2024.07.22

오전

k8s 시스템에서 사용되는 : kube-system
k8s pod와 deployment는 spec, status등의 값을 가지고 있음.
이러한 값을 가지고 있는 pod와 deployment를 개별 속성을 포함해 부르는 단위를 Object 정의함
k8s 여러 유형의 object를 제공
***기본적으로 파드는 디렉토리가 임시로 사용. 저장과 보존이 필요함 : Volume
Pod 접속을 내/외부 연결하는 오브젝트 : Service
- ClusterIp, NodePort, LoadBalance

masternode 의 맨위에
drwxrwxr-x 4 encore encore 4096 Jul 17 01:23 .kube/
scp -r ./.kube/ myserver02:/home/encore

kube 명령어가 먹힐려면 ./.kube가 있어야한다.
즉 masternode에 있는 ./.kube를 scp -r로 보내야한다.
scp -r ~/.kube/ myserver02:/home/encore/

pki는 인증서.

초기화하기..
.kube폴더를 지워줘야한다.

creaete -f
apply -f

kube-proxy
쿠버네티스 클러스트는 파드가 위치한 노드에 kube-proxy를 통해 파드가 통신할 수 있는 네트워크를 설정
이때 실제 통신은 br_netfilter와 iptables로 관리

브리지 관리(Bridge Management)
리눅스 커널에서 네트워크 브리지를 관리하는 사용
물리적 또는 가상의 네트워크 인터페이스를 연결하고 이를 통해 패킷을 전달하는 역할을 함
br_netfilter는 이러한 브리지 동작을 가능하게 하며, 다양한 네트워크 인터페이스를 연결하여 네트워크 트래픽을 브리지를 통해 중계하고 분배

modeprobe -r br_netfilter
wokrder node에서 kubelet

sudo systemctl status kubelet
notready로 바뀐다
sudo systemctl stop kubelet
kubelet은 마스터의 kube-apisever와 통신하면서 파드의 생성, 관리, 삭제를 담당

masternode에서
k get nodes

statefulset

deployment하고 비슷한거
각 pod마다 독자성유지
pod관리 object
**

- 사용하는 경우
- 인정적이고, 고유한 네트워크 식별자
- 인정적인 Persistent Storage
- 순차적인 원할한 배포와 스케일링
  - 순차적, 자동 롤링 업데이트
  apiVersion: v1
  kind: Service
  metadata:
  name: sfs-server01
  spec:
  selector:
  app.kubernets.io/name: web-sfs01
  type: ClusterIP
  clusterIP: None
  ports:
  - protocol: TCP
  port: 80
  selecdtor web-sfs01를 statefulset가 연동
  cluastip none 헤드리스 서비스로 설정
  헤드리스 pod 개별네트워크 식별서비스
  kubectl get svc**

**
apiVersion: apps/v1
kind: StatefulSet

metadata:
name: sfs-test01
spec:
replicas: 3
selector:
matchLabels:
app.kubernetes.io/name: web-sfs01
serviceName: sfs-service01
template:
metadata:
labels:
app.kubernetes.io/name: web-sfs01
spec:
containers:
- name: nginx
image: nginx:latest
selector 를 통하여 pod를 하나로 묶음

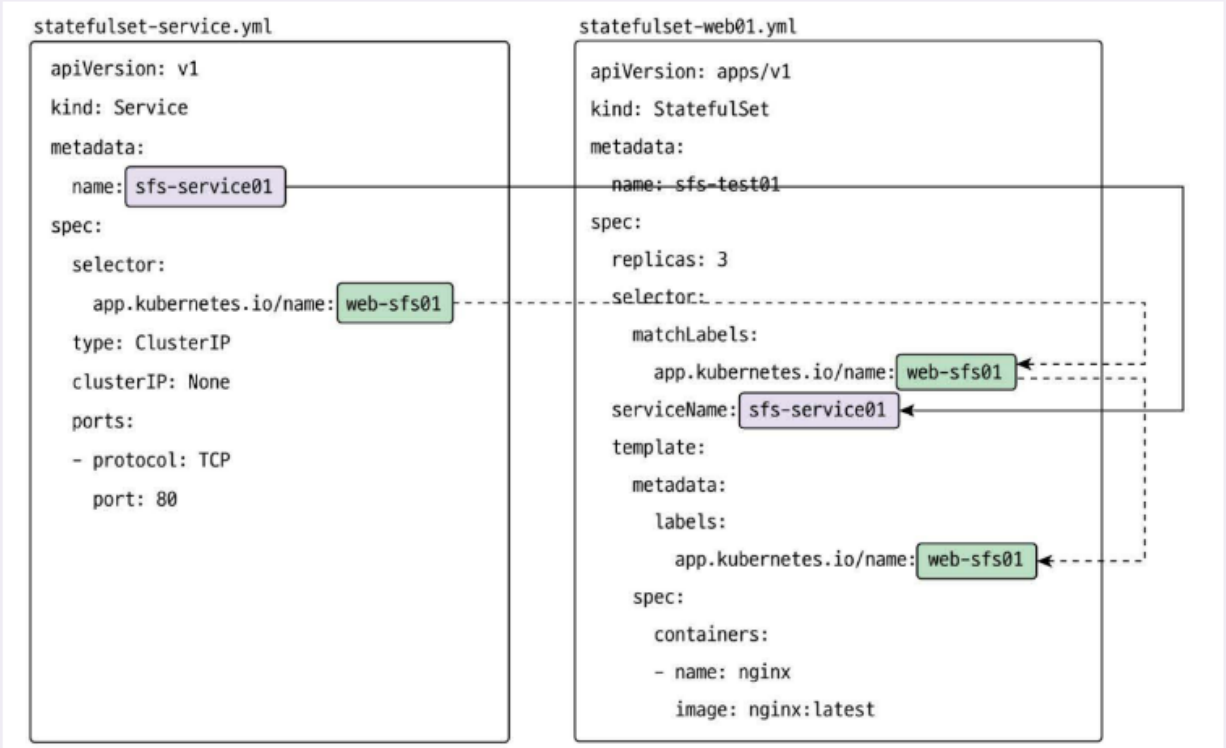

그림 9-31 스테이트풀셋과 헤드리스 서비스 연결

복사해서 replicas:2로 변경
그러면 2번이 사라졌다..순차적이므로
/bin/bash들어가서 curl ip

k delete pods --all
k delete services,statefulsets --all

kubectl delete deployment --all

statefulset volume

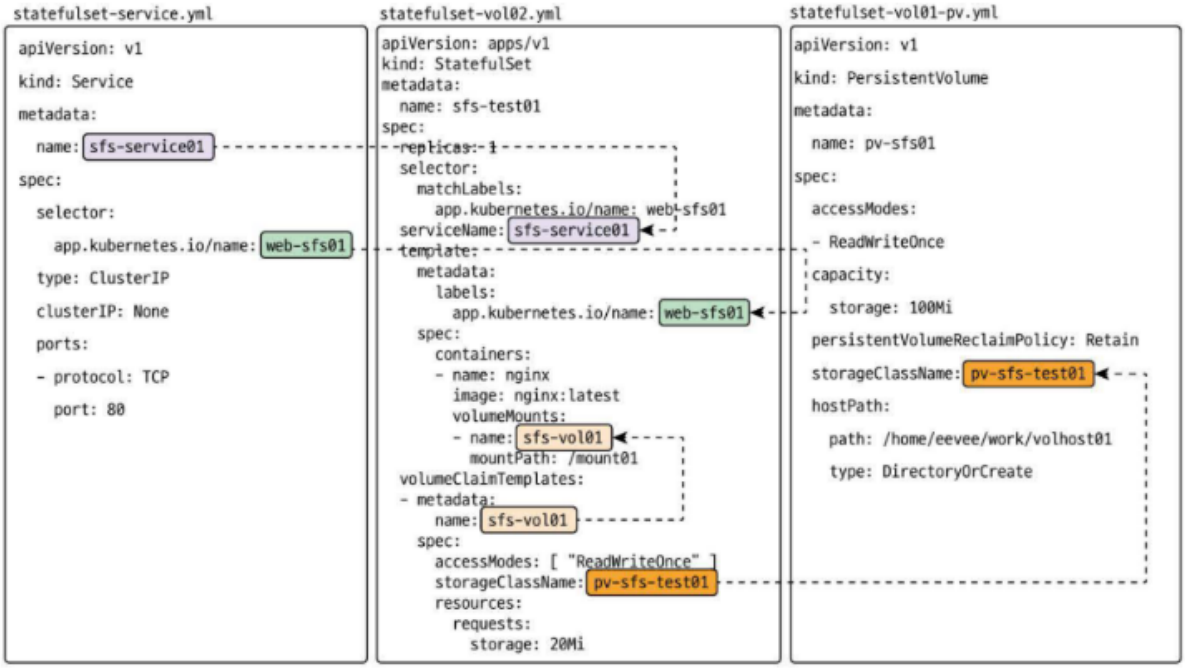

그림 9-32 스테이트풀셋 볼륨 yml 파일 관계

**apiVersion: v1
kind: PersistentVolume
metadata:
name: pv-sfs01
spec:
accessModes:

- ReadWriteOnce
  capacity:
  storage: 100Mi
  persistentVolumeReclaimPolicy: Retain
  storageClassName: pv-sfs-test01
  hostPath:
  path: /home/encore/work/volhost01
  type: DirectoryOrCreate

apiVersion: apps/v1
kind: StatefulSet

metadata:
name: sfs-test01
spec:
replicas: 1
selector:
matchLabels:
app.kubernetes.io/name: web-sfs01
serviceName: sfs-service01
template:
metadata:
labels:
app.kubernetes.io/name: web-sfs01
spec:
containers:
- name: nginx
image: nginx:latest
volumeMounts:
- name: sfs-vol01
mountPath: /mount01
volumeClaimTemplates:
- metadata:
name: sfs-vol01
spec:
accessModes: ["ReadWriteOnce"]
storageClassName: pv-sfs-test01
resources:
requests:
storage: 20Mi

pv가 100m 빌렸는데 임대는 20m

서비스먼저 올리고
pv올리고

뒤즌게 replicaset을 바꾸면 pendiㅎ상태가 된다..
watch -n 1 kubectl get pod,pv,pvc

k describe pvc sfs-vol01-sfs-test01-1
이미 먼저 실행한 자원이 쓰고있다...
pv 동적으로 할당하기 위해서는 Rook 을 사용하면 새롱ㄴ pvc가 생성될때 적절한 pv ㅑ가 자동생성되고 바인딩됨
지울때는 거꾸로...

k delete -f statefulset-vol02.yaml
k delete pv,pvc --all
k delete pv,pvc --all

헬름(helm)

pod svc 여러 yaml 생성
helm k8s 클러스터에 application 배포를 위해 파일들을 하나의 패지지 형태로 관리
yaml을 만들지 않고도 k8s에서 어플리케이션 쉽고 편ㄴ하게 설치

chart

k8s 리소스 생성위 | 한 파일디렉토리
helm repoistory -> 다양한 helm차트 공유
docker hub ->docker reposiotry
helm chart 다운로드 -> 디렉토리 생성 -> 커스텀시켜서 사용

/usr/bin/bash script파일

chmod 700 get_helm.sh
./get_helm.sh
helm version

helm repo add bitnami https://charts.bitnami.com/bitnami
helm repo update
helm repo list
mkdir nginx-ingress-controller && cd nginx-ingress-controller
helm search repo nginx
helm pull bitnami/nginx-ingress-controller
tar xvfz nginx-ingress-controller-11.3.15.tgz

```
encore@myserver01:~/nginx-ingress-controller/nginx-ingress-controller$ ls
Chart.lock  charts  Chart.yaml  README.md  templates  values.yaml
```

맨마지막 대문자 G
맨처음 소문자 gg
cp values.yaml ./my-values.yaml
k delete namespace ingress-nginx
kubectl delete namespace ingress-nginx --grace-period=0 --force
kubectl get namespace ingress-nginx -o json | jq '.spec.finalizers=[]' | kubectl replace --raw "/api/v1/namespaces/ingress-nginx/finalize" -f -

helm install --namespace mynginx --generate-name bitnami/nginx-ingress-controller -f ./my-values.yaml

//에러시 ingress-nginx class가있다.
k get ingressclass
k delete ingressclass nginx

helm ls --namespace mynginx - 명령어 인식이 되는가

kubectl get all --namespace mynginx

MetalLb
온프레미스(On-premises) 에서 로드밸런서를 사용하려면 내부에 로드밸런서 서비스를 받아주는 구성이 필요

- 이걸 지원해주는 MetalLB
- 베어메탈(bare metal)로 구성된 k8s에서 로드밸런서를 사용할 수 있도록 고안된 프로젝트
- L2 L3 네트워크 로드밸런서로 굿현
  **

ARP
ARP(Address Resolution Protocol)는 IP 및 MAC Address 사이의 상관관계를 결정하는 데 사용
하나의 IP 노드가 특정 목적지를 지정하길 원하지만 MAC Address를 알지 못하는 경우, ARP를 사용하여 요청
이를 위해 발신 노드는 해당 네트워크로 ARP 요청을 발송
응답을 수신한 후(ARP Response), 포함된 MAC Address를 ARP 캐시에 저장한 후 전달
NDP
NDP(Neighbor Discovery Protocol)는 IPv6를 사용 시 ARP 프로토콜을 대체하며 ICMPv6에 기초를 두고 있음
프로토콜은 다음과 같은 목적으로 사용됩니다.
Router Discovery(라우터 찾기): 네트워크 내에 있는 기존의 라우터를 식별
Prefix Discovery(프리픽스 찾기): 로컬 및 원격 노드에 대한 주소 프리픽스(IPv6 Address의 네트워크 bit)의 결정
DHCP를 사용하지 않으면서 네트워크에 대한 IPv6 주소의 자동 설정 지원(Local Address Generation 링크).
Parameter Discovery(파라미터 찾기): Hop Limit와 같은 다양한 파라미터의 설정
BGP
Border Gateway Protocol의 약자로 서로 다른 AS를 연결해 주는 경계 게이트웨이 프로토콜
**

- LoadBalancer 구성
kubectl get configmap kube-proxy -n kube-system -o yaml

kubectl get configmap kube-proxy -n kube-system -o yaml | grep strictARP

kubectl get configmap kube-proxy -n kube-system -o yaml | sed -e "s/strictARP: false/strictARP: true/" | kubectl apply -f - -n kube-system

cd..
mkdir metallb && cd metallb
/home/encore/nginx-ingress-controller/metallb
helm repo add metallb https://metallb.github.io/metallb
helm repo update

**
helm search repo metallb
helm pull metallb/metallb
압축풀기 tar xvzf

kubectl create namespace mymetallb
helm install --namespace mymetallb --generate-name metallb/metallb -f ./my-values.yaml
k get all --namespace mymetallb

0/4 Evicted 0 20s

k describe pod metallb-1721626474-speaker-kpjtj --namespace mymetallb

kubectl get configmap --namespace mymetallb

k delete namespace mymetallb
kubectl delete namespace mymetallb --grace-period=0 --force
kubectl get namespace mymetallb -o json | jq '.spec.finalizers=[]' | kubectl replace --raw "/api/v1/namespaces/mymetallb/finalize" -f -

해결법

kubectl delete crd bfdprofiles.metallb.io
kubectl delete crd bgpadvertisements.metallb.io
k delete crd communities.metallb.io
k delete crd bgppeers.metallb.io
k delete crd ipaddresspools.metallb.io
k delete crd l2advertisements.metallb.io
k delete crd servicel2statuses.metallb.io
kubectl delete validatingwebhookconfiguration metallb-webhook-configuration
kubectl delete secret metallb-webhook-cert -n mymetallb

sudo journalctl --vacuum-time=7d

k get all --namespace mymetallb

차트를 설치할때 crd deployment service configmap 리소스 생성, webhook 설치