Star Fork Watch Follow @collabnix

Deny Ingress Traffic That Has No Rules

Kubernetes - Beginners | Intermediate | Advanced

View on GitHub Join Slack

Deny Ingress Traffic That Has No Rules

An effective network security rule starts with denying all traffic by default unless explicitly allowed. This is how firewalls work. By default, Kubernetes regards any pod that is not selected by a NetworkPolicy as “non-isolated”. This means all ingress and egress traffic is allowed. So, a good foundation is to deny all traffic by default unless a NetworkPolicy rule defines which connections should pass. A NetworkPolicy definition for denying all ingress traffic may look like this:

Steps

```
git clone https://github.com/collabnix/kubelabs.git
cd kubelabs/Network_Policies101/
```

- We’ll use a new namespace for this guide. Run the following commands to create it and a plain nginx service listening on port 80.

  ```
  kubectl create ns network-policy-demo
  ```

```
kubectl run --namespace=network-policy-demo nginx --replicas=2 --image=
kubectl expose --namespace=network-policy-demo deployment nginx --port=
```

- Create the busy box pod to test policy access.

```
[node1 ~]$ kubectl run --generator=run-pod/v1  --namespace=network-poli
If you don't see a command prompt, try pressing enter.
/ #
```

- Now from within the busybox “access” pod execute the following command to test access to the nginx service.

```
wget -q --timeout=5 nginx -O -
```

It should return the HTML of the nginx welcome page.

- Still within the busybox “access” pod, issue the following command to test access to google.com.

```
wget -q --timeout=5 google.com -O -
```

It should return the HTML of the google.com home page.

Enable ingress isolation on the namespace by deploying a default-deny-ingress policy

```
kubectl apply -f default-deny-ingress.yaml
```

Verify Access - Denied All Ingress

Because all pods in the namespace are now selected, any ingress traffic which is not explicitly allowed by a policy will be denied.

We can see that this is the case by switching over to our “access” pod in the namespace and attempting to access the nginx service.

```
kubectl run --namespace=advanced-policy-demo access --rm -ti --image busybox

/ # wget -q --timeout=5 nginx -O -
wget: download timed out
/ #
```

Next, try to access google.com.

```
wget -q --timeout=5 google.com -O -
<!doctype html><html itemscope="" itemt
```

We can see that the ingress access to the nginx service is denied while egress access to outbound internet is still allowed.

- Cleanup step

You can clean up after this tutorial by deleting the network-policy-demo namespace.

```
kubectl delete ns network-policy-demo
```

Join KubeDaily

8 Members Online

Support

MEMBERS ONLINE

h3ll_boy
MEE6
Nitinkashyap Apex Legends
ojaswa
Parmeshwar
prasad
Shady
vikas027

Free voice chat from Discord

Connect

Tweets by collabnix

kubelabs is maintained by collabnix.

This page was generated by GitHub Pages.