Kubernetes networking in the telco space

LFN Developer Forum
Gergely Csatari (using the materials of Robert Springer, and Peter Megyesi with the help of Levente Kale, Laszo Janosi and Gergo Huszty)
26-03-2018
Whoami: Gergely Csatari
Nokia
gerely.csatari@nokia.com
csatari on freenode
@GergelyCsatari on Twitter
Basic overview of Kubernetes networking

Most prominent CNI plugins

Features important for Telcos

Solutions
Kubernetes networking basics

Diagram showing the networking architecture for two pods, Pod 1 and Pod 2, within their respective network namespaces (net ns) and how they connect to the host. Each pod has an eth0 interface, and within the pods, there are veth pairs (vethnn and vethyy) connected to the cbr0 device, which is located in the root net ns. The diagram illustrates the connectivity between the host, pods, and network namespaces.
Kubernetes networking basics

1. Kubernetes
2. Pod network ns
3. CNI plugin
4. IPAM plugin

- Create
- Add
- Add veth
- Added
- Do magic
- Get IP
- IP
- Added
Kubernetes networking basics

Kubernetes

Delete

Pod network ns

Remove veth

CNI plugin

Free IP

OK

IPAM plugin

Undo magic

Deleted

Okay
Most relevant reference CNI plugins

**bridge**
Creates a bridge, adds the host and the container to it.

**ipvlan**
Creates a new IP address on the host interface, forwards all traffic of that to the container.

**macvlan**
Creates a new MAC address, forwards all traffic of that to the container.

**dhcp**
Runs a daemon on the host to make DHCP requests on behalf of the container.
Most relevant CNI plugins for production

**Flannel**

Uses VXLAN tunnels between the hosts using a kernel implementation. Flannel uses etcd to store metadata.

**Calico**

Provides integration to Calico what defines BGP agents and advertises the pod subnets to the fabric.

**Nuage Networks**

Provides integration to Nuage, the highly scalable policy-based Software-Defined Networking (SDN) platform. Nuage uses the open source Open vSwitch for the data plane along with a feature rich SDN Controller built on open standards.

**Weave**

An overlay based generic networking solution for containers.

**Multus**

A CNI plugin to cascade other CNI plugins.
Things what our telco workload misses from these
Support for multiple interfaces

**What?** Some pods should have two or more interfaces.

**Why?** Support for load balancers for telco protocols, SCTP MH, separation of different traffic types.
Things what our telco workload misses from these
Fixed IP address

**What?** It should be possible to manually set the IP address of some pods.

**Why?** These are well known IP addresses distributed in configuration.
Things what our telco workload misses from these Overlay and NAT less

What? There should be no overlay used to implement the host-to-host communication and there should be no NAT used in the hosts.

Why? Both NAT-ting and overlay networks introduce extra latency into packet handling. For radio handling applications a ms delay can be serious. There are protocols which can not be used together with NAT.
Things what our telco workload misses from these SR-IOV support

**What?** The CNI plugin should be able to utilize SR-IOV capabilities of the host if there are any. This should be possible without NIC vendor lock in.

**Why?** SR-IOV makes packet processing faster and in telco we need fast packet processing.
Things what our telco workload misses from these DPDK

What? The CNI plugin should be able to utilize the DPDK capabilities of the host if there are any.

Why? DPDK makes packet processing faster and in telco we need fast packet processing.
Things what our telco workload misses from these Cloud compliancy

What? The CNI networking solution should not state any special networking requirements to the underlying cloud infrastructure in case of Kubernetes is running on top of VM-s of a cloud infra.

Why? Our solutions are deployed to both VM based clouds and to bare metal and we would like to support both of them using the same software stack.
Things what our telco workload misses from these
Firewall or network policy support

What? The CNI plugin should support explicit firewall rules, static and policy based routing to control the traffic between the different pods.

Why? There should be a way to control the traffic between the pods.
Multiple interfaces There are several discussions in kubernetes-sig-network and there are concrete plans for 2018. Intels Multus provides a solution to cascade different CNI plugins.

SR-IOV and DPDK There is an CNI plugin for DPDK with SR-IOV support. There is a vhostuser CNI plugin which can support DPDK accelerated OvS or VPP.

Network policies Calico can be run in policy enforcement mode what is also called Canal.
Originally it was built as part of the container infrastructure of one Nokia VNF.

Uses ipvlan for network segregation. There is no overlay or NAT.

Supports both VLAN and VxLAN interfaces.

Can cascade other CNI plugins, this is used for SR-IOV.

Has its own ipam plugin which supports fixed IP-s and IP routes.

Works based on the annotations in the pod manifest.
Solutions
Nokia danm example

**IP a&b** – These IPs are created without VLAN and mapped to Pods 1&2.

**IP c&d** – These IPs are created with VLAN ID. The VLAN tagging happens on the Host side.

**IP e&f** – These IPs are created with different VLAN IDs and placed on different NICs. This mode is useful for SCTP traffic.

**IP g&h&i** – These IPs are created with VXLAN (not simple VLAN) and placed above a team interface which provides redundancy. This mode is useful for container internal traffic.
Q&A
References

1. https://kubernetes.io/docs/concepts/cluster-administration/networking/