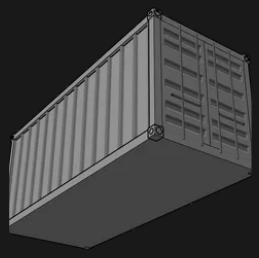


Pod 운영하기

1. Pod 란?

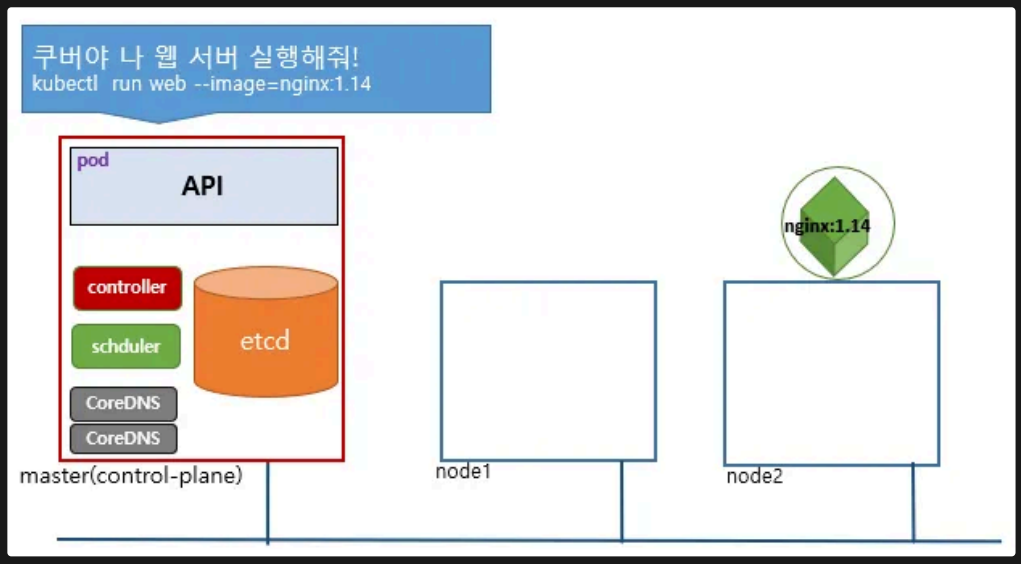
- 컨테이너를 표현하는 k8s API의 최소 단위
- Pod에는 하나 또는 여러 개의 컨테이너가 포함될 수 있음
- The pod is a group of one or more containers and the smallest deployable unit in Kubernetes.
- Each pod is isolated by the following Linux namespaces:
  - Process ID(PID) namespace
  - User namespace
  - Mount namespace
  - Interprocess Communication (IPC) namespace
  - Unix Time Sharing (UTS) namespace
  - Network namespace

▶ Namespace

2. Pod 동작 방법

CLI

- kubectl run NAME --image=image [--env="key=value"] [--port=port] [--dry-run=server|client] --[COMMAND] [args...]




```
kubectl run --help # web이라는 이름으로 nginx version 1.14 컨테이너 실행하기 kubectl run webserver --image=nginx:1.14 --port=80 kubectl get pods kubectl delete od webserver
```

Yaml

- yaml 파일을 미리 만든 후에 적용

```
kubectl run webserver --image nginx:1.14 --port 80 --dry-run=client -o yaml > test.yaml cat test.yaml apiVersion: v1 kind: Pod metadata: name: webserver spec: containers: - name: nginx image: nginx:1.14 ports: - containerPort: 80 kubectl apply -f test.yaml
```

3. Static Pod

- API 서버 없이 특정 노드에 있는 kubelet 에 의해 직접 관리
- static pod 디렉토리에 Pod yaml이 저장되면 kubelet 이 동작 시킴. 파일 삭제 시 kubelet은 삭제함.

- kubelet 의 configuration file: /var/lib/kubelet/config.yaml

```
ssh k8s-worker1 vi /var/lib/kubelet/config.yaml … staticPodPath:
/etc/kubernetes/manifests
```

hk8s-worker1에 static Pod 만들기
① ssh로 hk8s-worker1 서버에 접속
② sudo -i로 root 권한 얻기
③ /etc/kubernetes/manifests 디렉토리에 pod 파일 생성하기

4. Multi container Pod

- Multi-container Pod

multi-container pod

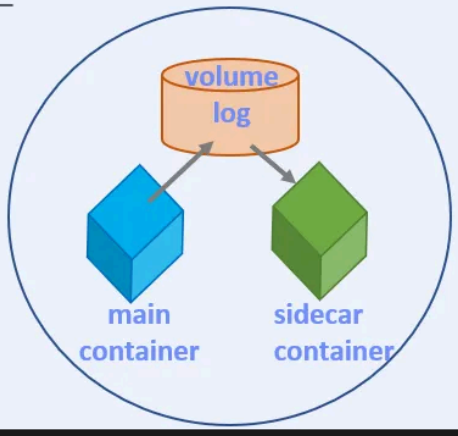
하나의 Pod에 여러 개의 컨테이너가 포함하여 실행


```
apiVersion: v1
kind: Pod
metadata:
  name: two-containers
spec:
  containers:
  - name: nginx-container
    image: nginx
    volumeMounts:
    - name: shared-data
      mountPath: /usr/share/nginx/html
  - name: debian-container
    image: debian
    volumeMounts:
    - name: shared-data
      mountPath: /pod-data
    command: ["/bin/sh"]
    args: ["-c", "echo Hello from the debian container > /pod-data/index.html"]
  restartPolicy: Never
  volumes:
  - name: shared-data
    emptyDir: {}
```

- Side-car container

기본 컨테이너 기능을 확장하기 위해 사용
본래의 컨테이너는 기본 서비스에 충실하고, 추가 기능을 별도의 컨테이너를 이용해 적용

웹서버는 기본서비스에 충실하고, 추가로 생성되는 웹서버 로그는 별도의 사이드 카 컨테이너가 수집하여 외부 log aggregator로 전송하는 형태의 서비스




```
apiVersion: v1
kind: Pod
metadata:
 name: webserver
spec:
 volumes:
 - name: shared-logs
  emptyDir: {}

 containers:
 - name: main
  image: nginx
  volumeMounts:
  - name: shared-logs
   mountPath: /var/log/nginx

 - name: sidecar-container
  image: busybox
  command: ["sh","-c","while true; do cat
/log/access.log /log/error.log; sleep 10; done"]
  volumeMounts:
  - name: shared-logs
   mountPath: /log
```

5. 네임스페이스

- namespace : 단일 클러스터 내에서의 리소스 그룹 격리 메커니즘을 제공한다. 리소스의 이름은 네임스페이스 내에서 유일해야 하며, 네임스페이스 간에서 유일할 필요는 없다. 네임스페이스 기반 스코핑은 네임스페이스 기반 오브젝트 *(예: 디플로이먼트, 서비스 등)* 에만 적용 가능하며 클러스터 범위의 오브젝트 *(예: 스토리지 클래스, 노드, 퍼시스턴트볼륨 등)* 에는 적용 불가능하다

6. 문제 풀이

❓ 1. Pod 생성하기 2%

- 작업 클러스터 : k8s
  - kubectl config use-context k8s
- Create a new namespace and create a pod in the namespace

- TASK:
  - namespace name: **cka**
  - pod Name: **pod-02**
  - image: **nginx:alpine**
  - resource request
    - cpu: 100m
    - memory : 100Mi

▶ 답

❓ 2. Static Pod 생성하기(CKAD)

- Configure kubelet hosting to start a pod on the node
- TASK:
  - Node: hk8s-worker1
  - pod Name: web01
  - image: nginx

❓ 3. multi-container Pod 생성 4%

- 작업 클러스터 : hk8s
  - kubectl config use-context hk8s
- Create a pod with 3 containers running: nginx, redis and memcached
  - pod name: `pod3`
    - image: nginx
    - image: redis
    - image: memcached

▶ 답

❓ 4. Side-car Container Pod 실행 7%

> 검색 키워드 : sidecar

- 작업 클러스터 : **k8s**
  - **kubectl config use-context k8s**

- 현재 운영 중인 `eshop-cart-app` Pod의 로그를 Kubernetes built-in logging 아키텍처(예: kubectl logs)에 통합하는 로그 스트리밍 사이드카 컨테이너를 운영하시오.
  - `busybox` 이미지를 사용하여 `price` 라는 이름의 사이드카 컨테이너를 기존 `eshop-cart-app`에 추가합니다.