One SDN to connect them all

February 3rd, 2024, FOSDEM Virt & Cloud Dev Room

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

KubeVirt
Agenda

● Motivation & problem we’re trying to solve
● Introduction
● Implemented use cases + Demos
● Roadmap
● Lessons learnt
Motivation & Problem
https://kubernetes.io/docs/concepts/services-networking/#the-kubernetes-network-model
For ingress: use services / ingress types
https://kubernetes.io/docs/concepts/services-networking/#the-kubernetes-network-model
Motivation

No batteries to access stuff on the physical network

Default cluster network is not suited for all use cases

Mix and match of technologies to implement use cases is expensive

https://www.pinterest.es/pin/213921051038330829
Objectives

Liberate the cluster / network admins

Push complexity to a programmable network

Multiple use-cases in a single plugin

https://www.pinterest.com/pin/465207836504568697
Intro
KubeVirt

- Kubernetes plugin
  - Try it out
- Runs VMs alongside pods in the same platform (Kubernetes)
- Each pod runs libvirt + qemu process for the VM
- VM networking requirements >> pod networking requirements
OVN / OVN-K

- **OVN** provides a higher-layer of abstraction than Open vSwitch
  - SDN
  - Open vSwitch orchestrator
  - Logical routers / logical switches, ACLs, etc rendered to openflow

- **OVN-Kubernetes => CNI plugin for Kubernetes**
  - Translates Kubernetes objects to OVN logical entities
    - secondary network => logical switch
    - Pod attachment => logical switch port
    - Network policy => port group + ACLs
Supported use cases
East / west communication
East / west mixed workload communication

- Network attachment configuration
- Workload definition
- Demo - Overlay mixed workloads
Access to the physical network
Configuring the underlay / physical network

- NMState & Kubernetes-nmstate
  - Kubernetes native
  - NodeNetworkConfigurationPolicy
Configuring the underlay / physical network

apiVersion: nmstate.io/v1
kind: NodeNetworkConfigurationPolicy
metadata:
  name: ovs-dedicated-bridge
spec:
  nodeSelector:
    node-role.kubernetes.io/worker: ""
  desiredState:
    interfaces:
      - name: ovs1
        description: separate bridge
        type: ovs-bridge
        state: up
        bridge:
          port:
            - name: ens4
  ovn:
    bridge-mappings:
      - localnet: default-network
        bridge: br-ex
      - localnet: tenantblue
        bridge: ovs1
Micro-segmentation
Physical network + micro-segmentation

- Policy manifest
- OVN northbound entities
- Demo - localnet topology + network policies
Roadmap
Next features

- **IPAM for virt workloads**
  - “Sticky” IPs for virt workloads

- **Selector policies for virt workloads**
  - Requires IPAM - OVN-Kubernetes must be aware of the IPs assigned to the workloads

- **Services for secondary networks**
  - Ingress
  - Egress

- **Self-service networks**
  - … create layer2 overlays without admin intervention
Lessons learnt

- Collaboration between different companies
  - Red Hat & NVIDIA
  - ... for exactly the same use cases
  - ... but with different scope in mind

- User experience
  - Physical network configuration was not originally in scope
    - NNCP & OVN bridge mappings API mitigated bad UX
  - Insert <JSON-in-yaml horror story> here
    - Vlan vs vlanID
    - The resource kind ...
    - Different clients have different behaviors - oc vs kubectl
Thank you! Questions ?...