K8s & NFV - Friends or Foes?

Feng Pan (RedHat)
Ivan Coughlan (Intel)
Gergely Csatari (Nokia)
Today's trends in telecom infrastructure

- NFV
- Virtualisation
- IT
- Cloud Native
- 5G
- 5G: Latency and capacity
- EDGE
- Distributed / Location aware infra
- Kubernetes
Deployment options

- Hardware
- IaaS / VM
- IaaS / Containers
- Application
5G: Pushing the limits to reach the next level
NFV Challenges

<table>
<thead>
<tr>
<th>Kubernetes Networking</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Data Plane Acceleration</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Enhanced Platform Awareness (EPA)</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Telemetry</th>
</tr>
</thead>
</table>
NFV Challenges

Multiple network interfaces for VNFs

Data Plane Acceleration

Enhanced Platform Awareness (EPA)

Telemetry
NFV Challenges

- Multiple network interfaces for VNFs
- High performance Data Plane (E-W)
- High performance Data Plane (N-S)

Enhanced Platform Awareness (EPA)

Telemetry
NFV Challenges

- Multiple network interfaces for VNFs
- High performance Data Plane (E-W)
- High performance Data Plane (N-S)
- Ability to request/allocate platform capabilities
- CPU Core-Pinning and isolation for K8s pods
- Dynamic Huge Page allocation
- Manage server platform devices with K8s
- Guarantee NUMA node resource alignment

Telemetry
NFV Challenges

- Multiple network interfaces for VNFs
- High performance Data Plane (E-W)
- High performance Data Plane (N-S)
- Ability to request/allocate platform capabilities
- CPU Core-Pinning and isolation for K8s pods
- Dynamic Huge Page allocation
- Manage server platform devices with K8s
- Guarantee NUMA node resource alignment
- Platform telemetry
CNI Intro

- Spec for i/f between runtime and plugins
- SW framework for dev of plugins
- Each CNI plugin is an executable

Runtime is responsible for:
- Creating the container net namespace
- Invoking the correct CNI plugin(s) in the correct order

Plugin is responsible for:
- Add or Del net i/f in container net namespace
- L3 configuration via CNI IPAM plugin

Kubernetes
- CNI is main integration point for pod networking
- Per kubelet CNI configuration
- Single implicit network

Set some environment variables and go!

```
CNI_COMMAND=ADD
CNI_CONTAINERID=$id
CNI_NETNS=/proc/$pid/ns/net
CNI_PATH=/opt/cni/bin
CNI_IFNAME=eth0
my-plugin < my-config
```

Source: https://www.slideshare.net/weaveworks/introduction-to-the-container-network-interface-cni
Device Plugin

- Framework for device vendors to provide support in k8s
- Dev Plugin discovers, advertises & provisions device resources
- Implemented as a daemon with gRPC interface
- Dynamically deployed with k8s daemonset
- Examples: GPU, FPGA, QuickAssist Technology, SR-IOV Network Device

Source: Kubernetes Device Plugin Proposal
Network Plumbing Working Group

- Founded during Kubecon 2017 as part of SIG Network to address lower level networking issues in Kubernetes
- Currently focused on multiple network attachments, in an out-of-tree solution.
- Gather use-cases and propose standard spec
- Implement reference plugin
- Will expand to further advanced networking use-cases
The specification uses annotations to call out a list of intended network attachments as “sidecar networks”.

Network Attachment Definition CRD
As currently proposed by Network Plumbing Working Group.

Pod annotations
```
apiVersion: v1
crds:
  - apiVersion: v1
    kind: Pod
    metadata:
      name: pod_c
      annotations:
        kubernetes.cni.cncf.io/networks:
          - name: "control"
          - name: "data"
spec:
  containers: [...]
```

CRD Object
```
Name:         data
Namespace:    default
Labels:       <none>
Annotations:  <none>
API Version:  cni.cncf.io/v1
Args:         

Kind:         NetworkAttachmentDefinition
Plugin:       macvlan
Metadata: [...]
```
**Pod without Multus**

- **Pod** with eth0 and flannel
- Flannel CNI
- Kubernetes

**Pod with Multus**

- **Pod** with eth0, net0, Flannel (default), and macvlan
- Flannel CNI, macvlan CNI
- Multus CNI
- Kubernetes
Multus

- Implemented in community by Intel, Red Hat and others
- Conforms to Network Plumbing Working Group V1 Spec
- Supports variety of delegated CNIs (macvlan, vlan, SR-IOV, userspace…)
- Red Hat is working on adding Multus support in OpenShift
SR-IOV Networking with Multus

- Multus as CNI plugin
- SR-IOV device plugin for VF device discovering, device state report, allocation, health checking
- Kubelet updates pod to DeviceID mapping
- Multus CNI retrieves DeviceID and pass it along with CNI configuration to SR-IOV CNI
- SR-IOV CNI move VF to POD namespace
**DP Acceleration: Userspace CNI**

**Challenge:**
- Userspace vSwitches being used to accelerate data plane
- How to leverage in container environment

**Solution:**
- Data Plane Acceleration with Userspace CNI plugin
- Enables vhost-user & memif interfaces in k8s pod
EPA: NUMA Awareness

Challenge:
• Non-optimal NUMA aware resource allocation can severely impact performance of latency sensitive workloads

Solutions:
• K8s NUMA Manager Proposal and PoC
• Initially addresses CPU Mgr and Dev Plugin
• Positive indication of support in upcoming release of k8s
Conclusion

**F.R.I.E.N.D.S.** but we need to work on the friendship

- With CNI plugins following the network plumbing groups specs, like Multus we can run VNFs using SCTP MH or implementing router functionality
- With the SR-IOV and the user space CNI plugins we can run VNFs with high network bandwidth need
- With NUMA awareness we can run VNFs with low I/O latency requirements
- Using the device plugins we are able to utilize GPUs and FPGAs for VNFs dealing with special workload
Call for action

• This is an ongoing effort in the community including Intel, RedHat and Nokia
• Our target is to run VNFs on Kubernetes using upstream components and no hacking
• If you have similar dreams, join us
• Mailing list: https://groups.google.com/d/forum/kubernetes-npwg
• fp@redhat.com, ivan.coughlan@intel.com, gergely.csatari@nokia.com
Links

• SIG Network: https://github.com/kubernetes/community/tree/master/sig-network
• Github
  • Kubernetes: https://kubernetes.io/ - https://github.com/kubernetes/kubernetes
  • K8s Dev Plugin: https://kubernetes.io/docs/concepts/extend-kubernetes/compute-storage-net/device-plugins/
  • CNI: https://github.com/containernetworking/cni
  • Multus: https://github.com/intel/multus-cni
  • SRIOV Device Plugin: https://github.com/intel/sriov-network-device-plugin
  • Intel Device Plugin Repo: https://github.com/intel/intel-device-plugins-for-kubernetes
  • SRIOV CNI: https://github.com/intel/sriov-cni
  • Userspace CNI: https://github.com/intel/userspace-cni-network-plugin
• RedHat OKD: https://github.com/openshift/origin
• Intel Experience Kits: https://networkbuilders.intel.com/network-technologies/container-experience-kits
• NUMA Manager Proposal: https://github.com/kubernetes/community/pull/1680
• NUMA Manager PoC: https://github.com/lmdaly/kubernetes/tree/dev/numa_manager
Thank You

Questions?