Praktik Penggunaan Kubernetes di Arch Linux

Disusun oleh: Muhammad Amir Al Aqwa

1. Pendahuluan

Kubernetes, yang sering disingkat K8s, adalah platform orkestrasi kontainer *open-source* yang dirancang untuk mengotomatisasi penyebaran, penskalaan, dan pengelolaan aplikasi dalam lingkungan kontainer. Dengan semakin banyaknya aplikasi yang menggunakan arsitektur *microservices*, kebutuhan akan sistem yang dapat mengelola ribuan kontainer secara efisien menjadi sangat penting. Kubernetes mengatasi tantangan ini dengan menyediakan kerangka kerja yang kuat dan fleksibel, memungkinkan pengembang dan operator fokus pada pengembangan fitur daripada mengurus infrastruktur. Dokumen ini akan menjelaskan secara rinci tentang komponen utama Kubernetes, langkah-langkah instalasi, dan perintah-perintah dasar yang esensial untuk mengelola sebuah *cluster*.

2. Komponen Utama Kubernetes

Arsitektur Kubernetes terdiri dari dua bagian utama: ***Master Node*** (atau *Control Plane*) dan ***Worker Node***. Masing-masing node memiliki komponen-komponen penting yang bekerja sama untuk menjaga *cluster* tetap berjalan.

- ***Master Node* (Control Plane):** Ini adalah "otak" dari *cluster* yang bertanggung jawab untuk mengelola semua *worker node* dan Pod. Komponen utamanya meliputi:
    - **kube-apiserver:** Bertindak sebagai antarmuka utama untuk semua komunikasi dalam *cluster*. Semua permintaan, baik internal maupun eksternal, harus melewati komponen ini.
    - **etcd:** Sebuah *database key-value* terdistribusi yang menyimpan semua data konfigurasi dan status *cluster*.
    - **kube-scheduler:** Bertugas memilih *node* yang paling sesuai untuk menjalankan Pod baru.
    - **kube-controller-manager:** Menjalankan berbagai "pengontrol" yang memastikan status *cluster* saat ini sesuai dengan yang diinginkan.
- ***Worker Node*:** Ini adalah mesin yang menjalankan aplikasi kontainer. Komponen utamanya adalah:
    - **kubelet:** Agen yang berjalan di setiap *worker node* dan memastikan kontainer dalam Pod berjalan dan sehat.
    - **kube-proxy:** Bertanggung jawab untuk mengelola aturan jaringan dan *load balancing* untuk *Service*.
    - **Container Runtime:** Perangkat lunak yang menjalankan kontainer, seperti Docker

atau Containerd.

3. Instalasi Kubernetes

Untuk memulai penggunaan Kubernetes, biasanya kita membutuhkan alat bantu seperti **kubectl** dan **Minikube**. **kubectl** adalah *command-line tool* yang digunakan untuk berkomunikasi dengan *cluster* Kubernetes, sedangkan **Minikube** adalah alat yang memungkinkan kita menjalankan *single-node cluster* Kubernetes secara lokal. Langkah-langkah instalasi umumnya adalah sebagai berikut:

1. Sebelum memulai diharapkan untuk uncomment member dari group wheel.

```
## Uncomment to allow members of group wheel to execute any command
%wheel ALL=(ALL:ALL) ALL
```

2. **Instalasi kubectl:** Unduh dan instal kubectl sesuai dengan sistem operasi Linux yang digunakan.

```
0s ~ / /Kuliah/SIB/7#
curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
```

```
0s ~ / /Kuliah/SIB/7#
curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl.sha256"
```

```
11s ~ / /Kuliah/SIB/7#
echo "$(cat kubectl.sha256)  kubectl" | sha256sum --check
kubectl: OK
```

```
0s ~ / /Kuliah/SIB/7#
sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl
[sudo] password for ketopraksx:
```

```
0s ~ / /Kuliah/SIB/7#
kubectl version --client
Client Version: v1.34.1
Kustomize Version: v5.7.1
```

3. **Instalasi Minikube:** Unduh dan instal Minikube.

```
0s ~ / /Kuliah/SIB/7#
curl -LO https://github.com/kubernetes/minikube/releases/latest/download/minikube-linux-amd64
sudo install minikube-linux-amd64 /usr/local/bin/minikube && rm minikube-linux-amd64
```

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100  133M  100  133M    0     0  5383k      0  0:00:25  0:00:25 --:--:-- 5737k
[sudo] password for ketopraksx:
```

Setelah itu, kita bisa memulai *cluster* lokal dengan perintah:

```
minikube start
```

```
minikube start
W0919 21:14:15.789322    9409 main.go:291] Unable to resolve the current Docker CLI context "default": context "default": context not found: open /home/ketopraksx/.docker/contexts/meta/37a8eec1ce19687d132fe29051dca629d164e2c4958ba141d5f4133a33f0688f/meta.json: no such file or directory
W0919 21:14:15.789352    9409 main.go:292] Try running `docker context use default` to resolve the above error
😄  minikube v1.37.0 on Arch
✨  Automatically selected the docker driver. Other choices: virtualbox, none, ssh
📌  Using Docker driver with root privileges
👍  Starting "minikube" primary control-plane node in "minikube" cluster
🚜  Pulling base image v0.0.48 ...
💾  Downloading Kubernetes v1.34.0 preload ...
    > gcr.io/k8s-minikube/kicbase...:  488.52 MiB / 488.52 MiB  100.00% 3.27 Mi
    > preloaded-images-k8s-v18-v1...:  337.07 MiB / 337.07 MiB  100.00% 2.19 Mi
🔥  Creating docker container (CPUs=2, Memory=3072MB) ...
🐳  Preparing Kubernetes v1.34.0 on Docker 28.4.0 ...
🔗  Configuring bridge CNI (Container Networking Interface) ...
🔎  Verifying Kubernetes components...
    • Using image gcr.io/k8s-minikube/storage-provisioner:v5
🌟  Enabled addons: storage-provisioner, default-storageclass
🏄  Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
```

4. Perintah Dasar dalam Kubernetes

Setelah *cluster* berjalan, kita bisa menggunakan kubectl untuk berinteraksi dengannya. Berikut adalah beberapa perintah dasar yang sering digunakan:

- **Mengecek status *cluster***:

```
0s ~
kubectl cluster-info
Kubernetes control plane is running at https://192.168.49.2:8443
CoreDNS is running at https://192.168.49.2:8443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
```

- **Melihat status semua Pod**:

```
kubectl get pods
```

  Berikut adalah contoh *output* dari perintah tersebut:

```
NAME                                READY   STATUS              RESTARTS   AGE
nginx-deployment-7457467ffd-pn7ft   0/1     ContainerCreating   0          19s
nginx-pod                           0/1     ContainerCreating   0          14s
```

- **Melihat status semua *Service***:

```
kubectl get services
```

  Berikut adalah contoh *output* dari perintah tersebut:

```
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP   14m
```

- **Melihat semua *resource***:

```
kubectl get all
```

  Berikut ini adalah contoh outputnya:

```
NAME                                    READY   STATUS    RESTARTS   AGE
pod/nginx-deployment-7457467ffd-pn7ft   1/1     Running   0          101s
pod/nginx-pod                           1/1     Running   0          96s

NAME                 TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
service/kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP   15m

NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/nginx-deployment    1/1     1            1           101s

NAME                                           DESIRED   CURRENT   READY   AGE
replicaset.apps/nginx-deployment-7457467ffd    1         1         1       101s
```

Dengan memahami komponen dan perintah dasar ini, kita bisa memulai perjalanan dalam mengotomatisasi dan mengelola aplikasi di lingkungan kontainer dengan Kubernetes. Platform ini menjadi standar industri karena kemampuannya dalam menyediakan orkestrasi yang efisien, membuat proses *deployment* menjadi lebih cepat dan andal.