Experiment: Load Balancer with Kind

This guide covers how to get service of type LoadBalancer working in a kind cluster using Metallb.

This guide complements metallb installation docs, and sets up metallb using layer2 protocol. For other protocols check metallb configuration docs.

With Docker on Linux, you can send traffic directly to the loadbalancer's external IP if the IP space is within the docker IP space.

On macOS and Windows, docker does not expose the docker network to the host. Because of this limitation, containers (including kind nodes) are only reachable from the host via port-forwards, however other containers/pods can reach other things running in docker including loadbalancers. We exposed pods and services using extra port mappings as shown in our Ingress experiment with Kind.

Create Cluster

For MacOS

Start a terminal and cd to ~/Projects/kind

For Windows

Run Git Bash from the c:\Projects\kind folder

For MacOS and Windows

Create a kind cluster with a here document for the cluster config.

```
$ kind create cluster --name loadbalancer-experiment --image
kindest/node:v1.21.12

Γ£ô Preparing nodes ≡ƒôª Creating cluster "ambassador-test" ...
 ΓÇó Ensuring node image (kindest/node:v1.21.1) ≡ƒû╝  ...
 Γ£ô Ensuring node image (kindest/node:v1.21.1) ≡ƒû╝
 ΓÇó Preparing nodes ≡ƒôª   ...
 Γ£ô Preparing nodes ≡ƒôª
 ΓÇó Writing configuration ≡ƒô£  ...
 Γ£ô Writing configuration ≡ƒô£
 ΓÇó Starting control-plane ≡ƒò╣∩╕Å  ...
 Γ£ô Starting control-plane ≡ƒò╣∩╕Å
 ΓÇó Installing CNI ≡ƒöî  ...
 Γ£ô Installing CNI ≡ƒöî
```

```
 ┌Çó Installing StorageClass ≡ƒ┘  ...
 ┌£ô Installing StorageClass ≡ƒ┘
Set kubectl context to "kind-ambassador-test"
You can now use your cluster with:

kubectl cluster-info --context kind-ambassador-test

Have a nice day! ≡ƒæï
```

$ kind get clusters

Remember that for **kubectl cluster-info** command, we need to prepend our cluster name with `kind-`

```
$ kubectl cluster-info --context kind-loadbalancer-experiment
```

Similarly unless we used the default cluster name “kind” we need to pass in **--name** for invocations to the kind cli.

$ kind get kubeconfig --name loadbalancer-experiment

```
apiVersion: v1
clusters:
. . .
```

MetalLB

Installing metallb using default manifests

Create the metallb namespace that is used to provide isolation.

$ **kubectl apply -f https://raw.githubusercontent.com/metallb/metallb/master/manifests/namespace.yaml**

```
namespace/metallb-system created
```

Create the memberlist secrets.

$ **kubectl create secret generic -n metallb-system memberlist --from-literal=secretkey="$(openssl rand -base64 128)"**

```
secret/memberlist created
```

Apply metallb manifest.

$ **kubectl apply -f https://raw.githubusercontent.com/metallb/metallb/master/manifests/metallb.yaml**

```
secret/memberlist created
```

Verify the pods are available.

$ **kubectl get pods -n metallb-system**

```
NAME                          READY   STATUS    RESTARTS   AGE
controller-6cc57c4567-rnsw5   1/1     Running   0          28s
speaker-vltbc                 1/1     Starting  0          28s
```

Wait for the pods to become available, if they are not already running state.

```
$ kubectl get pods -n metallb-system --watch
```

```
NAME                          READY   STATUS    RESTARTS   AGE
controller-6cc57c4567-rnsw5   1/1     Running   0          28s
speaker-vltbc                 1/1     Running   0          28s
```

Setup address pool used by loadbalancers

To complete layer2 configuration, we need to provide metallb a range of IP addresses it controls. We want this range to be on the docker kind network. View the network that has been created for our kind cluster.

$ **docker network inspect -f '{{.IPAM.Config}}' kind**

The output will contain a CIDR such as 172.18.0.0/16. We want our loadbalancer IP range to come from this subclass. We can configure metallb, for instance, to use 172.18.255.200 to 172.18.255.250 by creating the configmap.

Note: Keep in mind that your network may vary and be 172.19.0.0 or other, so ensure you're using the right network CIDR. Modify the config map as appropriate.

.

```
apiVersion: v1
kind: ConfigMap
metadata:
  namespace: metallb-system
  name: config
data:
  config: |
    address-pools:
    - name: default
      protocol: layer2
      addresses:
```

```
      - 172.19.255.200-172.19.255.250
```

The metallb-configmap.yaml is available in the GitHub repository.

Apply the contents of the yaml file.

$ kubectl apply -f metallb-configmap.yaml

Now we'll apply an manifest to allow us to test the communication.

$ export LB_IP=$(kubectl get svc/foo-service -o=jsonpath='{.status.loadBalancer.ingress[0].ip}')

Now verify that the loadbalancer works by sending traffic to it's external IP and port.

$ echo $LB_IP

Apply a microservice manifest

$ kubectl apply -f metallb-echo.yaml

Test with

$ curl -s localhost/great

And

$ curl -s localhost/good

Cleanup

Delete our kind cluster with a here document for the cluster config.

```
$ kind delete cluster --name loadbalancer
```