KUBERNETES (ENGLISH)

3. forduló

A kategória támogatója: Nokia

Ismertető a feladatlaphoz

Kérjük, hogy a feladatlap indítása előtt mindenképp olvasd el az alábbi útmutatót:

> Amennyiben olyan kategóriában játszol, ahol van csatolmány, de hibába ütközöl a letöltésnél, ott valószínűleg a vírusirtó korlátoz, annak ideiglenes kikapcsolása megoldhatja a problémát. (Körülbelül minden 3000. letöltésnél fordul ez elő.)

> Helyezéseket a 4. forduló után mutatunk, százalékos formában: adott kategóriában a TOP 20-40-60%-hoz tartozol.
>
> A feltűnően rövid idő alatt megoldott feladatlapok kizárást vonnak maguk után, bármilyen más gyanús esetben fenntartjuk a jogot a forduló érvénytelenítésére!

Kubernetes - Networking

In Kubernetes, "networking" refers to the mechanisms that enable communication between various components within the cluster. Each pod within the cluster is assigned a unique IP address, facilitating direct communication between pods. Services act as an abstraction layer, providing a consistent IP and load balancing for pods. Network Policies offer fine-grained control over traffic flow, allowing administrators to enforce communication rules between pods based on their labels.

Before you start, please read the following hint:

KDiff3 is a free and open-source diff and merge tool that can be helpful during the solution of coding-related exercises.
Download link: https://sourceforge.net/projects/kdiff3/files/

Good luck!

1. feladat 1 pont

You have a Kubernetes cluster with Calico as the network plugin. Two deployments, **web-app** and **database**, reside in the same namespace called **production**. After a recent set of changes, traffic from **web-app** to **database** has been inadvertently blocked.

Given the following output from '**calicoctl get networkpolicy -n production**':

```
NAME                 ORDER   SELECTOR
web-to-db-egress     1000    app == 'web-app'
db-ingress           1500    app == 'database'
default-allow        2000    app in {"web-app", "database"}
```

The policies are defined as:

web-to-db-egress

```
apiVersion: projectcalico.org/v3
kind: NetworkPolicy
metadata:
  name: web-to-db-egress
  namespace: production
spec:
  selector: app == 'web-app'
  types:
  - Egress
  egress:
  - action: Allow
    destination:
      selector: app == 'database'
  - action: Deny
    destination:
      notSelector: app == 'database'
```

db-ingress

```
apiVersion: projectcalico.org/v3
kind: NetworkPolicy
metadata:
  name: db-ingress
  namespace: production
spec:
  selector: app == 'database'
  types:
  - Ingress
  ingress:
  - action: Deny
    source:
      selector: app == 'web-app'
```

default-allow

```
apiVersion: projectcalico.org/v3
kind: NetworkPolicy
metadata:
  name: default-allow
  namespace: production
spec:
  selector: app in {"web-app", "database"}
  types:
  - Ingress
  - Egress
  egress:
  - action: Allow
  ingress:
  - action: Allow
```

Which modification(s) would you make to restore traffic from **web-app** to **database**?

Válasz

- Modify the web-to-db-egress policy to remove the Deny action for destinations not labeled as database
- Modify the db-ingress policy to change the Deny action for sources labeled as web-app to Allow
- Delete the db-ingress policy
- Modify the default-allow policy to increase its order to be less than 1500
- Modify the db-ingress policy's selector to app == 'web-app'

2. feladat 3 pont

You are tasked with setting up a node in your Calico-enabled Kubernetes cluster to act as a BGP Route Reflector. To ensure seamless network connectivity and correct route reflection, which of the following steps and configurations are required?

Válaszok

- [ ] Deploy a special "route-reflector" version of the calico-node pod on the designated node
- [ ] Ensure that the Calico node resource for the route reflector has the routeReflectorClusterID specified
- [ ] Update the Calico global BGP configuration with routeReflectorClusterID
- [ ] Adjust the nodeToNodeMeshEnabled setting in the BGP configuration to false
- [ ] Establish explicit BGPPeer resources to point non-route-reflector nodes to the route reflector node

3. feladat 2 pont

You're troubleshooting a networking issue in a Kubernetes cluster where a specific pod is unable to communicate with services outside its node. You suspect an issue with the virtual Ethernet (veth) interface of the pod. Which of the following steps can help identify and troubleshoot the issue?

Válaszok

- [ ] Use ip a or ip link inside the pod to inspect the status of the eth0 interface
- [ ] Check the veth pair's connectivity using the ping command from the node's root network namespace to the pod's eth0 IP address
- [ ] Examine the node's root network namespace using tools like "ip a" to identify the veth pair associated with the problematic pod
- [ ] Inspect CNI (Container Network Interface) logs, as CNI plugins are responsible for creating veth pairs during pod creation
- [ ] Delete and recreate the pod to reset its veth pair, as veth pairs cannot be modified once created
- [ ] Use netstat inside the pod to examine the routing table, ensuring that the default route is correctly set

Megoldások beküldése