Troubleshooting the Network Overlay

Lab 19

What are you Learning?

In this lesson you'll learn how to troubleshoot network connectivity within Kubernetes clusters, including the CNI plugin.

Why is it important?

Because of the distributed nature of Kubernetes, networking is difficult. CNI, or Container Network Interface, is the contract that plugin developers use to make sure that distributed systems like Kubernetes can properly communicate. These plugins automate the creation of network routing over technologies like iptables, ipvs, or even OVN. Essentially there are 3 networks in play in a Kubernetes cluster, that may not overlap. The host network, the pod network and the services network. The CNI plugin is responsible for the latter two. Some CNI plugins also support the Kubernetes Network Policy API. This superset of network functionality acts generates micro-firewall rules between components running within Kubernetes.

Troubleshooting the Kubernetes Network

1. Make sure all of your ports are configured properly.
2. Next you can test the overlay network.
3. Create a Daemonset component using the busybox image.

```
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: overlaytest
spec:
  selector:
      matchLabels:
        name: overlaytest
  template:
    metadata:
      labels:
        name: overlaytest
    spec:
      tolerations:
      - operator: Exists
      containers:
      - image: busybox:1.28
        imagePullPolicy: Always
        name: busybox
        command: ["sh", "-c", "tail -f /dev/null"]
        terminationMessagePath: /dev/termination-log
```

4. Launch it using `kubectl create -f ds-overlaytest.yml`

5. Once the daemonset has rolled out, run this command. It will let each container ping the others.

```
echo "=> Start network overlay test"; kubectl get pods -l
name=overlaytest -o jsonpath='{range .items[*]}{@.metadata.name}{"
"}{@.spec.nodeName}{"\n"}{end}' | while read spod shost; do kubectl
get pods -l name=overlaytest -o jsonpath='{range
.items[*]}{@.status.podIP}{" "}{@.spec.nodeName}{"\n"}{end}' | while
read tip thost; do kubectl --request-timeout='10s' exec $spod --
/bin/sh -c "ping -c2 $tip > /dev/null 2>&1"; RC=$?; if [ $RC -ne 0
]; then echo $shost cannot reach $thost; fi; done; done; echo "=>
End network overlay test"
```

6. This result is an example of a positive result.

```
=> Start network overlay test
=> End network overlay test
```

7. This is a result where a UDP port was blocked.

```
=> Start network overlay test
command terminated with exit code 1
NODE2 cannot reach NODE1
command terminated with exit code 1
NODE3 cannot reach NODE1
command terminated with exit code 1
NODE1 cannot reach NODE2
command terminated with exit code 1
NODE1 cannot reach NODE3
=> End network overlay test
```

8. Some clouds, and CNI plugins have issues with MTU autodetection, make sure your MTU settings are correct as well.

References

- Container Network Interface - https://github.com/containernetworking/cni
- Network Policies - https://kubernetes.io/docs/concepts/services-networking/network-policies/
- Networking Requirements - https://rancher.com/docs/rancher/v2.x/en/cluster-provisioning/node-requirements/#networking-requirements/