Network setting

K3s will come with pretty much everything pre-configured like `traefik`.

More info about *traefik*: https://doc.traefik.io/traefik/

However, I would like to have LoadBalancer, and in essence to be able to give services (pods) an external IP, just like my Kubernetes nodes, not from internal Kubernetes ranges. Normally, this is an external component, and your cloud provider should somehow magically give that to you, but since we are our own cloud provider, and we are trying to keep everything in one cluster... in short `MetalLB` is the answer.

What is MetalLB

https://metallb.universe.tf/

Deployment

This is a two-step process: we deploy MetalLB load balancer, and then push configuration to it, and tell it what range of IPs to use.

Apply the following: first we will create a namespace called `metallb-system`, and second we will deploy MetalLB into it.

> **Note**
>
> Look here https://metallb.universe.tf/installation/ for the most up to date version of metallb links.

```
kubectl apply -f https://raw.githubusercontent.com/metallb/metallb/v0.9.5/manifests/namespace.yaml
kubectl apply -f https://raw.githubusercontent.com/metallb/metallb/v0.9.5/manifests/metallb.yaml
```

> **Note**
>
> I prefer to store configuration files in folders named after components we deploy to cluster so that I can easily delete the service later without looking at links pointing to the Internet. So, you can just `create folder MetalLB`, and download the yaml file into it for later use.

We need to create a secret key for the `speakers` (the MetalLB pods) to encrypt speaker communications:

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

Configuration

Next, create config.yaml in your MetalLB folder; here we are going to tell MetalLB what IPs to use:

```
apiVersion: v1
kind: ConfigMap
metadata:
  namespace: metallb-system
  name: config
data:
  config: |
    address-pools:
    - name: default
      protocol: layer2
      addresses:
      - 192.168.0.230-192.168.0.250
```

As you can see, I specified a range from 192.168.0.230 to 192.168.0.250. That will give me 20 "external" IPs to work with for now.

Apply the config:

```
kubectl apply -f config.yaml
```

Check

Check if everything deployed OK

```
root@control01:~# kubectl get pods -n metallb-system
NAME                          READY   STATUS    RESTARTS   AGE
controller-65db86ddc6-7h59v   1/1     Running   0          6d5h
speaker-6vjzn                 1/1     Running   0          6d5h
speaker-b25rk                 1/1     Running   0          6d5h
speaker-dw2pv                 1/1     Running   0          6d5h
speaker-gdjzr                 1/1     Running   0          6d5h
speaker-hc72j                 1/1     Running   0          6d5h
speaker-k9nzq                 1/1     Running   0          6d5h
speaker-mfmkq                 1/1     Running   0          6d5h
speaker-qzvvz                 1/1     Running   0          6d5h
speaker-z6dk6                 1/1     Running   0          6d5h
```

You should have as many `speaker-xxxx` as you have nodes in the cluster, since they run one per node.

Now services that use LoadBalancer should have an external IP assigned to them.

For example:

```
root@control01:~# kubectl get svc --all-namespaces
NAMESPACE              NAME                        TYPE           CLUSTER-IP      EXTERNAL-IP     PORT(S)                      AGE
default                kubernetes                  ClusterIP      10.43.0.1       <none>          443/TCP                      8d
kube-system            kube-dns                    ClusterIP      10.43.0.10      <none>          53/UDP,53/TCP,9153/TCP       8d
kube-system            metrics-server              ClusterIP      10.43.246.167   <none>          443/TCP                      8d
kube-system            traefik                     LoadBalancer   10.43.61.64     192.168.0.230   80:31712/TCP,443:31124/TCP   8d
kube-system            traefik-prometheus          ClusterIP      10.43.178.172   <none>          9100/TCP                     8d
kubernetes-dashboard   dashboard-metrics-scraper   ClusterIP      10.43.147.66    <none>          8000/TCP                     8d
kubernetes-dashboard   kubernetes-dashboard        ClusterIP      10.43.151.250   <none>          443/TCP                      8d
longhorn-system        csi-attacher                ClusterIP      10.43.125.73    <none>          12345/TCP                    8d
longhorn-system        csi-provisioner             ClusterIP      10.43.118.73    <none>          12345/TCP                    8d
longhorn-system        csi-resizer                 ClusterIP      10.43.245.224   <none>          12345/TCP                    8d
longhorn-system        csi-snapshotter             ClusterIP      10.43.230.3     <none>          12345/TCP                    8d
longhorn-system        longhorn-backend            ClusterIP      10.43.118.82    <none>          9500/TCP                     8d
longhorn-system        longhorn-frontend           ClusterIP      10.43.204.227   <none>          80/TCP                       8d
openfaas               alertmanager                ClusterIP      10.43.79.30     <none>          9093/TCP                     6d6h
openfaas               basic-auth-plugin           ClusterIP      10.43.163.133   <none>          8080/TCP                     6d6h
openfaas               gateway                     ClusterIP      10.43.187.155   <none>          8080/TCP                     6d6h
openfaas               gateway-external            NodePort       10.43.71.53     <none>          8080:31112/TCP               6d6h
openfaas               nats                        ClusterIP      10.43.123.99    <none>          4222/TCP                     6d6h
openfaas               prometheus                  ClusterIP      10.43.78.247    <none>          9090/TCP                     6d6h
```

Look how `traefik` automatically got an IP from the external range. In the end, this is what you would want: Not to point to a single node IP and be redirected based on DNS, which would stop working the moment the node with that IP died. This way, we make the external IP node independent. Now, you can point DNS to that IP and be sure it will be routed correctly.

> **Note**
>
> This is how *I* prefer my network settings, and makes most sense to me when creating external services. I'm sure there are like a hundred different methods using external load balancers, Nginx ingress (basically reverse proxy) and who knows what in production, but hey, there is no official one standardized setting for Kubernetes (which can be such a pain sometimes) so who's to say this is not OK? 🙂

Liked it ? Buy me a drink :)

Last update: October 20, 2021

Comments

What do you think?

6 Responses

Upvote Funny Love Surprised Angry Sad

7 Comments https://rpi4cluster.com Disqus' Privacy Policy Login

♡ Favorite Tweet Share Sort by Best

Join the discussion…

LOG IN WITH

OR SIGN UP WITH DISQUS ?

Name

Alex Ellis • 6 months ago

Have you tried the inlets-operator yet? It works really well with MetalLB and Traefik to give you public LoadBalancer IPs alongside any local ones you want to keep.

No messy NAT or port-forwarding rules required. https://blog.zespre.com/inl...

1 ^ | ˅ • Reply • Share ›

vladoportos Mod ➦ Alex Ellis • 6 months ago

Not yet, I knew about it from you, I think you have introduced the concept to me.. Either on blog or in your e-book (can't remember now). They are definitely a good way to expose service, but I did not want that at the time. I'm in the process of rebuilding my cluster from scratch right now. Including design of new enclosure for RPIs and Power system.. So as soon as the enclosure is done, I will be reinstalling everything again and improving this guide as well to be more up to date. I will add inlets as well.

1 ^ | ˅ • Reply • Share ›

Freman • 2 months ago

So, I've done this 6 or 7 times now and it worked out perfectly, but this last time I moved my kubes cluster onto it's own vlan at last and started rebuilding, only I forgot to change my config from 10.0.0.10-10.0.0.30 to 10.0.1.200-10.0.1.250... now my traefik is on 10.0.0.30 and I can't figure out how to make it move (even after I changed configs)

^ | ˅ • Reply • Share ›

Karthik • a year ago

I had a nightmare with metallb and k3s. It seems k3s has inbuilt service that assigns IP to loadbalancer type services.

Check this

https://github.com/metallb/...

Might be useful to someone.

btw, your guide is really detailed and very much helpful for those who start with k3s like me. thanks.

^ | ˅ • Reply • Share ›

vladoportos Mod ➦ Karthik • a year ago

Yes the issue is known, I did use --disable servicelb when installing K3s :)

^ | ˅ • Reply • Share ›

Devin Flake ➦ vladoportos • 6 months ago

Do you know if this is still an issue? I've been playing around with k3s and MetalLB but I'm not experienced enough to find any problems.

^ | ˅ • Reply • Share ›