Ingress Controller

13 January 2021 10:44

Bare Metal OR On-Prem Kubernetes Cluster OR Unmanaged Cloud K8s

https://kubernetes.github.io/ingress-nginx/deploy/baremetal/ - Why MetalLB -- or a LB

- Create a Kubernetes Deployment.
- Deploy NGINX Ingress Controller with Helm.
- Set up an Ingress Resource object for the Deployment.

Objectives

- Deploy a simple Kubernetes web *application* Deployment.
- Deploy *NGINX Ingress Controller* using the stable Helm chart.
- Deploy an *Ingress Resource* for the *application* that uses *NGINX Ingress* as the controller.
- Test NGINX Ingress functionality by accessing the Google Cloud L4 (TCP/UDP) load balancer frontend IP address and ensure that it can access the web application.

Pre-Requisites

- Kubernetes cluster
  - Minimum 1 Master and 1 worker
  - Namespace -
- Weave Network (CNI)
- Helm v3.x

Building Blocks

- Metal LB // Hardware LB
  - https://metallb.universe.tf/installation/
- NGNIX Controller
- Helm

Implementation Steps:

- Master with one worker
  - Weave CNI

```
[root@master ~]# kubectl  get nodes -o wide
NAME     STATUS   ROLES    AGE     VERSION   INTERNAL-IP    EXTERNAL-IP   OS-IMAGE                KERNEL-VERSION                CONTAINER-RUNTIM
master   Ready    master   2d13h   v1.19.3   192.168.1.74   <none>        CentOS Linux 7 (Core)   3.10.0-1160.11.1.el7.x86_64   docker://19.3.11
worker   Ready    <none>   2d13h   v1.19.3   192.168.1.75   <none>        CentOS Linux 7 (Core)   3.10.0-1160.11.1.el7.x86_64   docker://19.3.11
[root@master ~]#
[root@master ~]#
[root@master ~]#
[root@master ~]# kubectl  get pods -A
NAMESPACE     NAME                                READY   STATUS      RESTARTS   AGE
kube-system   coredns-f9fd979d6-6kmt2             0/1     Completed   1          2d13h
kube-system   coredns-f9fd979d6-fq24p             1/1     Running     2          2d13h
kube-system   etcd-master                         1/1     Running     3          2d13h
kube-system   kube-apiserver-master               1/1     Running     3          2d13h
kube-system   kube-controller-manager-master      1/1     Running     3          2d13h
kube-system   kube-proxy-dlz5q                    1/1     Running     2          2d13h
kube-system   kube-proxy-qrn7s                    1/1     Running     2          2d13h
kube-system   kube-scheduler-master               1/1     Running     3          2d13h
kube-system   weave-net-7kwx7                     2/2     Running     6          2d13h
kube-system   weave-net-zpf6s                     2/2     Running     7          2d13h
[root@master ~]#
```

1) Install Metal LB

kubectl edit configmap -n kube-system kube-proxy

- Change ARP from false to true

To install MetalLB, apply the manifest:

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

```
[root@master ~]# kubectl apply -f https://raw.githubusercontent.com/metallb/metallb/v0.9.5/manifests/namespace.yaml
namespace/metallb-system created
[root@master ~]# kubectl apply -f https://raw.githubusercontent.com/metallb/metallb/v0.9.5/manifests/metallb.yaml
podsecuritypolicy.policy/controller created
podsecuritypolicy.policy/speaker created
serviceaccount/controller created
serviceaccount/speaker created
clusterrole.rbac.authorization.k8s.io/metallb-system:controller created
clusterrole.rbac.authorization.k8s.io/metallb-system:speaker created
role.rbac.authorization.k8s.io/config-watcher created
role.rbac.authorization.k8s.io/pod-lister created
clusterrolebinding.rbac.authorization.k8s.io/metallb-system:controller created
clusterrolebinding.rbac.authorization.k8s.io/metallb-system:speaker created
rolebinding.rbac.authorization.k8s.io/config-watcher created
rolebinding.rbac.authorization.k8s.io/pod-lister created
daemonset.apps/speaker created
deployment.apps/controller created
[root@master ~]#
```

```
[root@master ~]# cat metal-config.yaml
apiVersion: v1
kind: ConfigMap
metadata:
  namespace: metallb-system
  name: config
data:
  config: |
    address-pools:
    - name: default
      protocol: layer2
      addresses:
      - 192.168.1.115-192.168.1.120
[root@master ~]#
```

```
[root@master ~]# kubectl create secret generic -n metallb-system memberlist --from-literal=secretkey="$(openssl rand -base64 128)"
secret/memberlist created
```

```
[root@master ~]# kubectl get pods -n metallb-system
NAME                          READY   STATUS    RESTARTS   AGE
controller-65db86ddc6-xqp8h   1/1     Running   0          108s
speaker-5v8pc                 1/1     Running   0          109s
speaker-8ww8z                 1/1     Running   0          109s
[root@master ~]#
```

```
[root@master ~]# kubectl create -f metallb-config.yaml
configmap/config created
[root@master ~]#
```

Add IP range in MetalLB for Services:

```
[root@master ~]# cat metal-config.yaml
apiVersion: v1
kind: ConfigMap
metadata:
  namespace: metallb-system
  name: config
data:
  config: |
    address-pools:
    - name: default
      protocol: layer2
      addresses:
      - 192.168.1.115-192.168.1.120
[root@master ~]#
```

2) Install NGINX Ingress Controller

yum install wget -y
wget https://get.helm.sh/helm-v3.5.0-rc.2-linux-amd64.tar.gz
tar -zxvf helm-v3.5.0-rc.2-linux-amd64.tar.gz
mv linux-amd64/helm /usr/local/bin/
helm version

==========

```
[root@master ~]# helm repo add stable https://charts.helm.sh/stable
"stable" has been added to your repositories
[root@master ~]# helm repo update
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "stable" chart repository
Update Complete. □Happy Helming!□
[root@master ~]#
```

```
[root@master ~]# helm repo add ingress-nginx https://kubernetes.github.io/ingress-nginx
"ingress-nginx" has been added to your repositories
[root@master ~]# helm repo update
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "ingress-nginx" chart repository
...Successfully got an update from the "stable" chart repository
Update Complete. □Happy Helming!□
[root@master ~]#
```

```
[root@master ~]# helm repo list
NAME            URL
stable          https://charts.helm.sh/stable
ingress-nginx   https://kubernetes.github.io/ingress-nginx
[root@master ~]#
```

```
[root@master ~]# helm show values ingress-nginx/ingress-nginx > ingress-values.yaml
[root@master ~]# vi ingress-values.yaml
```

```
[root@master ~]# cat ingress-values.yaml |egrep -i 'hostPort|Deployment' -A3
  # since CNI and hostport don't mix yet. Can be deprecated once https://github.com/kubernetes/kubernetes/issues/23920
  # is merged
  #hostNetwork: false
  hostNetwork: true
--
  hostPort:
    enabled: true
    #enabled: false
    ports:
--
  ## DaemonSet or Deployment
  ##
  kind: DaemonSet
#  kind: Deployment
```

```
[root@master ~]# kubectl create ns nginx-ingress
namespace/nginx-ingress created
[root@master ~]#
```

```
[root@master ~]# helm install ng-ingress ingress-nginx/ingress-nginx -n nginx-ingress --values ingress-values.yaml
NAME: ng-ingress
LAST DEPLOYED: Fri Jan  8 04:08:19 2021
NAMESPACE: nginx-ingress
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
The ingress-nginx controller has been installed.
It may take a few minutes for the LoadBalancer IP to be available.
You can watch the status by running 'kubectl --namespace nginx-ingress get services -o wide -w ng-ingress-ingress-nginx-controller'
```

```
[root@master ~]# kubectl  get all -n nginx-ingress
NAME                                            READY   STATUS    RESTARTS   AGE
pod/ng-ingress-ingress-nginx-controller-kz4vz   1/1     Running   0          52s

NAME                                                 TYPE           CLUSTER-IP       EXTERNAL-IP     PORT(S)                      AGE
service/ng-ingress-ingress-nginx-controller          LoadBalancer   10.110.17.159    192.168.1.115   80:30269/TCP,443:32680/TCP   52s
service/ng-ingress-ingress-nginx-controller-admission   ClusterIP      10.110.248.107   <none>          443/TCP                      52s

NAME                                                 DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR            AGE
daemonset.apps/ng-ingress-ingress-nginx-controller   1         1         1       1            1           kubernetes.io/os=linux   52s
[root@master ~]#
```

3) Do a deployment and expose

```
[root@master ~]# kubectl create deployment myweb --image=nginx -n nginx-ingress
deployment.apps/myweb created
[root@master ~]#
```

```
[root@master ~]# kubectl expose deployment -n nginx-ingress myweb --port 80
service/myweb exposed
[root@master ~]#
[root@master ~]#
[root@master ~]#
[root@master ~]# kubectl  describe svc -n nginx-ingress myweb
Name:              myweb
Namespace:         nginx-ingress
Labels:            app=myweb
Annotations:       <none>
Selector:          app=myweb
Type:              ClusterIP
IP Families:       <none>
IP:                10.108.171.218
IPs:               10.108.171.218
Port:              <unset>  80/TCP
TargetPort:        80/TCP
Endpoints:         10.32.0.2:80
Session Affinity:  None
Events:            <none>
[root@master ~]#
```

4) Define ingress resource

```
[root@master ~]# kubectl get all -n nginx-ingress
NAME                                             READY   STATUS    RESTARTS   AGE
pod/myweb-855c667ff6-v6c2s                       1/1     Running   0          5d21h
pod/ng-ingress-ingress-nginx-controller-kz4vz    1/1     Running   0          5d21h

NAME                                                TYPE           CLUSTER-IP       EXTERNAL-IP     PORT(S)                      AGE
service/myweb                                       ClusterIP      10.108.171.218   <none>          80/TCP                       5d21h
service/ng-ingress-ingress-nginx-controller         LoadBalancer   10.110.17.159    192.168.1.115   80:30269/TCP,443:32680/TCP   5d21h
service/ng-ingress-ingress-nginx-controller-admission   ClusterIP      10.110.248.107   <none>          443/TCP                      5d21h

NAME                                             DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR            AGE
daemonset.apps/ng-ingress-ingress-nginx-controller   1         1         1       1            1           kubernetes.io/os=linux   5d21h

NAME                    READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/myweb   1/1     1            1           5d21h

NAME                               DESIRED   CURRENT   READY   AGE
replicaset.apps/myweb-855c667ff6   1         1         1       5d21h
```

```
[root@master ~]# kubectl create -f  myweb.yaml
ingress.networking.k8s.io/myweb-inress created
[root@master ~]# cat myweb.yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: myweb-inress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
  namespace: nginx-ingress
spec:
  rules:
  - host: myweb.example.com
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: myweb
            port:
              number: 80
[root@master ~]#
```

```
[root@master ~]# cat /etc/hosts
192.168.1.74 master
192.168.1.75 worker
192.168.1.115 myweb.example.com
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
[root@master ~]#
```

```
[root@master ~]# curl http://myweb.example.com
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>
```

```
[root@master ~]# kubectl  create -f namebasedvs.yaml
ingress.networking.k8s.io/ingress-resource-2 created
[root@master ~]#
```

Name based Routing

```
[root@master ~]# kubectl get all
NAME                                       READY   STATUS    RESTARTS   AGE
pod/nginx-deploy-blue-9784c656c-brwfh      1/1     Running   0          2m5s
pod/nginx-deploy-green-786b88cb6-5vfw6     1/1     Running   0          2m5s
pod/testweb-56744c9b58-d4nk2               1/1     Running   0          14m

NAME                 TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)   AGE
service/kubernetes   ClusterIP   10.96.0.1        <none>        443/TCP   2d15h
service/my-web       ClusterIP   10.102.123.172   <none>        80/TCP    13m

NAME                                  READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/nginx-deploy-blue     1/1     1            1           2m5s
deployment.apps/nginx-deploy-green    1/1     1            1           2m5s
deployment.apps/testweb               1/1     1            1           14m

NAME                                             DESIRED   CURRENT   READY   AGE
replicaset.apps/nginx-deploy-blue-9784c656c      1         1         1       2m5s
replicaset.apps/nginx-deploy-green-786b88cb6     1         1         1       2m5s
replicaset.apps/testweb-56744c9b58               1         1         1       14m
[root@master ~]#
```

```
[root@master ~]# kubectl  expose deployment nginx-deploy-blue --port 80
service/nginx-deploy-blue exposed
[root@master ~]# kubectl  expose deployment nginx-deploy-green --port 80
service/nginx-deploy-green exposed
[root@master ~]#
```

```
[root@master ~]# curl http://green.nginx.example.com
<h1>I am <font color=green>GREEN</font></h1>
[root@master ~]#
[root@master ~]#
[root@master ~]# curl http://blue.nginx.example.com
<h1>I am <font color=blue>BLUE</font></h1>
[root@master ~]#
[root@master ~]#
[root@master ~]#
[root@master ~]#
[root@master ~]#
[root@master ~]# cat /etc/hosts
192.168.1.74 master
192.168.1.75 worker
192.168.1.115 myweb.example.com nginx.example.com blue.nginx.example.com green.nginx.example.com
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
```

Path based routing

```
[root@master ~]# kubectl  create -f pathbased.yaml
ingress.networking.k8s.io/ingress-resource-3 created
[root@master ~]#
```

```
[root@master ~]# curl http://nginx2.example.com/green
<h1>I am <font color=green>GREEN</font></h1>
[root@master ~]# curl http://nginx2.example.com/blue
<h1>I am <font color=blue>BLUE</font></h1>
[root@master ~]# kubectl get ingress
Warning: extensions/v1beta1 Ingress is deprecated in v1.14+, unavailable in v1.22+; use networking.k8s.io/v1 Ingress
NAME                 CLASS    HOSTS                                                  ADDRESS         PORTS   AGE
ingress-resource-2   <none>   nginx.example.com,blue.nginx.example.com,green.nginx.example.com   192.168.1.115   80      10m
ingress-resource-3   <none>   nginx2.example.com                                     192.168.1.115   80      79s
myweb-ingress        <none>   myweb.example.com                                      192.168.1.115   80      21m
[root@master ~]#
```

ON Cloud

- Create a Kubernetes Deployment.
- Deploy NGINX Ingress Controller with Helm.
- Set up an Ingress Resource object for the Deployment.

Objectives

- Deploy a simple Kubernetes web *application* Deployment.
- Deploy *NGINX Ingress Controller* using the stable Helm chart.
- Deploy an *Ingress Resource* for the *application* that uses *NGINX Ingress* as the controller.
- Test NGINX Ingress functionality by accessing the Google Cloud L4 (TCP/UDP) load balancer frontend IP address and ensure that it can access the web application.

Pre-Requisites

- Account on GCP
- A Project on GCP
- Enable Billing on the project
- GKE API
- A GKE Cluster
- Helm v3.x [available in cloud shell/VM]

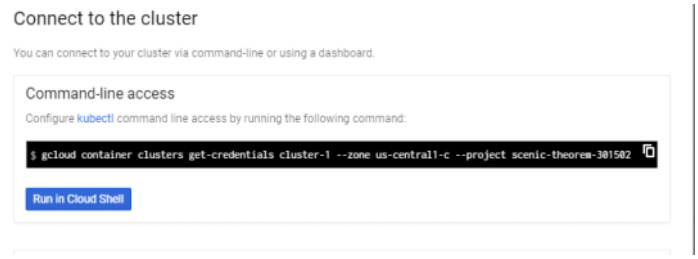
Building Blocks

- NGNIX LB
- With LoadBalancer as a service in GKE
  - TCP LB with NGINX as backend
- Firewall Rules (GCP)

Implementation Steps:

Deploy Application

5) Connect to your GKE cluster




```
opensourcetechk8s@cloudshell:~ (scenic-theorem-301502)$ gcloud container clusters get-credentials cluster-1 --zone us-central1-c --project scenic-theorem-301502
Fetching cluster endpoint and auth data.
kubeconfig entry generated for cluster-1.
opensourcetechk8s@cloudshell:~ (scenic-theorem-301502)$
```

```
opensourcetechk8s@cloudshell:~/nginx-ingress (scenic-theorem-301502)$ kubectl get nodes -o wide
NAME                                          STATUS   ROLES    AGE   VERSION           INTERNAL-IP   EXTERNAL-IP      OS-IMAGE                             KERNEL-VERSION   CONTAINER-RUNTIME
gke-cluster-1-default-pool-3dedd729-bpvf   Ready    <none>   31m   v1.16.15-gke.4901   10.128.0.6    35.188.109.41    Container-Optimized OS from Google   4.19.112+        docker://19.3.1
gke-cluster-1-default-pool-3dedd729-pf5q   Ready    <none>   31m   v1.16.15-gke.4901   10.128.0.8    34.121.115.153   Container-Optimized OS from Google   4.19.112+        docker://19.3.1
gke-cluster-1-default-pool-3dedd729-tq1m   Ready    <none>   31m   v1.16.15-gke.4901   10.128.0.7    34.71.118.54     Container-Optimized OS from Google   4.19.112+        docker://19.3.1
opensourcetechk8s@cloudshell:~/nginx-ingress (scenic-theorem-301502)$
```

6) Deploy a application form GCR

kubectl create deployment hello-app --image=gcr.io/google-samples/hello-app:1.0

```
opensourcetechk8s@cloudshell:~ (scenic-theorem-301502)$ kubectl create deployment hello-app --image=gcr.io/google-samples/hello-app:1.0
deployment.apps/hello-app created
opensourcetechk8s@cloudshell:~ (scenic-theorem-301502)$
```

7) Expose your deployment
kubectl expose deployment hello-app --port=8080 --target-port=8080

```
opensourcetechk8s@cloudshell:~ (scenic-theorem-301502)$ kubectl get svc
NAME         TYPE        CLUSTER-IP    EXTERNAL-IP   PORT(S)    AGE
hello-app    ClusterIP   10.8.10.21    <none>        8080/TCP   32s
kubernetes   ClusterIP   10.8.0.1      <none>        443/TCP    16m
opensourcetechk8s@cloudshell:~ (scenic-theorem-301502)$
```

Deploy NGINX Ingress

**** Can use GKE Ingress (managed Ingress)**

8) Add NGINX repo into Helm

$ helm repo add nginx-stable https://helm.nginx.com/stable
$ helm repo update

```
opensourcetechk8s@cloudshell:~ (scenic-theorem-301502)$ helm repo update
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "nginx-stable" chart repository
...Successfully got an update from the "prometheus-community" chart repository
Update Complete. ⎈ Happy Helming!⎈
opensourcetechk8s@cloudshell:~ (scenic-theorem-301502)$
```

9) Install NGINX using helm repo

helm install nginx-ingress nginx-stable/nginx-ingress

```
opensourcetechk8s@cloudshell:~/nginx-ingress (scenic-theorem-301502)$ helm install nginx-ingress nginx-stable/nginx-ingress
NAME: nginx-ingress
LAST DEPLOYED: Wed Jan 13 05:47:30 2021
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
The NGINX Ingress Controller has been installed.
opensourcetechk8s@cloudshell:~/nginx-ingress (scenic-theorem-301502)$
```

10) Verify deployment and notice your LB IP

```
opensourcetechk8s@cloudshell:~/nginx-ingress (scenic-theorem-301502)$ kubectl get deployment nginx-ingress-nginx-ingress
NAME                          READY   UP-TO-DATE   AVAILABLE   AGE
nginx-ingress-nginx-ingress   1/1     1            1           64s
opensourcetechk8s@cloudshell:~/nginx-ingress (scenic-theorem-301502)$ kubectl get service nginx-ingress-nginx-ingress
NAME                          TYPE           CLUSTER-IP    EXTERNAL-IP      PORT(S)                      AGE
nginx-ingress-nginx-ingress   LoadBalancer   10.8.12.33    104.197.235.144   80:30156/TCP,443:32688/TCP   66s
opensourcetechk8s@cloudshell:~/nginx-ingress (scenic-theorem-301502)$
```

11) Export your LB External IP

export myapppp=$(kubectl get service nginx-ingress-nginx-ingress -ojson | jq -r '.status.loadBalancer.ingress[].ip')

```
opensourcetechk8s@cloudshell:~/nginx-ingress (scenic-theorem-301502)$ export myapppp=$(kubectl get service nginx-ingress-nginx-ingress -ojson | jq -r '.status.loadBalancer.ingress[].ip')
opensourcetechk8s@cloudshell:~/nginx-ingress (scenic-theorem-301502)$ echo $NGINX_INGRESS_IP
104.197.235.144
opensourcetechk8s@cloudshell:~/nginx-ingress (scenic-theorem-301502)$
```

Configure NGINX Ingress Controller

**** Annotation -**

annotations: kubernetes.io/ingress.class: nginx

12) Simple Ingress controller:
   a. Use YAML file to deploy
   b. Domain name **xip.io**

   # kubectl apply -f simple-Ingress.yaml

```
opensourcetechk8s@cloudshell:~/nginx-ingress (scenic-theorem-301502)$ kubectl apply -f simple-ingtress.yaml
ingress.networking.k8s.io/ingress-resource created
opensourcetechk8s@cloudshell:~/nginx-ingress (scenic-theorem-301502)$ kubectl get ingress
NAME               HOSTS            ADDRESS           PORTS   AGE
ingress-resource   myapppp.xip.io   104.197.235.144   80      69s
opensourcetechk8s@cloudshell:~/nginx-ingress (scenic-theorem-301502)$
```