Network interface hotplug for Kubernetes

February 5th, 2022, FOSDEM Virt & IaaS Dev Room

Miguel Duarte Barroso
mdbarroso@redhat.com
https://github.com/maiqueb
Agenda

● Introduction
  ○ CNI
  ○ Multus
  ○ KubeVirt

● Motivation, problem, and Goals

● Implementation
  ○ Multus
  ○ KubeVirt

● PoC demo

● Conclusions

● Next steps
Introduction
Kubernetes networking model

- pods on a node can communicate with all pods on all nodes without NAT
- agents on a node can communicate with all pods on that node
- pods in the host network of a node can communicate with all pods on all nodes without NAT

https://kubernetes.io/docs/concepts/cluster-administration/networking/#the-kubernetes-network-model
CNI - how does it work?

```
{
  "name": "any_name",
  "cniVersion": "0.1.0",
  "type": "calico",
  "kubernetes": {
    "kubeconfig": "/path/to/kubeconfig"
  },
  "ipam": {
    "type": "calico-ipam"
  }
}
```

$ KUBEVIRTCI_RUNTIME=podman cluster-up/ssh.sh node01
  "ls -lah /opt/cni/bin"

...  
-rw-r-xr-x. 1 root root  35M Nov 15 09:12 calico
-rw-r-xr-x. 1 root root  35M Nov 15 09:12 calico-ipam
Multus

- Meta CNI plugin
- Enables multiple interfaces per pod
- N to N interface to network association

https://github.com/k8snetworkplumbingwg/multus-cni/
Multus - how to use

The specification uses annotations to call out a list of intended network attachments as “additional networks”, or “secondary networks”

CNI network configurations are packed inside CRD objects.
Multus

Pod without Multus

Pod

eth0

flannel

Flannel CNI

Kubernetes

Pod with Multus

Pod

eth0

net0

Flannel (default)

macvlan

Flannel CNI

macvlan CNI

Multus CNI

Kubernetes
KubeVirt

- Virtual machine add-on for Kubernetes
- Libvirt / qemu running within a kubernetes pod
- Common platform for virt / containers
- Use cases
  - Migration path from VM workloads to containerized solution
    - Decompose VMs to containers
  - Centralized development workflow
  - Centralized operations
KubeVirt architecture

- kubectl (user commands)
- API Server
  - virt-controller
  - virt-api
- Node
  - kubelet
  - Pod (DaemonSet)
    - KubeVirt agent
  - Pod (per VMI)
    - Launcher process
    - libvirtd
    - qemu
- docker (runtime)
  - Pod
    - container
Motivation & Goals
Motivation, Problem, and Goals

- Motivation
  - (some) VMs cannot tolerate a restart when attaching / removing networks
  - Workload created prior to the network
  - Add / remove nics to running VMs is an industry standard available in multiple platforms
Motivation, Problem, and Goals

- **Motivation**
  - (some) VMs cannot tolerate a restart when attaching / removing networks
  - Workload created prior to the network
  - Add / remove nics to running VMs is an industry standard available in multiple platforms

- **Problem**
  - Dynamic attachment of L2 networks *without* restarting the workload (pod / VM)
Motivation, Problem, and Goals

- **Motivation**
  - (some) VMs cannot tolerate a restart when attaching / removing networks
  - Workload created prior to the network
  - Add / remove nics to running VMs is an industry standard available in multiple platforms

- **Problem**
  - Dynamic attachment of L2 networks *without* restarting the workload (pod / VM)

- **Goals**
  - Adding network interfaces to running VMs
  - Removing networking interfaces from running VMs
  - A VM can have multiple interfaces connected to the same (secondary) network(s).
  - The previous goals also target pods - not only VMs
Implementation
Multus
Multus controller

Kubernetes objects → watch → Pod Controller

update
Multus changes

CNI ADD / DEL

Multus-shim (on host)

JSON

```
{
  "containerID": "xyz",
  "command": "ADD",
  "netnsPath": "/var/run/netns/…",
  ...
}
```

Multus-controller (daemon; on pod)

deploy

pod
Multus changes

CNI ADD / DEL

Multus-shim (on host)

JSON
{
  "containerID": "xyz",
  "command": "ADD",
  "netnsPath": "/var/run/netns/…",
  ...
}

Multus-controller (daemon; on pod)

pod

delegate
Multus changes - adding a pod controller

CNI ADD / DEL

Multus-shim (on host)

JSON

```
{
    "containerID": "xyz",
    "command": "ADD",
    "netnsPath": "/var/run/netns/…",
    ...
}
```

Pod networks annotation changed

Multus-controller (daemon; on pod)

pod
Multus changes - adding a pod controller

CNI ADD / DEL

Multus-shim (on host)

JSON
{
  "containerID": "xyz",
  "command": "ADD",
  "netnsPath": "/var/run/netns/…",
  ...
}

Pod networks annotation changed

Multus-controller (daemon; on pod)

delegate

pod
KubeVirt
Current pod network diagram
Hotplug request sent via CLI

```
addinterface vmi-xyz
--net tenantANetwork
--ifaceName=eth1000

PUT apis/subresources.kubevirt.io/v1/namespaces/default/virtualmachineinstances/vmi-xyz/addinterface
{
    "name": "tenantA",
    "interfaceName": "eno141"
}
```

```
PATCH apis/./virtualmachineinstances/
[
    {"op": "test", "path": "/spec/networks", "value": <oldNetworkJSON>},
    {"op": "add", "path": "/spec/networks", "value": <newNetworkJSON>},
    {"op": "test", "path": "/spec/domain/devices/interfaces", "value": <oldInterfaceJSON>},
    {"op": "add", "path": "/spec/domain/devices/interfaces", "value": <newInterfaceJSON>}
]```
Mutating the pod - hot plug into pod

Mutating pod networks annotations

controller

- VM spec updates

 PATCH pod annotations
`k8s.v1.cni.cncf.io/networks: [{ "ifaceName": "netXYZ", "name": "tenantANetwork", ... }, ...]`

multus
POD: Multus adds new interface
Mutating the VM status

controller

PATCH apis/.../virtualmachineinstances
[
  { "op": "test", "path": "/status/interfaces", "value": "<old interfaces JSON>" },
  { "op": "replace", "path": "/status/interfaces", "value": "<new interfaces JSON>" }
]

KubeVirt agent
Creating auxiliary pod networking infrastructure

- KubeVirt agent
- VM pod
- vmi status update
- prepare pod networking
KubeVirt’s agent setups pod networking infra
KubeVirt’s agent reconciles the VM
Plug interface into domain

- Libvirt’s attach / detach device API call

```bash
$ virsh attach-device --domain <domain-name> --file /dev/stdin --live <<EOF
  <interface type='ethernet'>
    <mac address=<tap device MAC/>/
    <target dev=<tap name> managed='no'/>/
    <model type='virtio-non-transitional'/>
    <mtu size='1480'/>/
  </interface>
EOF

$ virsh detach-device --domain <domain-name> --file /dev/stdin --live <<EOF
  <interface type='ethernet'>
    <mac address=<tap device MAC/>/
    <target dev=<tap name> managed='no'/>/
    <model type='virtio-non-transitional'/>
    <mtu size='1480'/>/
  </interface>
EOF
```
Pod’s networking diagram *after* hot-plug
Machine type limitations

**Q35 machine type** (modern machine type)

- Supports hotplug of a *single* interface (passthrough or emulated)
- What if we want need more?
  - Add a suitable number of pcie-root-port controllers when defining the guest
- Solution:
  - Expose a knob (VM) to specify the number of PciE root port controllers
  - `domain.devices.numberPciPorts`
  - Mimicked Openstack Nova implementation
Demos

- **Hotplug**
  - Q35 machine type out of the box
  - Q35 machine type w/ 24 PciE root controllers
- **Hotunplug**
Demos

- **Hotplug**
  - [Q35 machine type out of the box](#)
  - [Q35 machine type w/ 24 PciE root controllers](#)
- **Hotunplug**
Demos

- **Hotplug**
  - Q35 machine type out of the box
  - Q35 machine type w/ 24 PciE root controllers
- **Hotunplug**
Conclusions

- Hot plug / unplug network interfaces into a VM
  - First requires the interface to be plugged into the pod

- Hot plug / unplug network interfaces into a pod
  - Requires multus
  - Unplug affects only interfaces added via multus - cluster default network is *off limits*.

- (some) Machine types require more changes - q35 / PciE root port controllers
Next steps
● Productify the PoC
  ○ Merge the multus code
    ■ Thick plugin refactor
    ■ React to cni cncf network annotation updates
  ○ Merge the KubeVirt code
    ■ Hot plug / unplug feature
Thank you !!!
Resources

- **KubeVirt**
  - Network interface hotplug for KubeVirt design document

- **CNI**
  - Intro to CNI
  - CNI deep dive

- **Multus**
  - Kubernetes Network Custom Resource Definition De-facto Standard