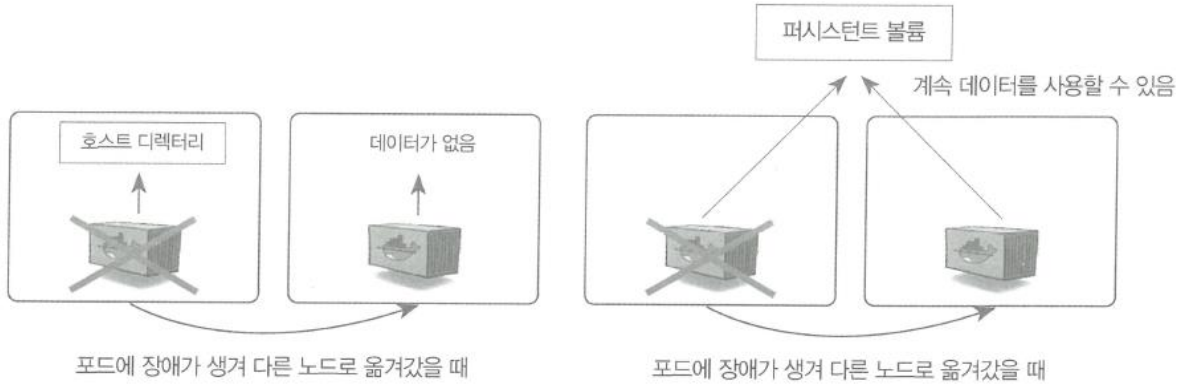
< Kubernetes PV, PVC >

쿠버네티스에서 stateful 애플리케이션의 경우 파드의 데이터를 영속적으로 저장하기 위한 방법이 필요합니다. 쿠버네티스에서 호스트에 위치한 디렉터리를 각 파드와 공유함으로써 데이터를 보존하는 것이 가능하지만 여러 개의 서버로 구성된 쿠버네티스와 같은 클러스터 환경에서는 특정 노드에 데이터를 저장해 놓으면 파드가 다른 노드로 옮겨갔을 때 데이터를 사용할 수 없게 되는 문제가 생깁니다.



PersistentVolume 은 워커 노드들이 네트워크 상에서 스토리지를 마운트해 영속적으로 데이터를 저장할 수 있는 볼륨을 의미합니다. 따라서 파드에 장애가 생겨 다른 노드로 옮겨가더라도 해당 노드에서 퍼시스턴트 볼륨에 네트워크로 연결해 데이터를 계속 사용할 수 있습니다.

1. 로컬 볼륨: hostPath, emptyDir

1.1) hostPath: 워커 노드의 로컬 디렉터리를 볼륨으로 사용

호스트의 디렉터리를 파드와 공유해 데이터를 저장하기 위해 사용합니다.

hostPath 는 디플로이먼트의 파드에 장애가 생겨 다른 노드로 파드고 옮겨가는 경우 이전 노드에 저장된 데이터를 사용할 수 없게 됩니다. 모든 워커노드에 배포해야하는 CAdvisor 과 같은 모니터링 툴을 사용할 경우 hostPath 를 사용하는 것이 옳지만 이런 경우를 제외한다면 바람직하지 않은 방법입니다.

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: hostpath-pod
spec:
  containers:
  - name: my-container
    image: busybox
    args: [ "tail", "-f", "/dev/null"]
    volumeMounts:
    - name: my-hostpath-volume
      mountPath: /etc/data
  volumes:
  - name: my-hostpath-volume
    hostPath:
    path: /tmp
```

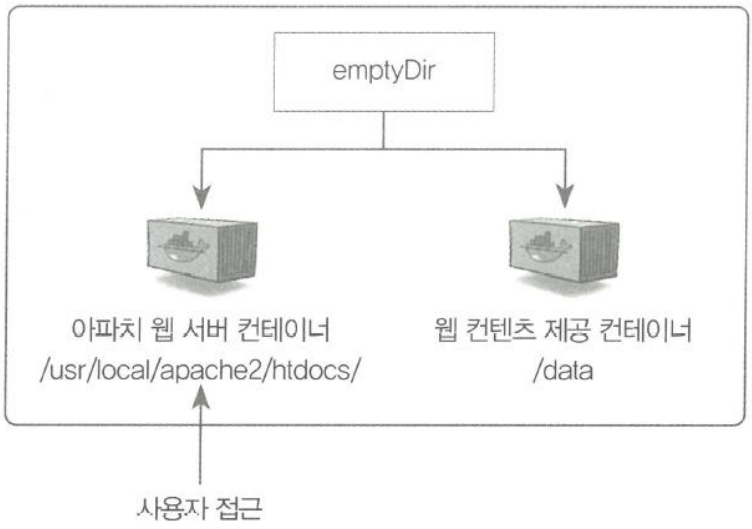
파드의 /etc/data 디렉터리에 호스트의 /tmp 를 마운트한 설정 파일입니다.

1.2) emptyDir : 파드 내의 컨테이너 간 임시 데이터 공유

파드가 실행되는 도중에만 필요한 휘발성 데이터를 각 컨테이너가 함께 사용할 수 있도록 임시 저장 공간을 생성합니다. 파드의 컨테이너 간에 볼륨을 공유하기 위해서 사용합니다.

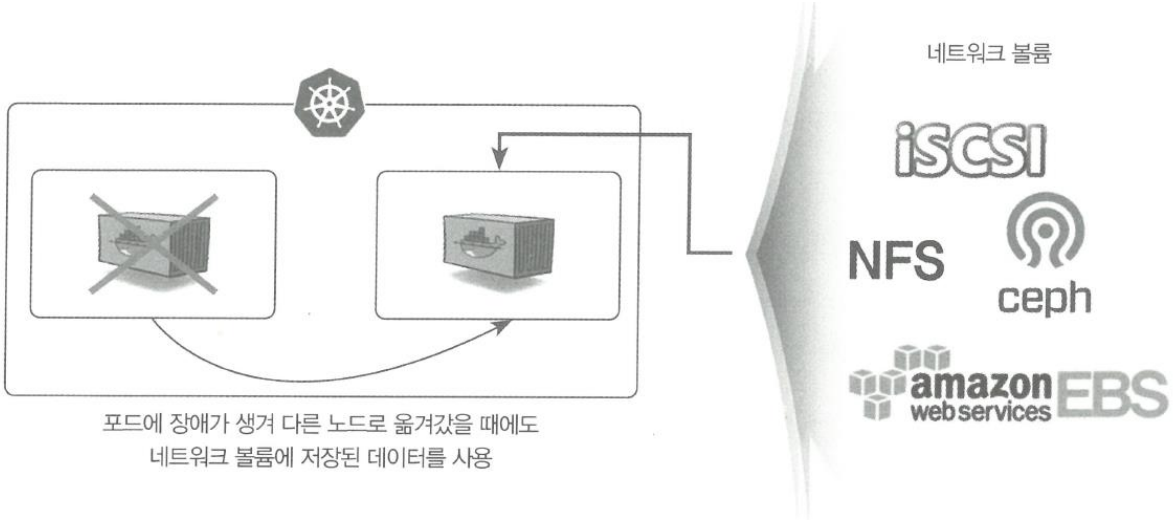
한 컨테이너가 파일을 관리하고 한 컨테이너가 그 파일을 사용하는 경우에 유용하게 사용할 수 있습니다.

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: emptydir-pod
spec:
  containers:
  - name: content-creator
    image:    suejin/alpine-wget:latest
    args: ["tail" , "-f" , "/dev/null"]
    volumeMounts:
    - name: my-emptydir-volume
      mountPath: Idata # 1. 이 컨테이너가 Idata에 파일을 생성하면
    - name: apache-webserver
      image: httpd:2
      volumeMounts:
      - name: my-emptydir-volume
        mountPath: lusr/local/apache2/htdocsl # 2. 아파치 웹 서버에서 접근할 수 있습니다.
    volumes:
      - name: my-emptydir-volume
        emptyDir: {}
```



2. 네트워크 볼륨

쿠버네티스에서는 별도의 플러그인을 설치하지 않아도 다양한 종류의 네트워크 볼륨을 파드에 마운트 할 수 있습니다. 온프레미스 환경에서도 구축할 수 있는 NFS, iSCSI 와 같은 네트워크 볼륨뿐만 아니라 AWS 의 EBS(Elastic Block Store), GCP 의 gcePersistentDisk 와 같은 클라우드 플랫폼의 볼륨을 파드에 마운트 할 수 있습니다.

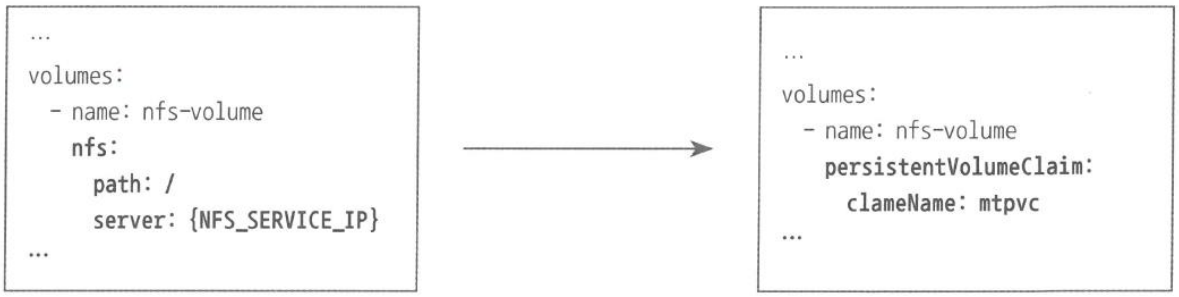


3. PV, PVC

PV(Persistent Volume) PVC(PersistentVolumeClaim) 오브젝트는 파드를 생성하는 YAML 입장에서 네트워크 볼륨에 상관없이 사용할 수 있도록 추상화해주는 역할을 합니다.

PersistentVolumeClaim 리소스는 영속화 영역을 이용하기 위한 리소스입니다

Config&Storage 리소스의 한 종류인 PersistentVolumeClaim 을 설명하기 전에 Volume 과 PersistentVolume, PersistentVolumeClaim 의 차이를 이해할 필요가 있습니다.

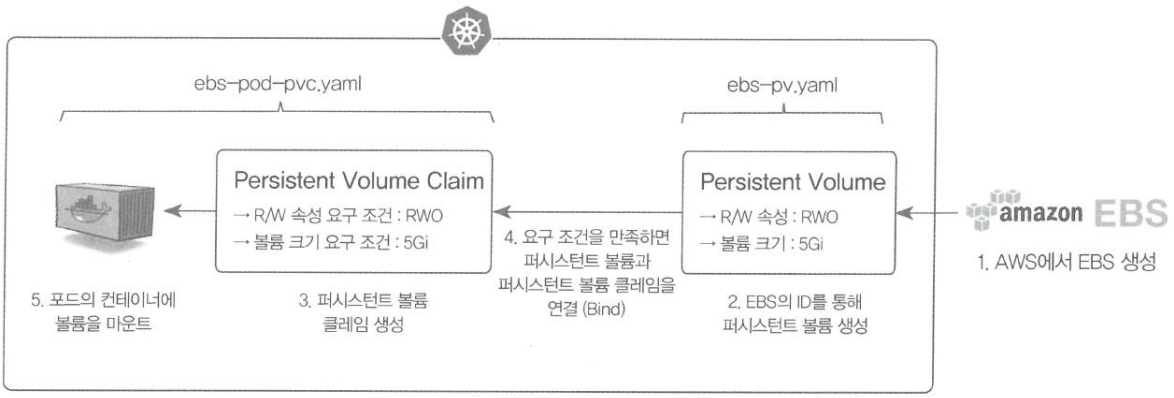


mysql-deployment-nfs.yaml　　mysql-deployment.yaml

Volume 은 미리 준비된 이용가능한 볼륨 (호스트의 영역/NFS/iSCSI) 등을 매니페스트로 직접 지정하여 이용 가능하게 하는 것입니다. 때문에 이용자가 Kubernetes 상에서 신규로 볼륨을 작성한다거나, 기존의 볼륨을 제거한다거나 하는 조작을 실행할 수 없습니다. 또한 매니패스트에서 Volume 리소스를 작성한다거나 하는 처리도 실행할 수 없습니다.

그래서 PersistentVolume 은 외부의 영속 볼륨을 제공하는 시스템과 연계하여 신규 볼륨의 작성과 기존 볼륨의 제거 등이 가능합니다. 구체적으로는 매니페스트 등에서 PersistentVolume 리소스를 별도 작성하는 형태로 됩니다.

PersistentVolume 과 Volume 에도 같은 플러그인이 준비되어 있습니다. 예를 들면 GCP (Google Cloud Platform)과 AWS(Amazon Web Service)의 볼륨 서비스에서는 PersistentVolume 플러그인과 Volume 플러그인 모두 준비되어 있습니다. PersistentVolume 플러그인에는 Volume 의 작성과 제거라고 하는 라이프 사이클을 처리하는 것이 가능 (PersistentVolumeClaim 을 이용하면 Dynamic Provisioning 도 가능) 하지만 Volume 플러그인의 경우에는 이미 있는 Volume 을 이용한 것만 가능합니다.



AWS 클러스터 위에서 생성돼 있는 쿠버네티스 클러스터

PersistentVolumeClaim 은 그 이름대로 작성된 PersistentVolume 리소스 중에서 Assign 하기 위한 리소스가 됩니다. PersistentVolume 은 클러스터에 볼륨을 등록하는 것뿐이라서 실제로 Pod 에서 이용하기 위해서는 PersistentVolumeClaim 을 정의하여 이용할 필요가 있습니다.

4. 쿠버네티스에서 Persistent Volume 사용하기 (NFS)

4.1) NFS 서버 준비 후 PV, PVC 생성

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: nfs-pc
spec:
  capacity:
    storage: 2Gi
  accessModes:
    - ReadWriteMany
  nfs:
    server: 10.104.175.113
    path: /suejin
```

```yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: nfs-pvc
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 2Gi
```

```
[ Suejin@kubernetes-master ch09]$ k get pv -o wide
NAME     CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM                 STORAGECLASS   REASON   AGE   VOLUMEMODE
nfs-pv   2Gi        RWO            Recycle          Bound    default/my-nfs-pvc                   31m   Filesystem
```

4.2) PVC 를 사용하는 Deployment 생성

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: dpl-nginx
spec:
  selector:
    matchLabels:
      app: dpl-nginx
  replicas: 2
  template:
    metadata:
      labels:
        app: dpl-nginx
    spec:
      containers:
      - name: master
        image: nginx
        ports:
        - containerPort: 80
        volumeMounts:
        - mountPath: /mnt
          name: pvc-volume
      volumes:
      - name: pvc-volume
        persistentVolumeClaim:
          claimName: nfs-pvc
```

```
10.104.175.113:/suejin          2.0G   12K  2.0G   1% /var/lib/kub
```

Reference

1. Storage - https://kubernetes.io/docs/concepts/storage/#types-of-volumes
2. Storage – '컨테이너 인프라 환경 구축을 위한 쿠버네티스/도커' 책 내용 공부하여 정리
3. Pv, pvc - https://kubernetes.io/ko/docs/concepts/storage/persistent-volumes/
4. Pv, pvc - https://thecodingmachine.tistory.com/15?category=852770
5. Pv, pvc - https://m.blog.naver.com/alice_k106/221360005336