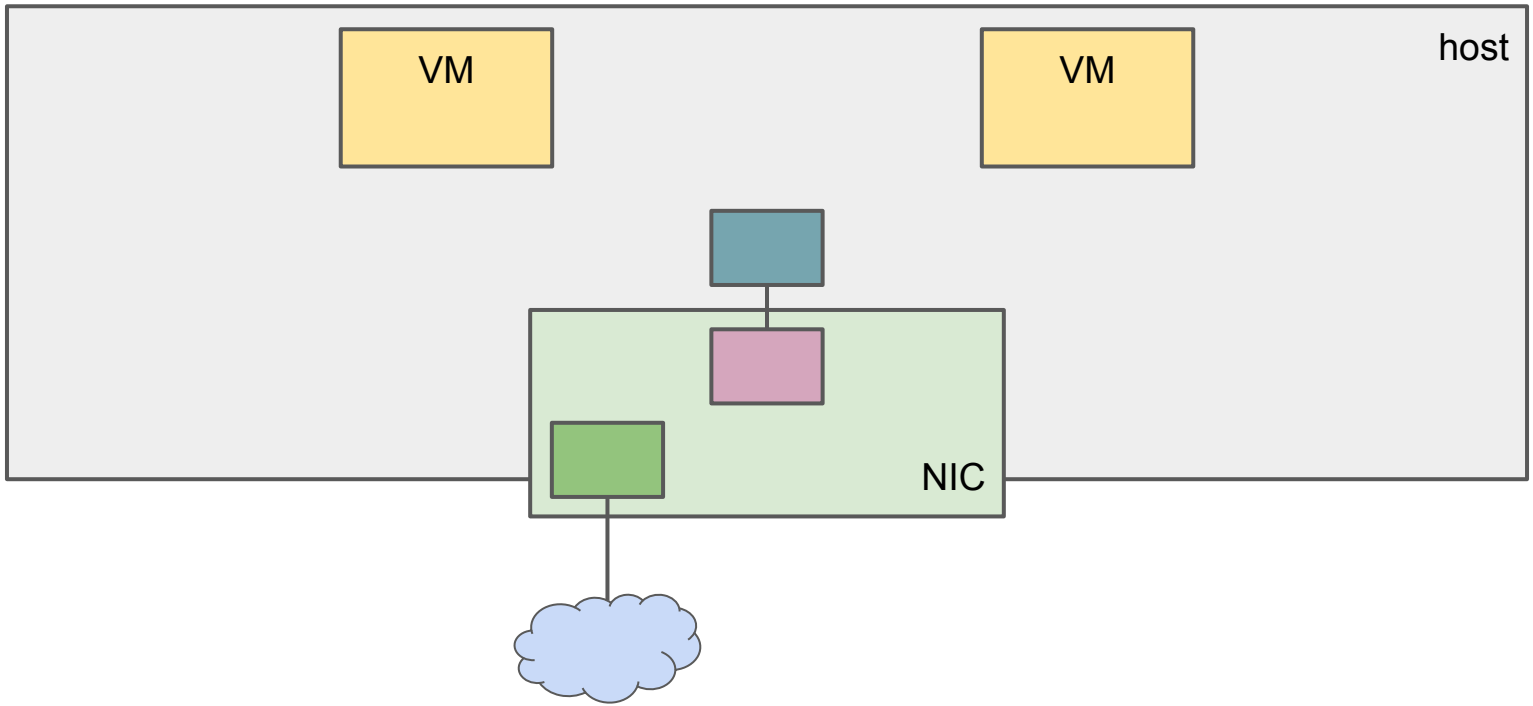
Fast Packet Processing (not SR-IOV)



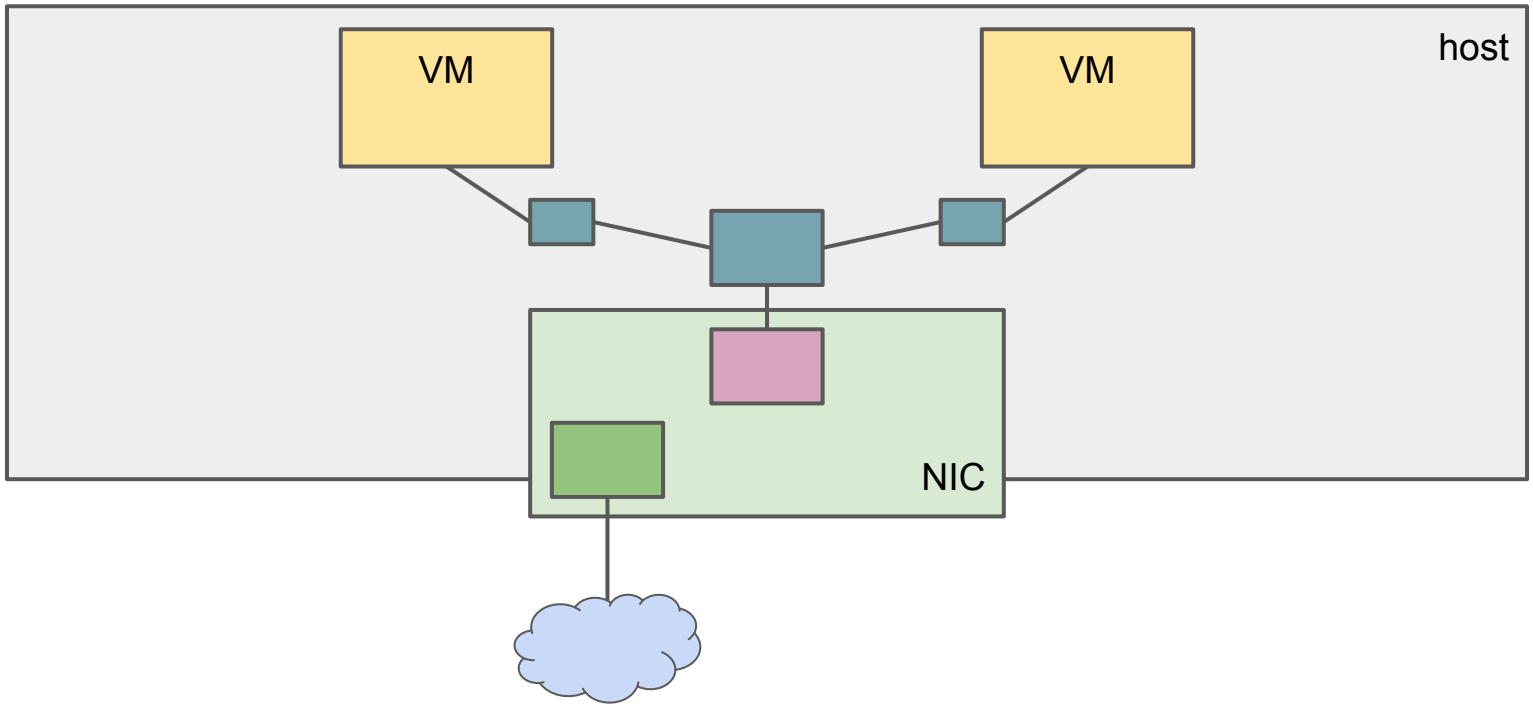
Fast Packet Processing (not SR-IOV)



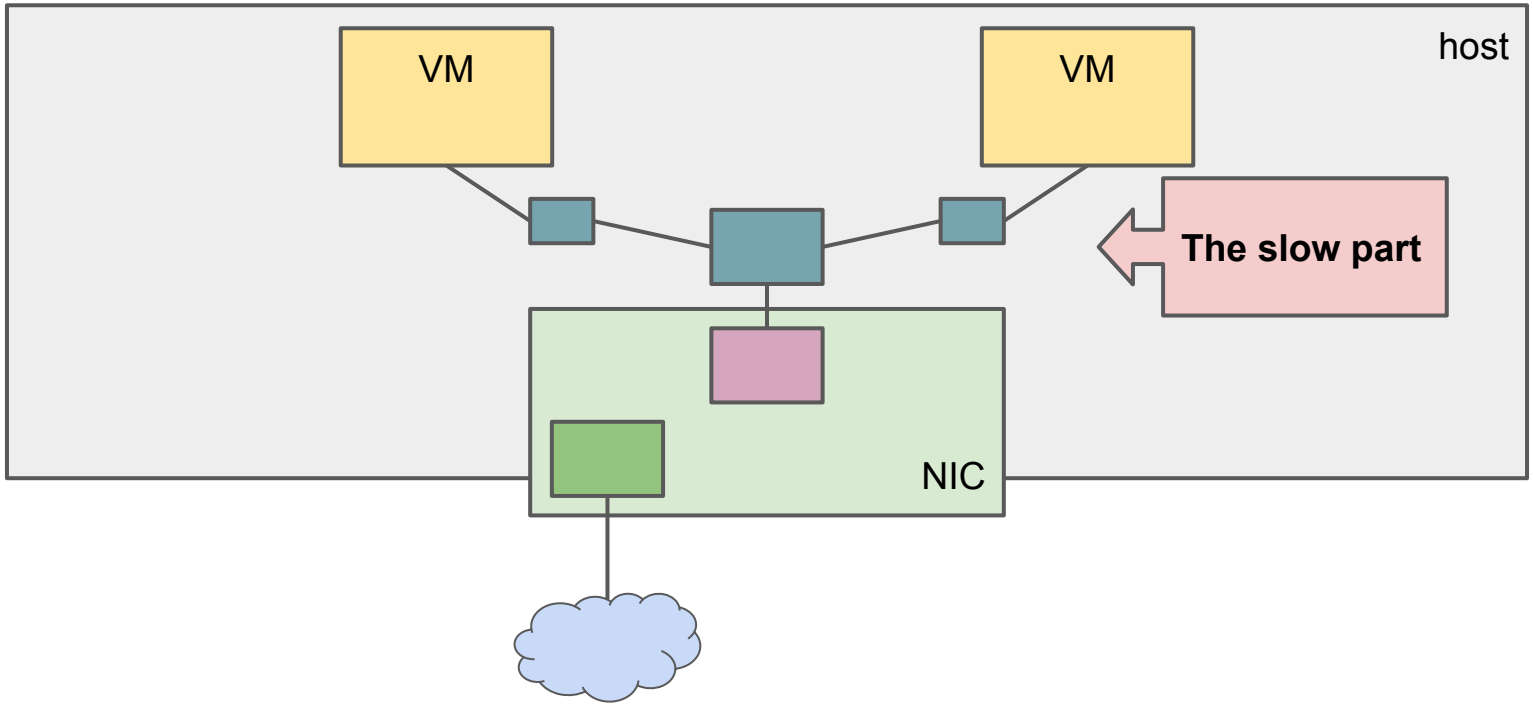
Fast Packet Processing (not SR-IOV)



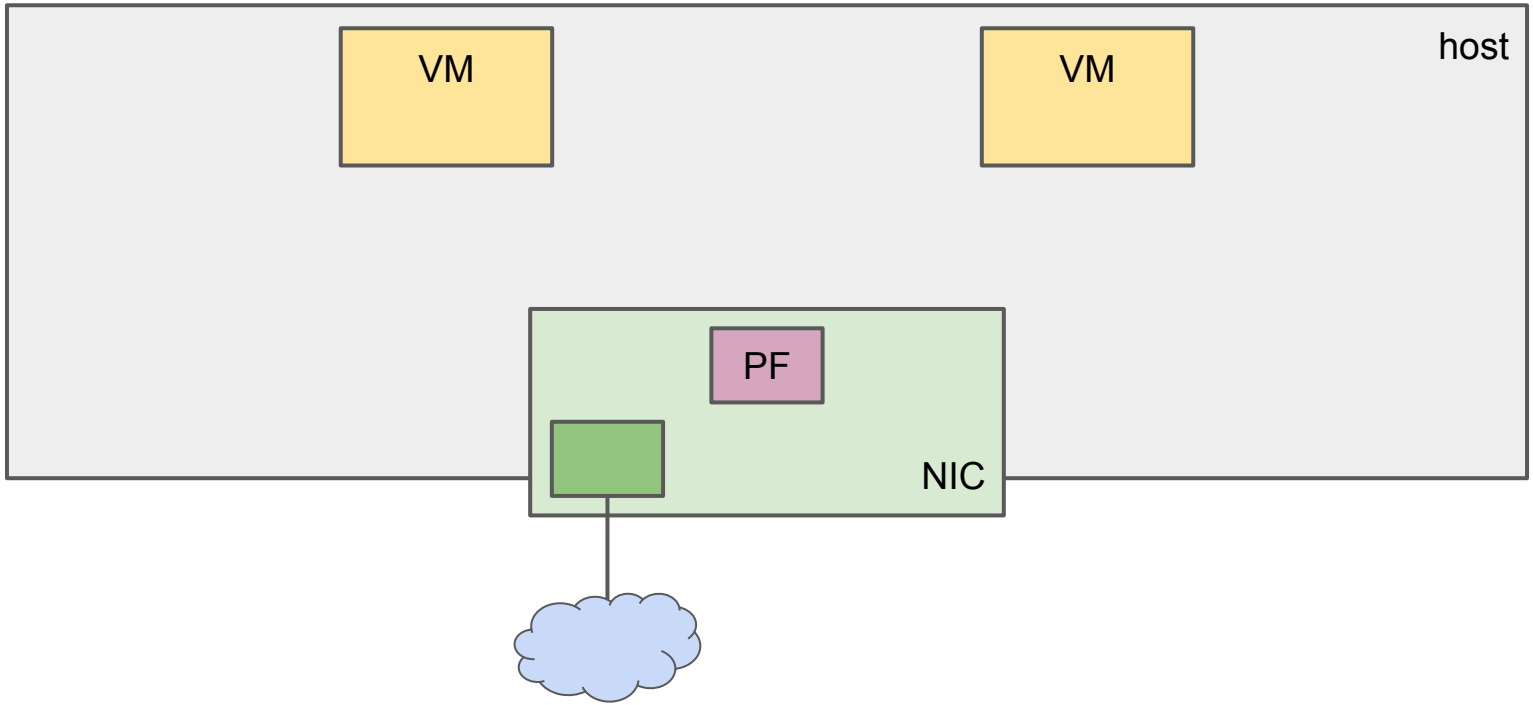
Fast Packet Processing (not SR-IOV)



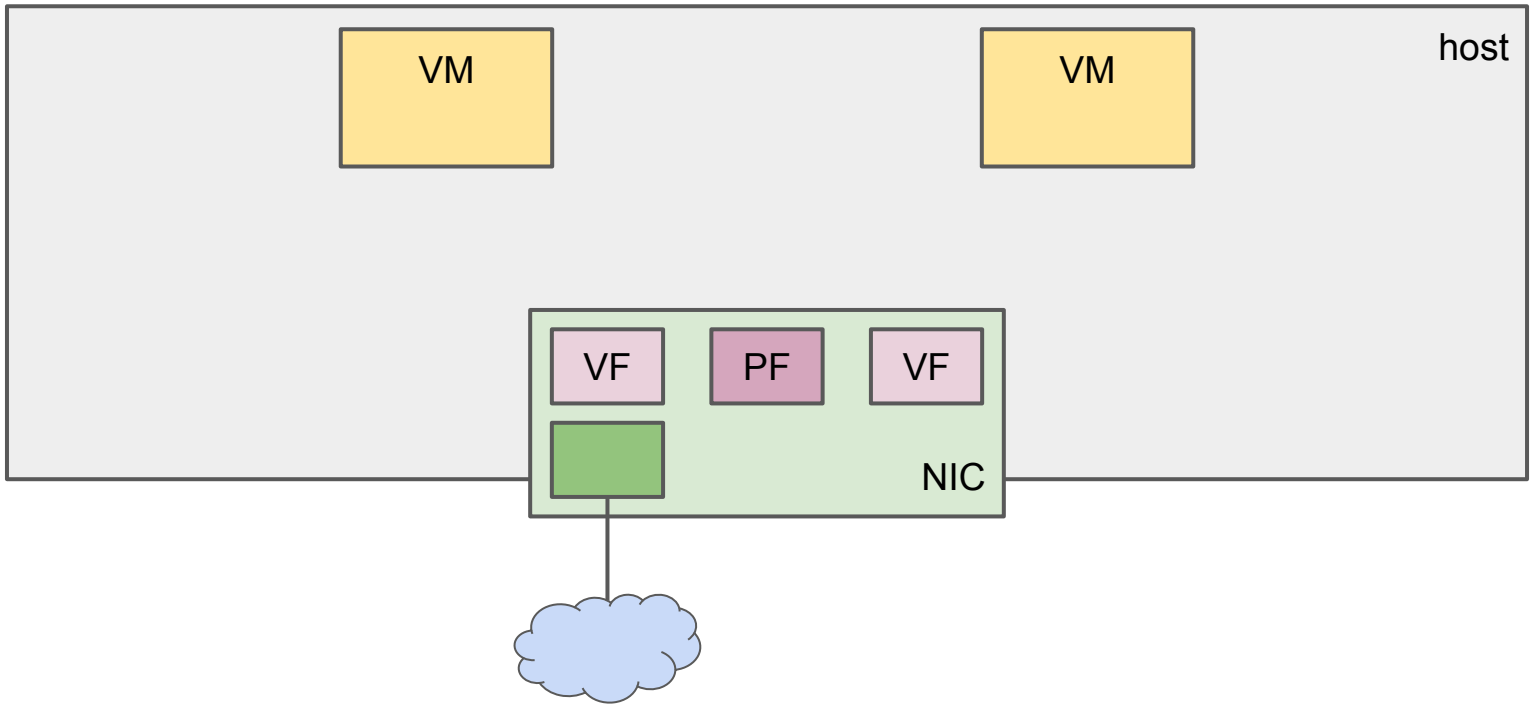
Fast Packet Processing (not SR-IOV)



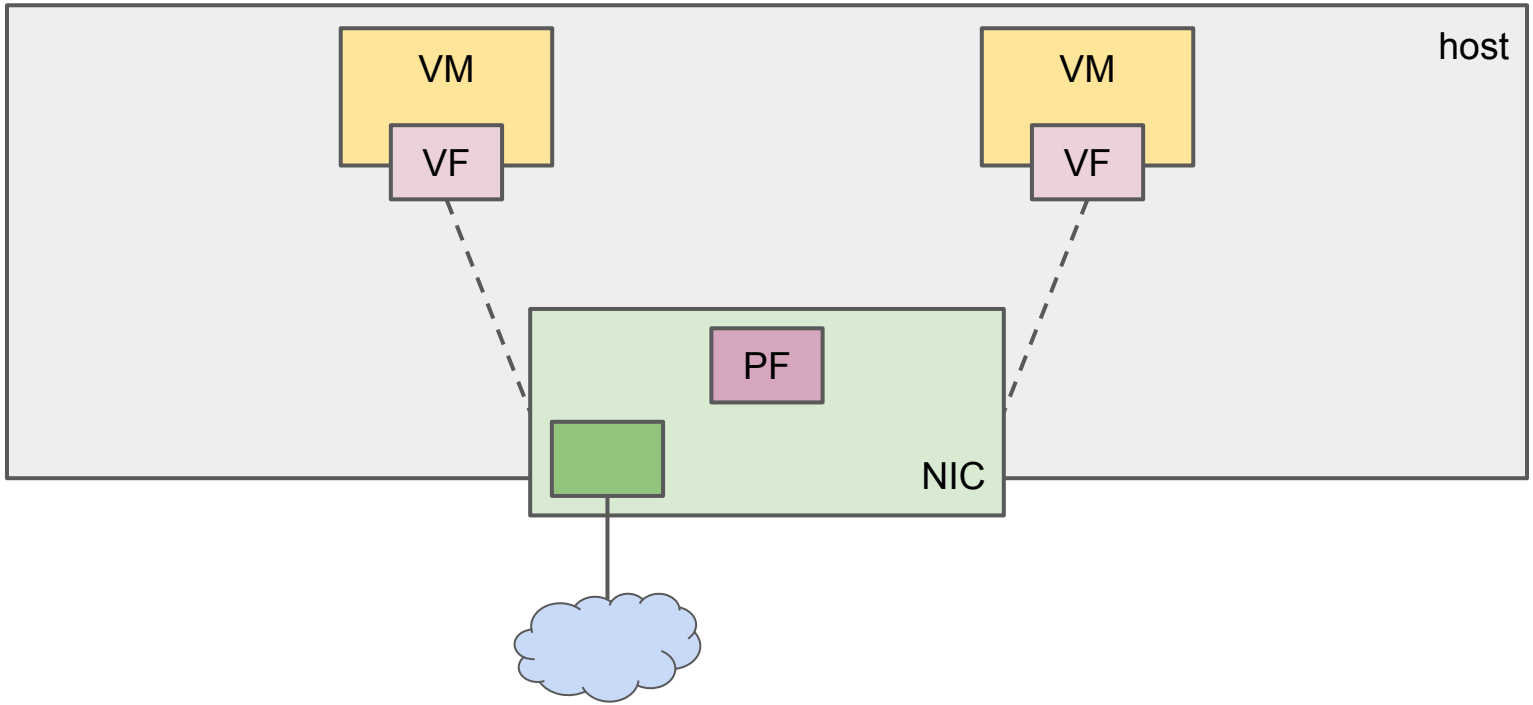
Fast Packet Processing (SR-IOV)



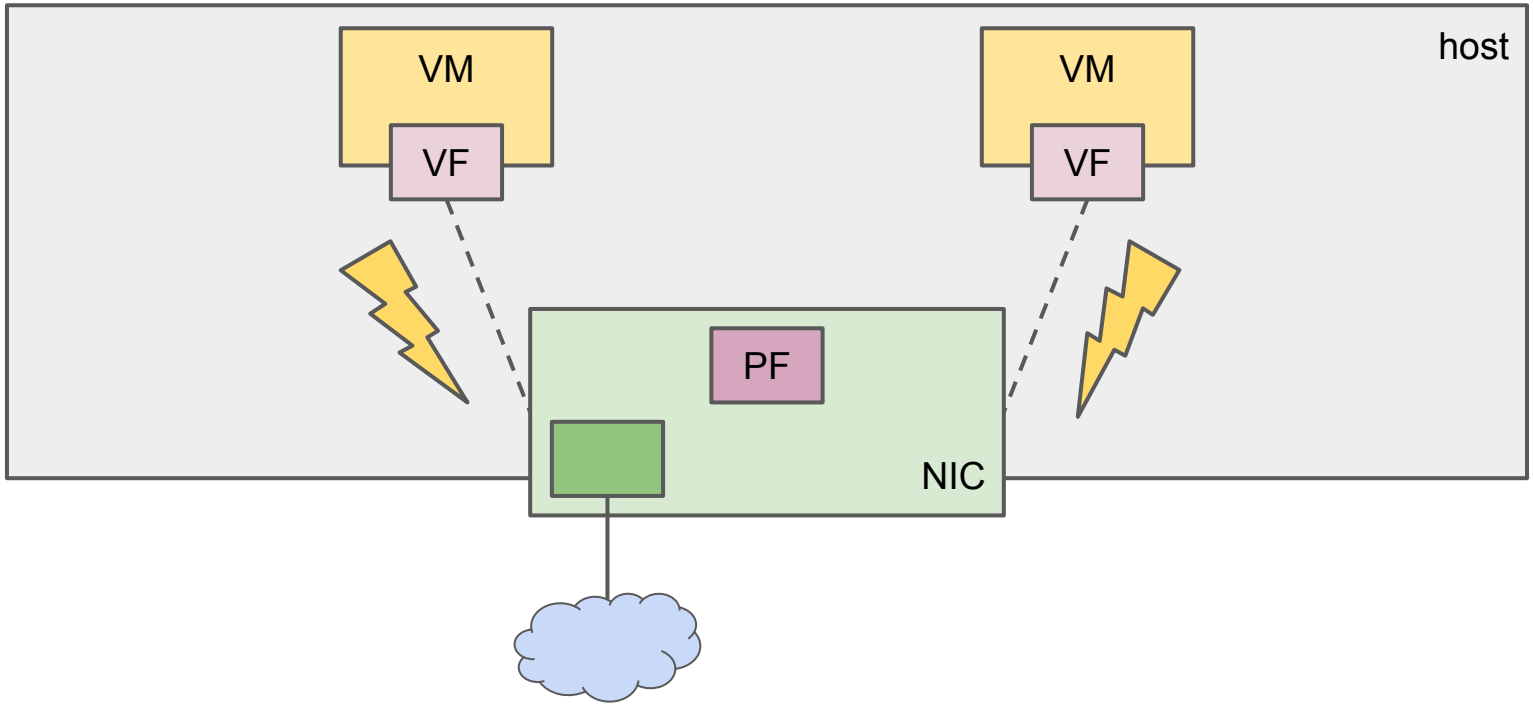
Fast Packet Processing (SR-IOV)



Fast Packet Processing (SR-IOV)



Fast Packet Processing (SR-IOV)



Fast Packet Processing (SR-IOV Operator)

- Optional, but so useful,
- reports available NICs,
- configures them,
- deploys other needed components,
- and all of that through Kubernetes API.

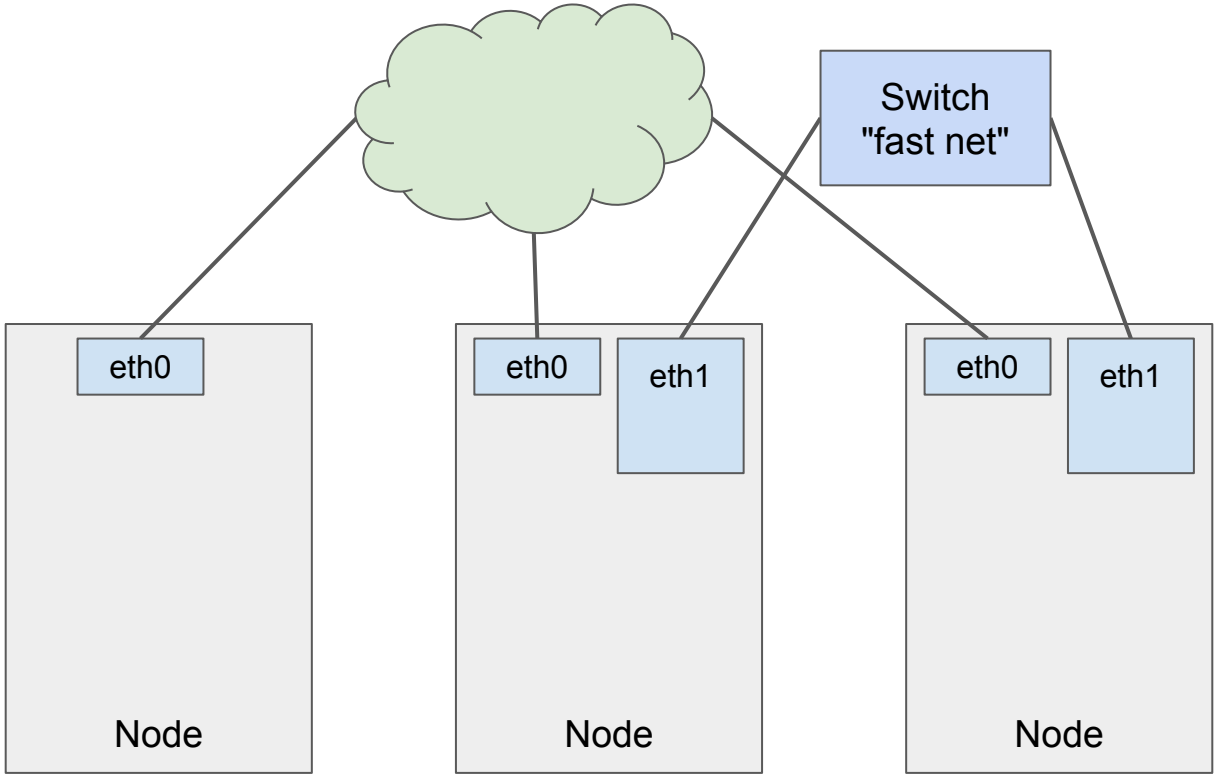
Fast Packet Processing (Device Plugin)

- Allocates countable resources and plugs them into containers,
- gRPC service cooperating with kubelet,
- discovery,
- advertising,
- allocation,
- health checking.

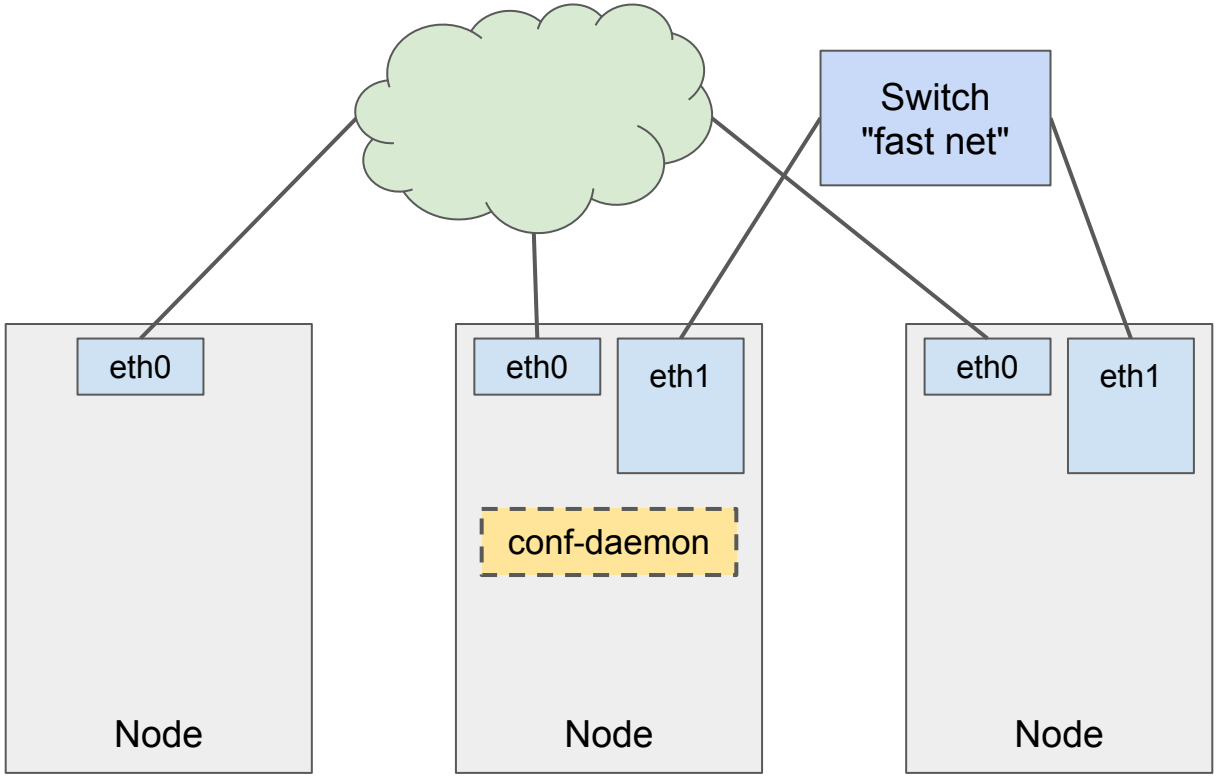
Fast Packet Processing (CNI)

- SR-IOV CNI complements the device plugin,
- it configures allocated VF,
- both VF parameters and configuration of container's netlink interface.

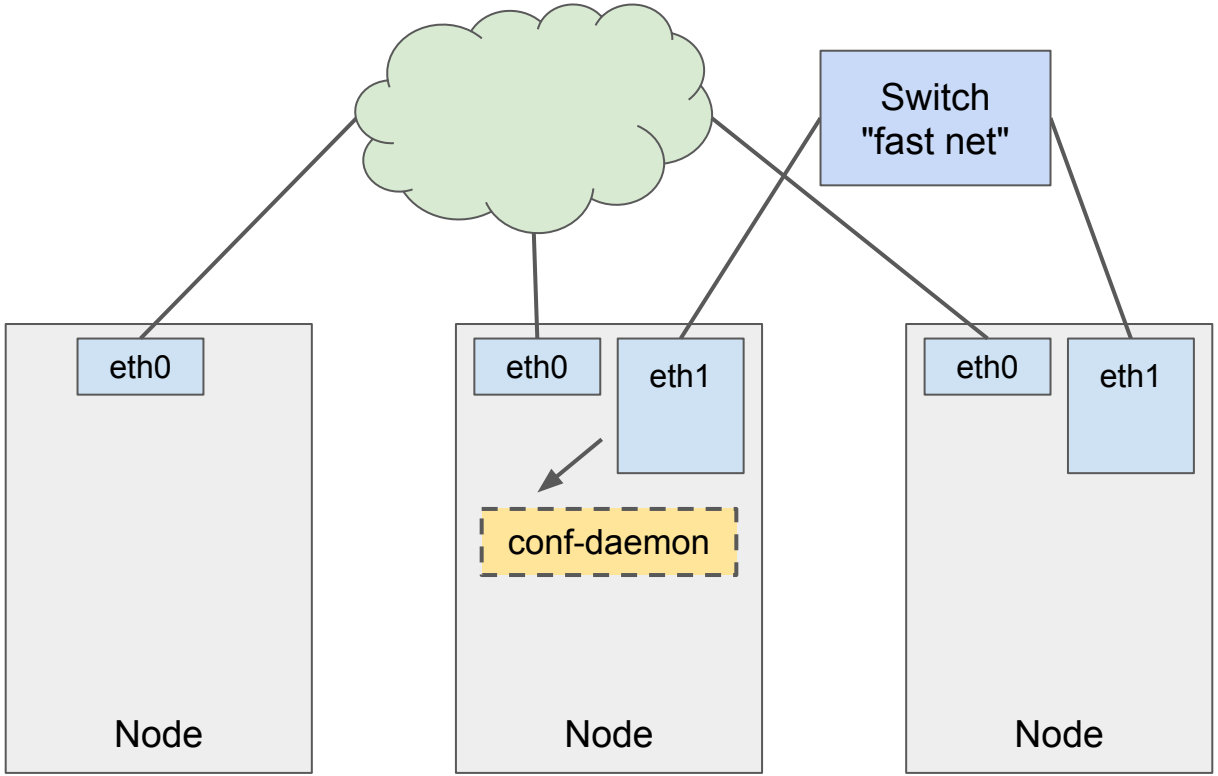
Fast Packet Processing (Configuration)



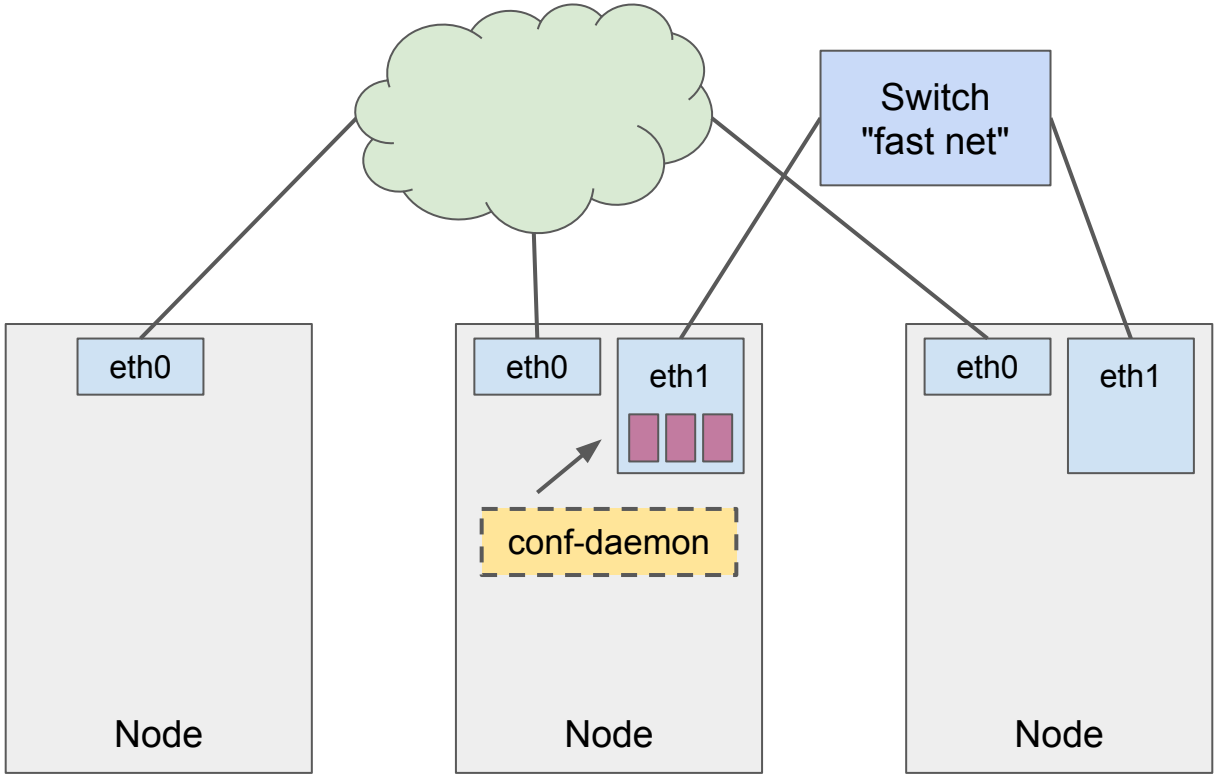
Fast Packet Processing (Configuration)



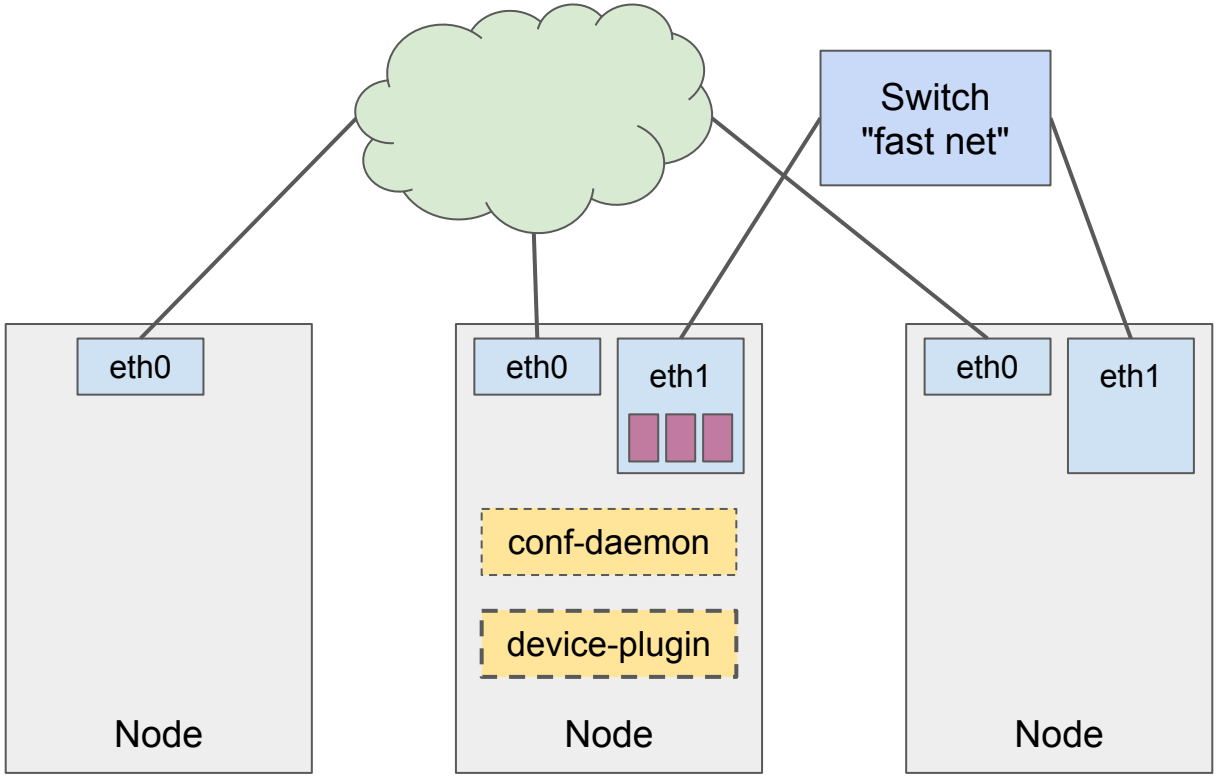
Fast Packet Processing (Configuration)



Fast Packet Processing (Configuration)

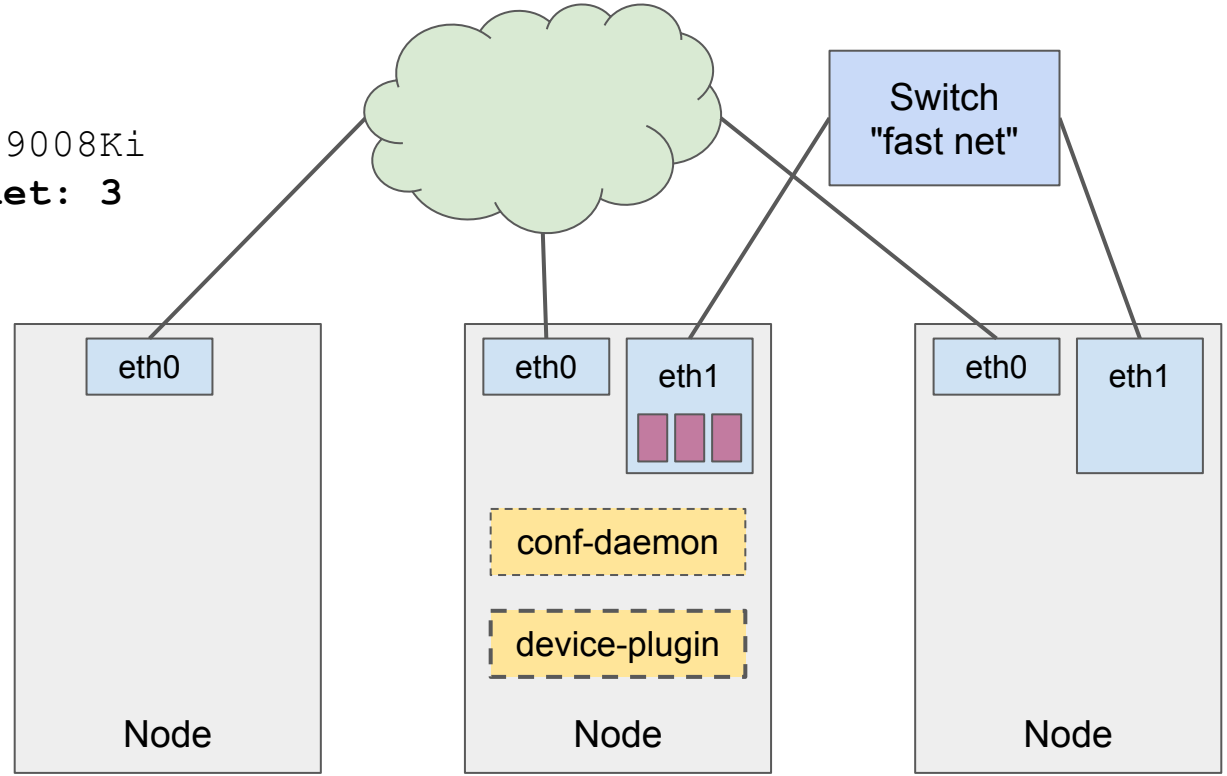


Fast Packet Processing (Advertisement)



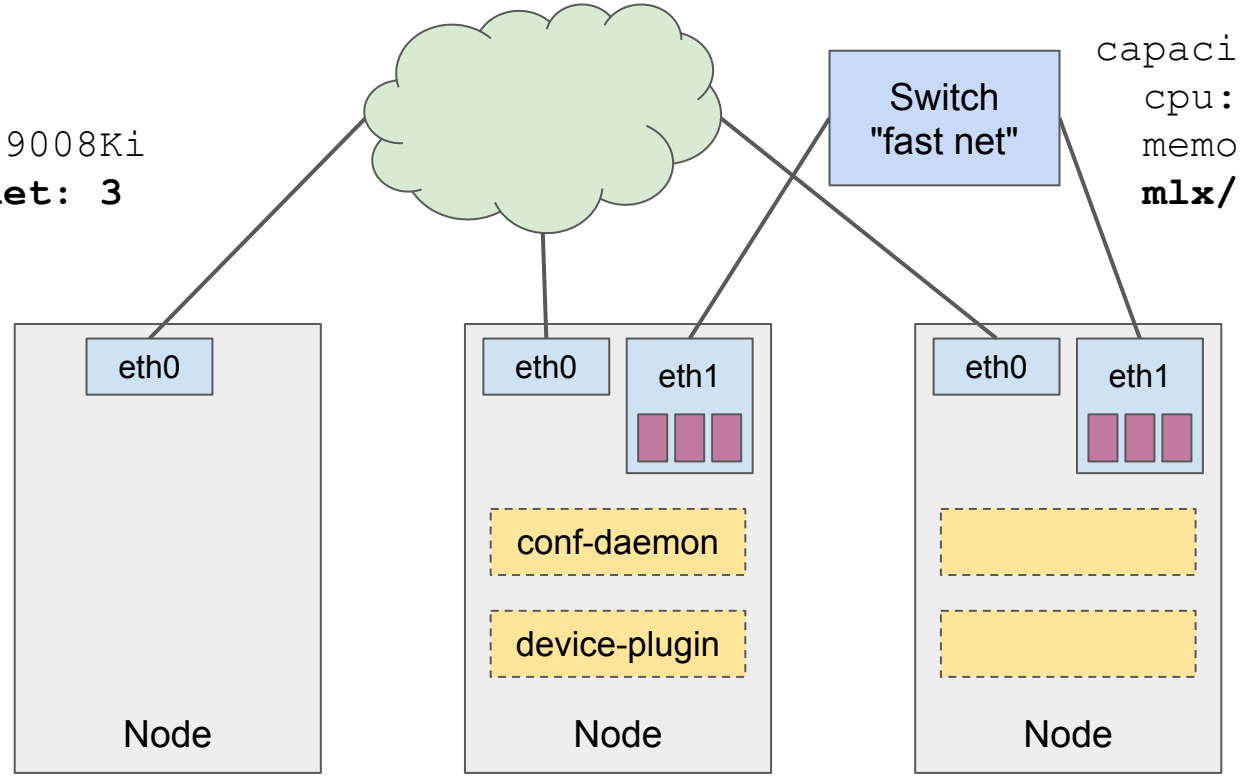
Fast Packet Processing (Advertisement)

```
capacity:
  cpu: 2
  memory: 2049008Ki
  mlx/sriov_net: 3
```



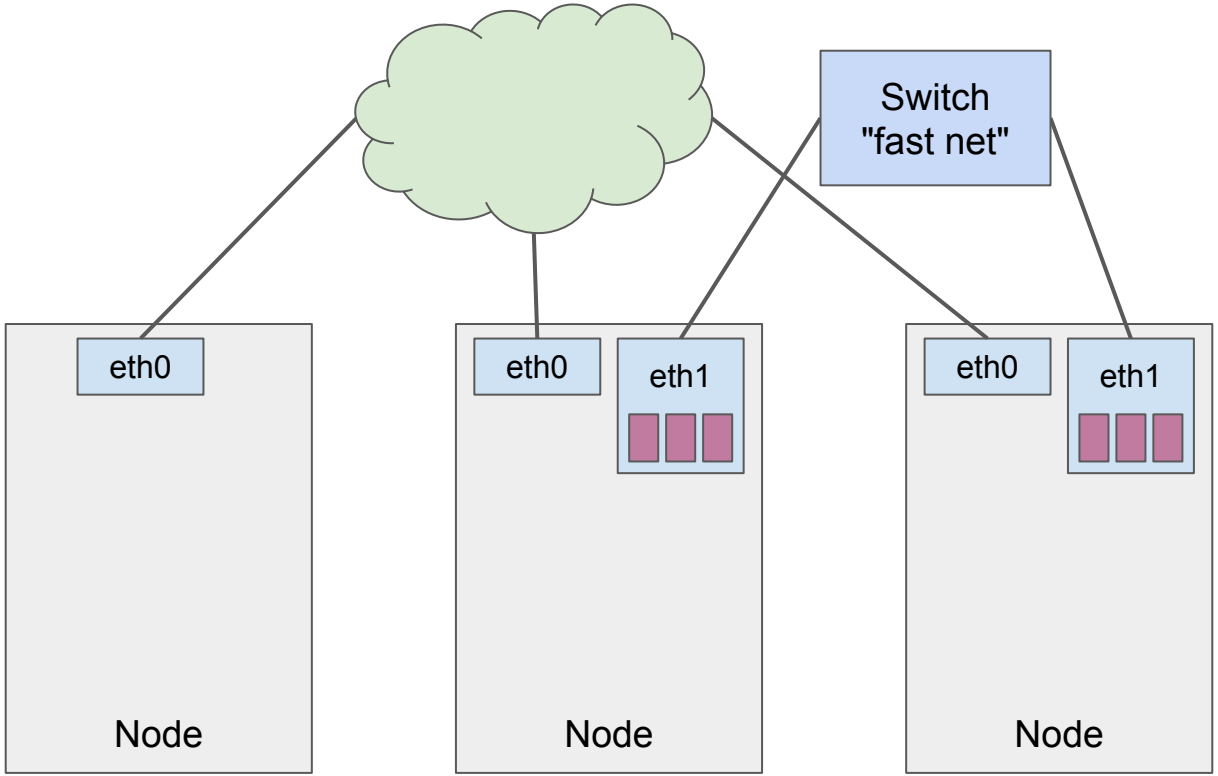
Fast Packet Processing (Advertisement)

```
capacity:
  cpu: 2
  memory: 2049008Ki
  mlx/sriov_net: 3
```

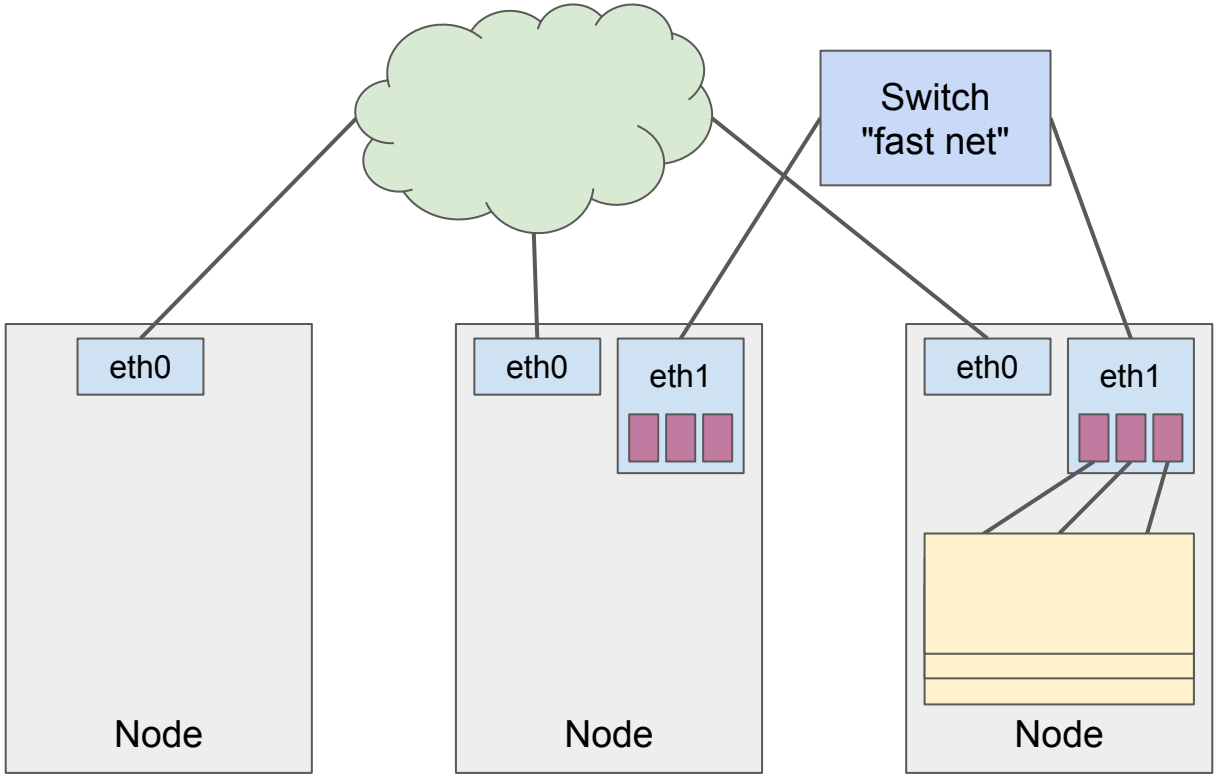



```
capacity:
  cpu: 2
  memory: 2049008Ki
  mlx/sriov_net: 3
```

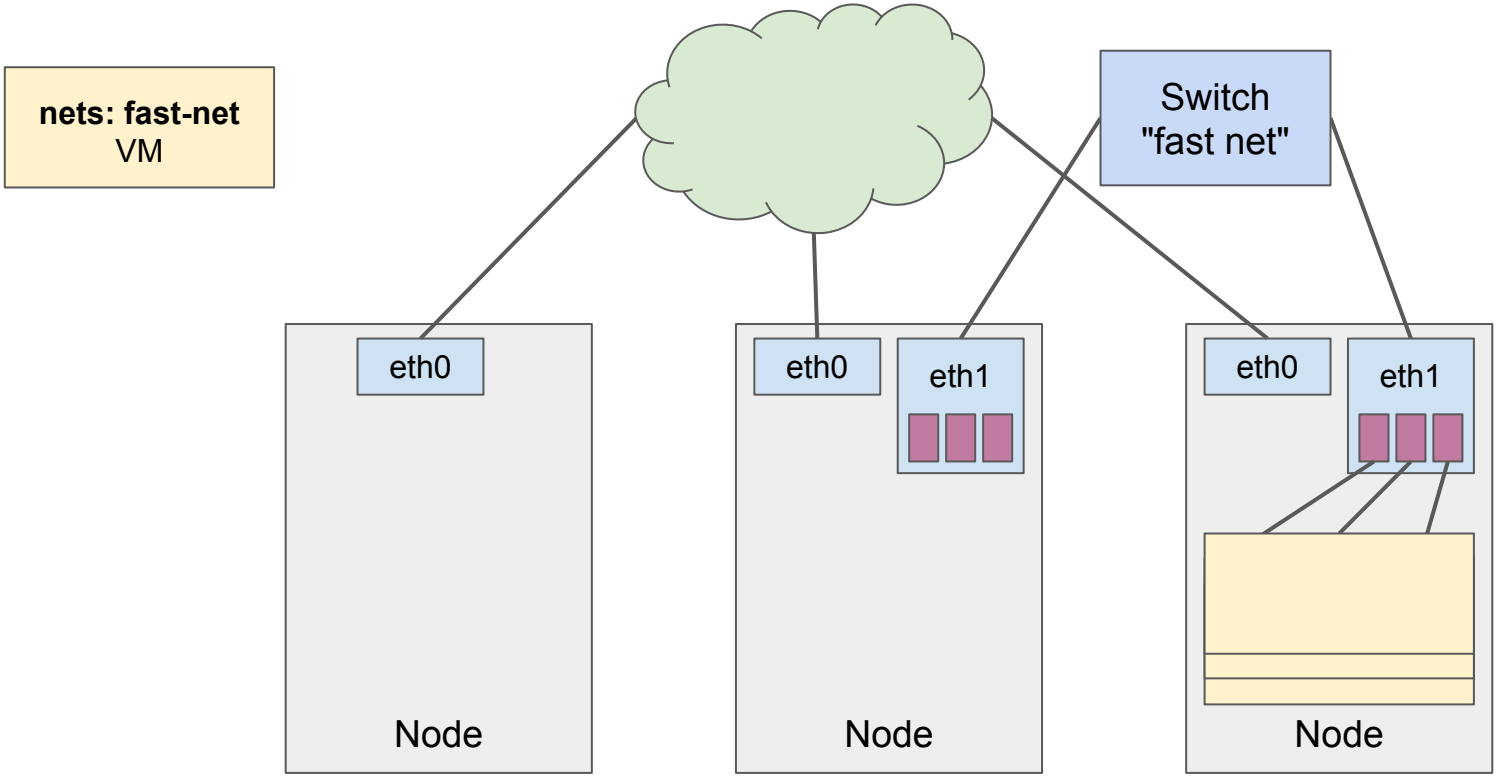
Fast Packet Processing (Scheduling)



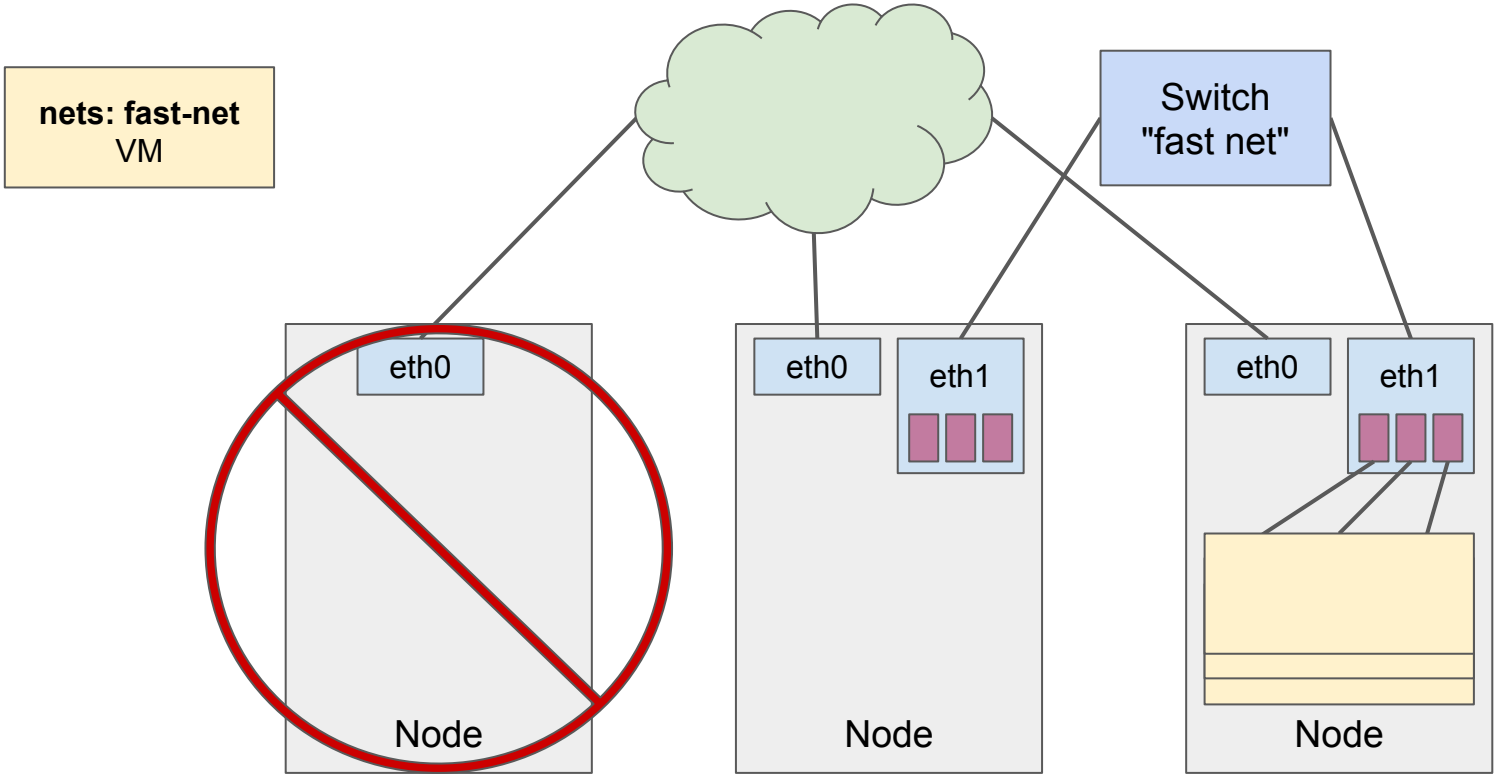
Fast Packet Processing (Scheduling)



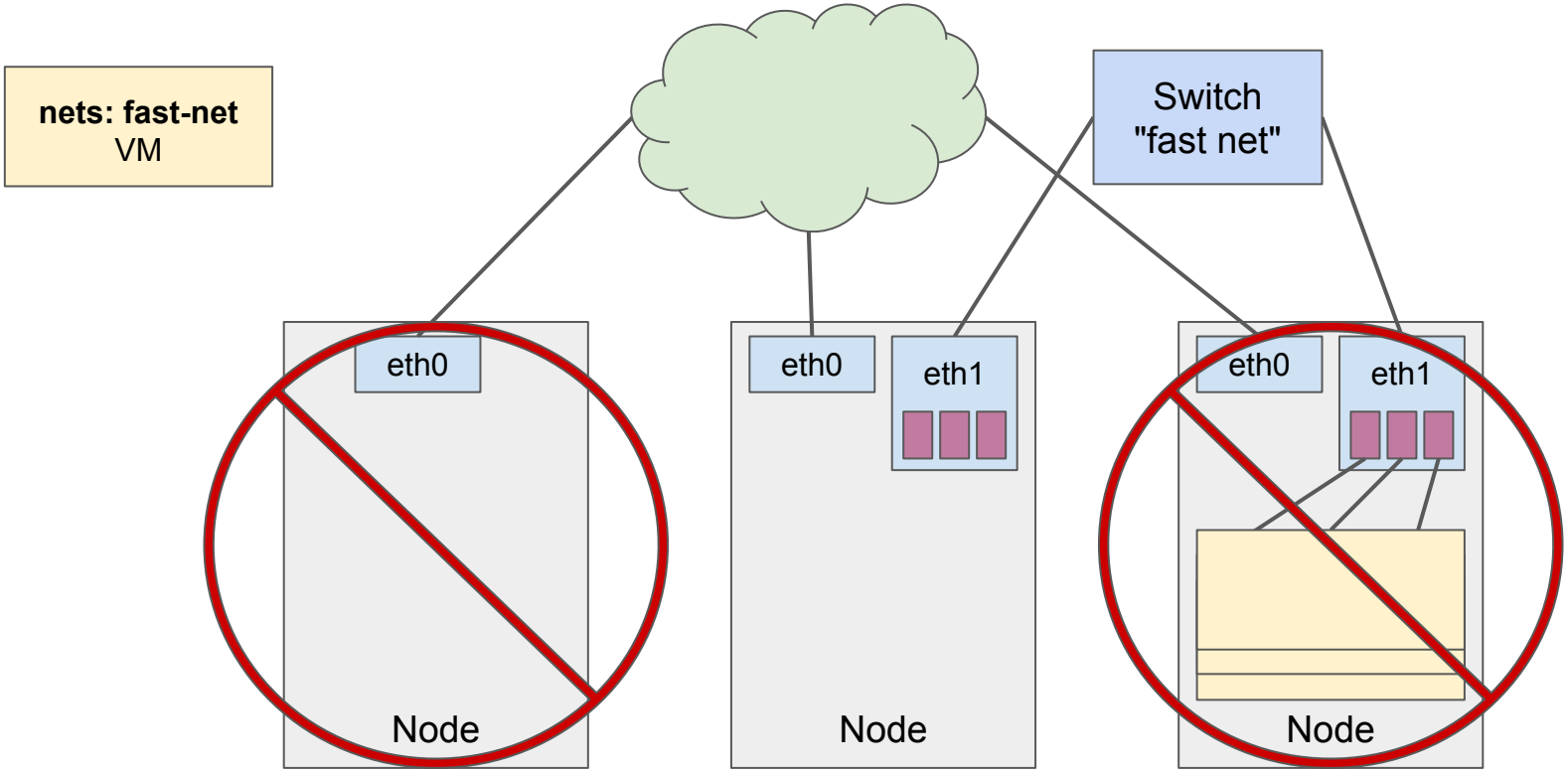
Fast Packet Processing (Scheduling)



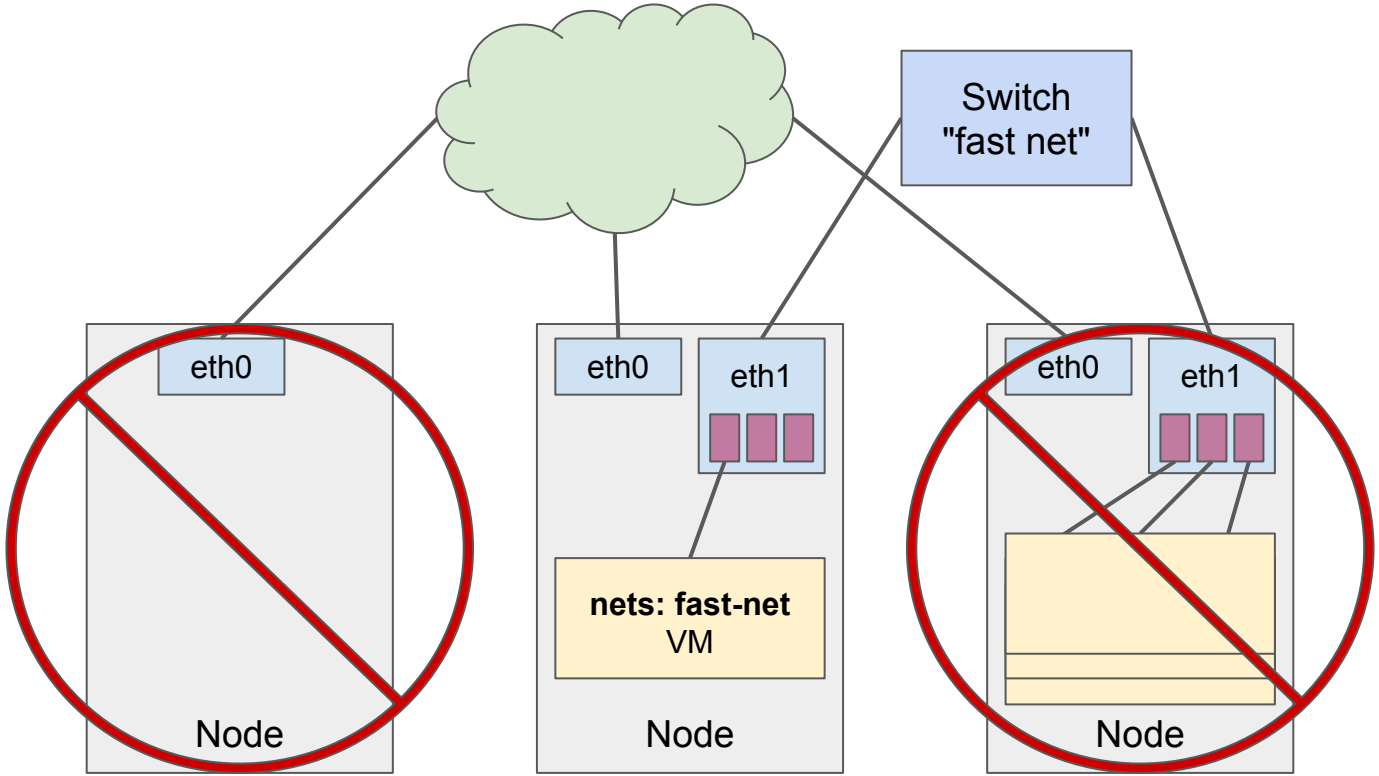
Fast Packet Processing (Scheduling)



Fast Packet Processing (Scheduling)



Fast Packet Processing (Scheduling)

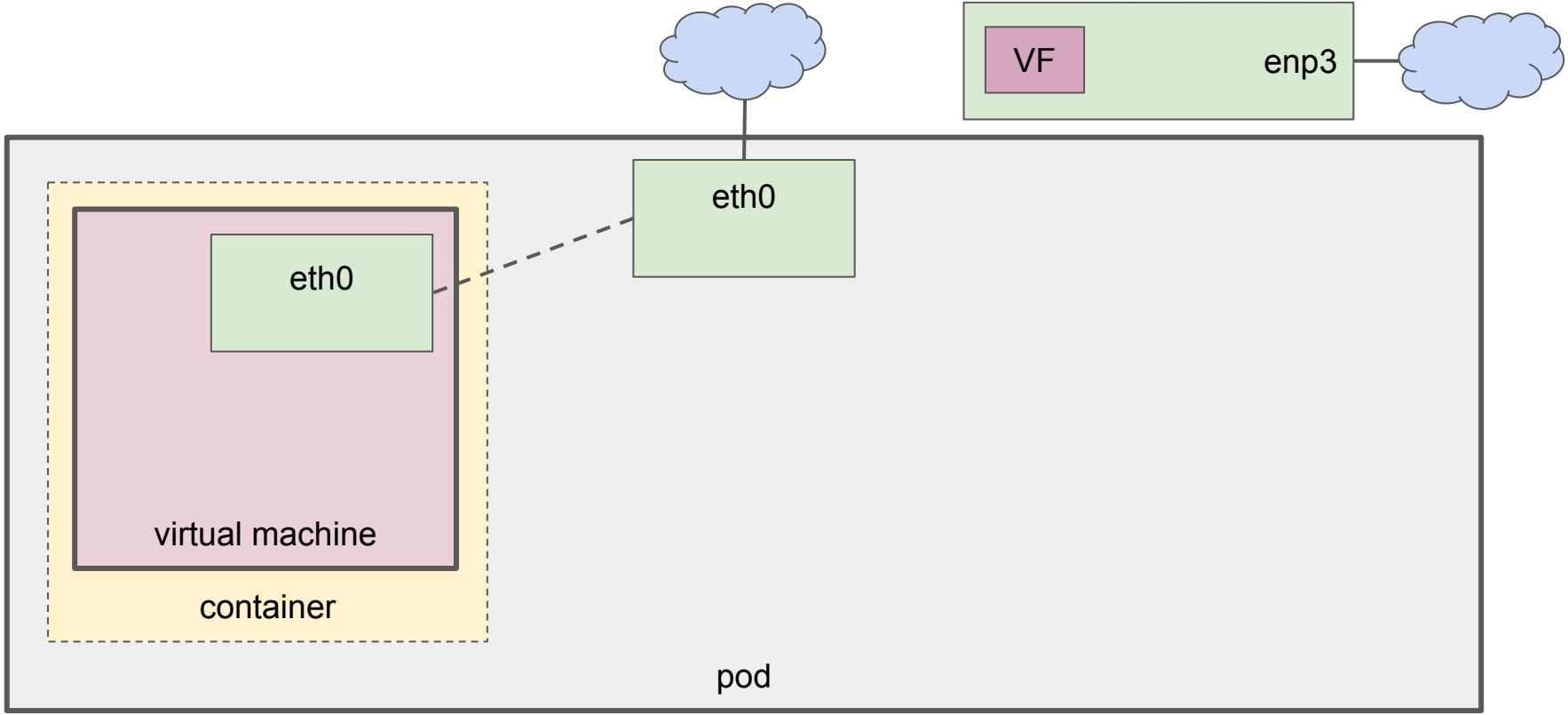


Fast Packet Processing (Attachment)

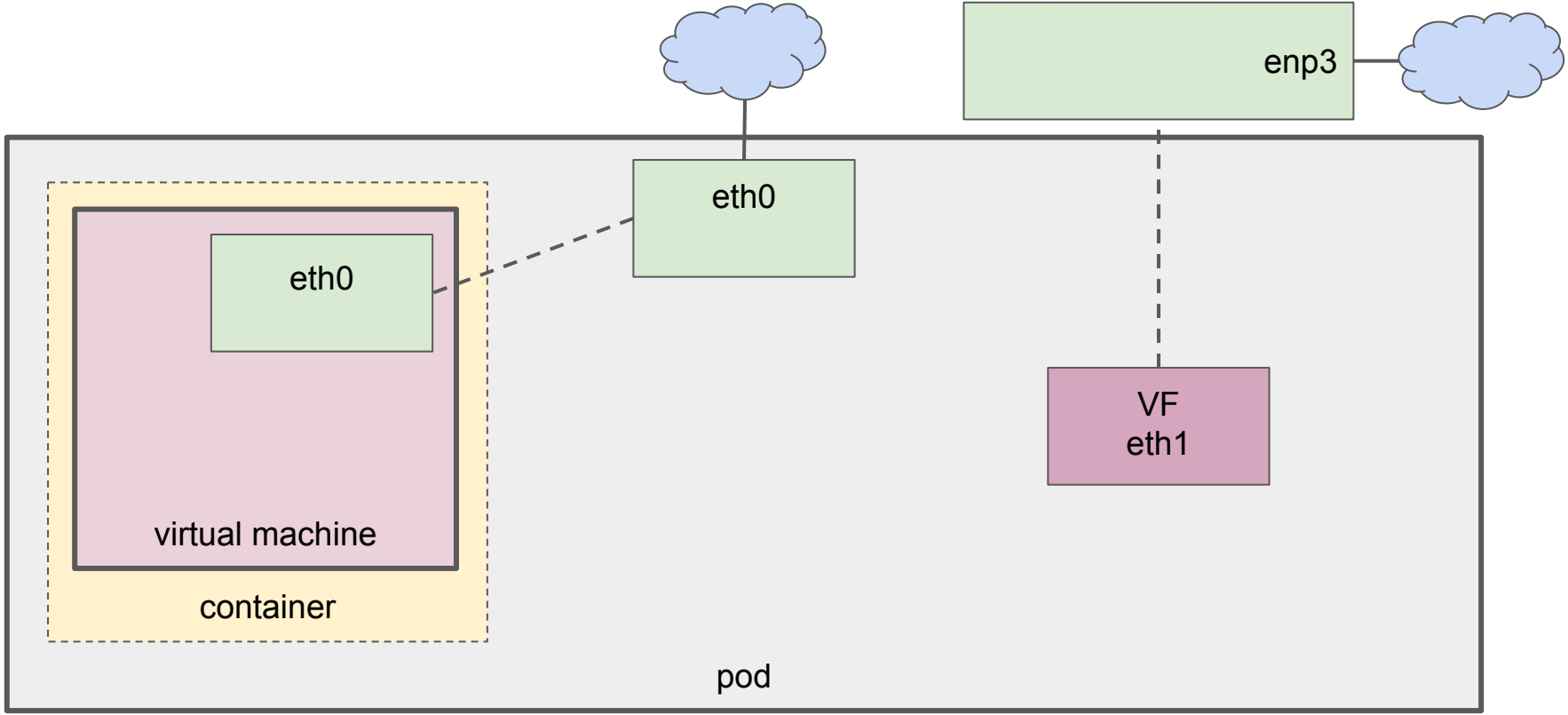
- After a Pod requesting SR-IOV gets scheduled
- device plugin would tell kubelet how to pass VF into a container,
- kubelet plugs in the VF,
- Multus fetches ID of the VF and passes it to SR-IOV CNI,
- SR-IOV CNI then configures the VF.

- Difference between netlink and vfio-pci method.

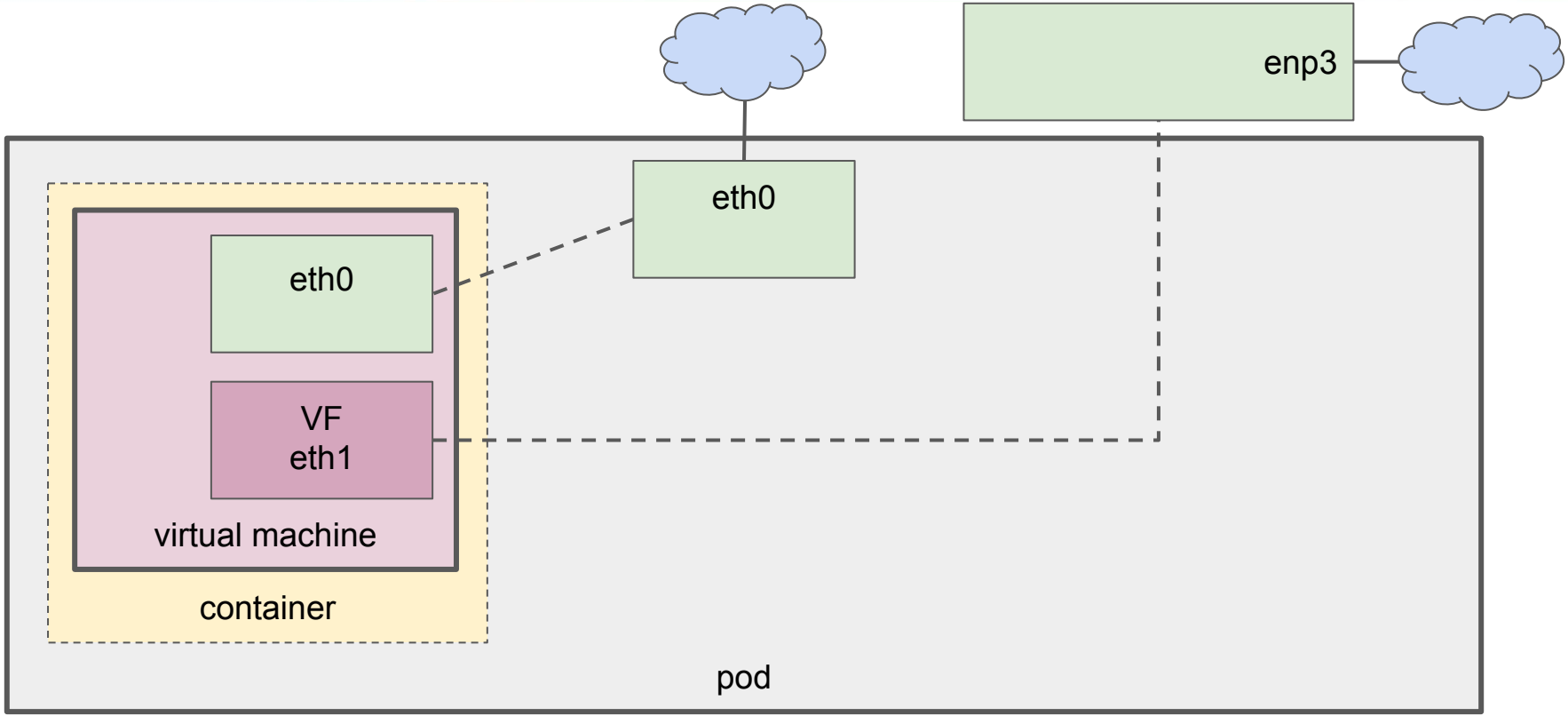
Fast Packet Processing (Binding)



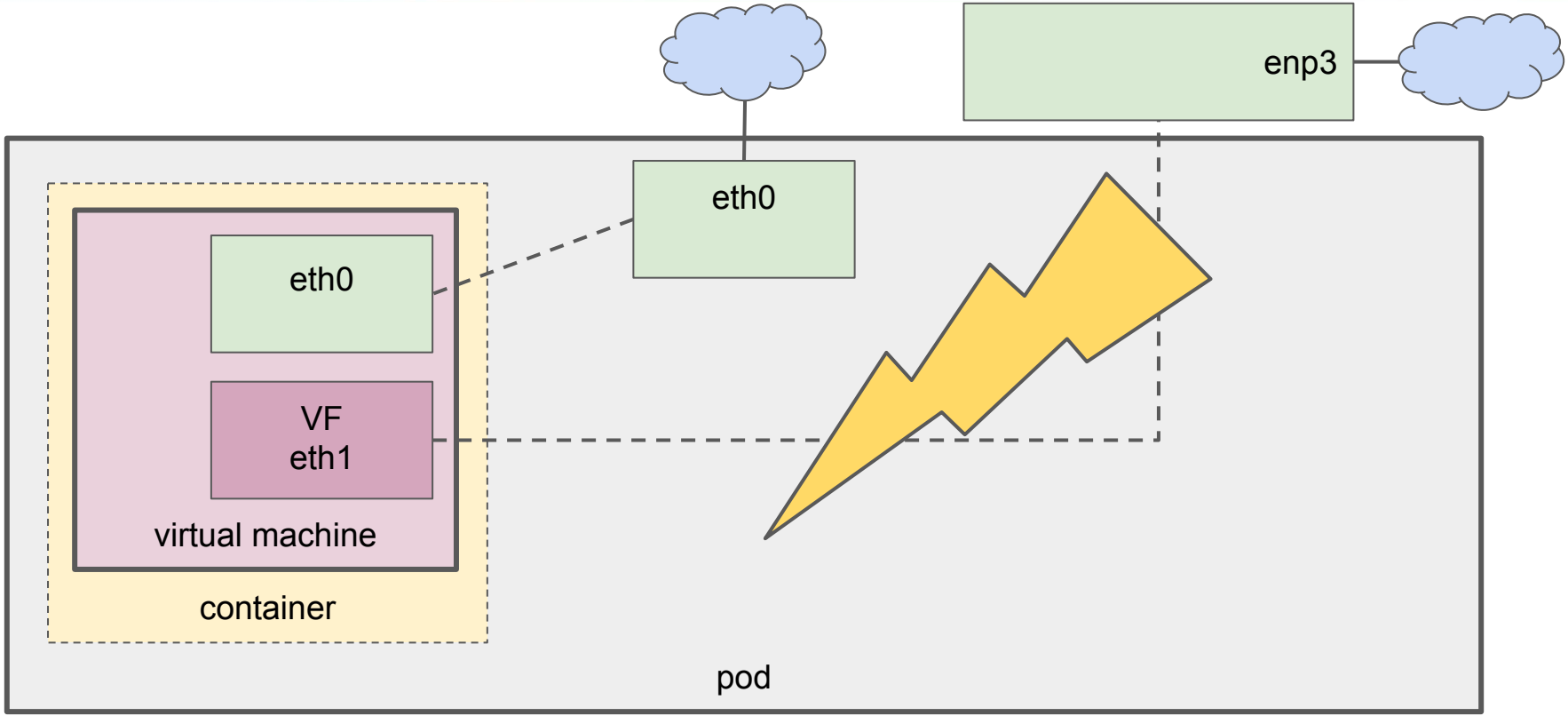
Fast Packet Processing (Binding)



Fast Packet Processing (Binding)



Fast Packet Processing (Binding)



Fast Packet Processing (Tweaking)

- SR-IOV can be made even faster with NUMA awareness and CPU management enabled on the cluster,
- the goal is to align all resources of a VM as close together as possible,
- KubeVirt leverages CPU manager when dedicated CPU placement is enabled.

Fast Packet Processing (DPDK)

- Very fast user-space networking,
- requires hugepages,
- hugepages can be propagated through Kubernetes and KubeVirt to the guest.

Fast Packet Processing (Example)

```
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovNetworkNodePolicy
metadata:
  name: policy-mlx
  namespace: sriov-network-operator
spec:
  deviceType: vfio-pci
  mtu: 9000
  nicSelector:
    deviceID: "1017"
    vendor: "15b3"
    pfNames:
    - ens801f0
  nodeSelector:
    feature.node.kubernetes.io/network-sriov.capable: "true"
  numVfs: 6
  resourceName: fast-network
```

Fast Packet Processing (Example)

```
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovNetwork
metadata:
  name: fast-network-10
  namespace: sriov-network-operator
spec:
  ipam: |
    {}
  vlan: 10
  spoofChk: "off"
  resourceName: fast-network
  networkNamespace: default
```

Fast Packet Processing (Example)

```
apiVersion: kubevirt.io/v1alpha3
kind: VirtualMachineInstance
metadata:
  name: vmi-test
  ...
spec:
  domain:
    cpu:
      sockets: 3
      cores: 1
      threads: 1
      model: host-model
      dedicatedCpuPlacement: true
      isolateEmulatorThread: true
    memory:
      hugepages:
        pageSize: "1Gi"
    resources:
      limits:
        cpu: 3
        memory: 4Gi
    devices:
      interfaces:
      - name: fast
        sriov: {}
    networkInterfaceMultiqueue: true
  networks:
  - name: fast
    multus:
      networkName: fast-network-10
  ...
```

Questions?

References

[Hyperconverged Cluster Operator]
[KubeVirt]
[Cluster Network Addons Operator]
[SR-IOV Operator]
[SR-IOV Device Plugin]
[SR-IOV CNI]

kubernetes