Kubernetes

Ben Gavan

August 26, 2020

Abstract

1 Definitions

- Node = A worker machine (physical and/or virtual) that Kubernetes can run applications on.
- Pod = Collection of Containers

  The smallest deployable unit of computing that can be created and managed in Kubernetes.

  basic unit of deployment.

  All containers in a pod are scheduled in the same node.
- Ingress = An API object that manages external access to the services in a cluster (typically HTTP). May provide:

  load balancing,

  SSL Termination,

  Name-based virtual hosting.
- Cluster = A Set of Nodes

  In most deployments, nodes in a cluster are NOT part of the public internet.
- Edge Router = A Router that enforces the firewall policy for the cluster.

  Could be physical piece of hardware

  or a gateway managed by a cloud provider.
- Cluster Network = A Set of links (logical or physical) that facilitate communication within a cluster (according to the networking model).
- Service = Identifies a set of pods (a cluster) using label selectors

  assumed to have virtual IPs only routable within the cluster.

2 Theory

2.1 Ingress

Ingress exposes HTTP and HTTPS routes from outside the cluster to services within the cluster.

```
  internet
     |
[ Ingress ]
--|-----|--
[ Services ]
```

3 Defining

Defined using a *deployment.yaml* file.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: kubernetes-tutorial-deployment
spec:
  replicas: 2
  selector:
    matchLabels:
      app: kubernetes-tutorial-deployment
  template:
    metadata:
      labels:
        app: kubernetes-tutorial-deployment
    spec:
      containers:
      - name: kubernetes-tutorial-application
      image: auth0blog/kubernetes-tutorial
      ports:
        - containerPort: 3000
```

All Kubernetes resources needs fields for:

- apiVersion
- kind
- metadata

4 Networking

4.1 Introduction to Ingress

Ingress exposed HTTP and HTTPS routes from outside to services within the cluster. Traffic routing is controlled by rules defined on the Ingress resource and can be configured to

- give services externally reachable URLs
- load balance traffic
- terminate SSL/TLS
- & offer name based virtual hosting.

An Ingress Controller is responsible for fulfilling the Ingress (usually with a load balancer).
An Ingress does not expose arbitrary ports or protocols.
Must have an Ingress Controller to satisfy an Ingress such as *ingress-nginx*. (only creating an Ingress Resource have not effect)

```
 internet
     |
[ Ingress ]
--|-----|--
[ Services ]
```

4.2 The Ingress Resource

A minimal Ingress resource example:

```
apiVersion: networking.k8s.io/v1beta1
kind: Ingress
metadata:
  name: test-ingress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  rules:
  - http:
    paths:
      - path: /testpath
      pathType: Prefix
      backend:
        serviceName: test
        servicePort: 80
```

The name of an Ingress object must be a valid DNS Subdomain name

```
www.subdomain.domain.com
```

Ingress resource only supports rules for directing HTTP traffic.

4.3 Internal Networking

4.4 External Networking

5 Launching

6 Modifying

7 Deleting

8 Basic Commands

8.1 Get all pods

```
kubectr got pods
```

8.2 exec into centOS

```
kubectl exec <name (specified in pod.yaml)> -c shell -i -t -- bash
```

we can then interface with it via

```
curl -s localhost:9876/info
```

9 References

- https://kubernetes.io/docs/concepts/services-networking/ingress/