(23.01.29) Service, Ingress

Service

서비스란?

서비스는 팟을 통해 실행되고 있는 어플리케이션을 네트워크에 노출시키는 가상의 컴포넌트입니다. 네트워크에 노출시킴으로써 어플리케이션은 내부의 다른 어플리케이션이나 외부의 어플리케이션 또는 사용자와 통신이 가능해집니다.

또 다른 관점으로는, (클러스터 안의 수많은 노드들 중 어딘가에 존재할-동일한 기능을 하는) 팟들의 논리적인 집합입니다. 이게 무슨 뜻일까요?

쿠버네티스 클러스터에서 팟은 언제든 삭제되고 생성될수 있기 때문에 **계속 아이피가 달라집니다.** 서비스를 생성하면 클러스터 내부의 유일한 IP가 할당이 됩니다. (clusterIP라고 불립니다.) 서비스가 살아있는 동안 이 아이피는 변하지 않습니다.

팟들을 서비스와 통신하도록 구성하고, 이제 서비스로 들어온 요청은 팟들로 로드밸런싱 됩니다.

서비스는 팟들로 향하는 일종의 진입점 or 로드밸런서 역할을 하게됩니다.

서비스 생성해보기

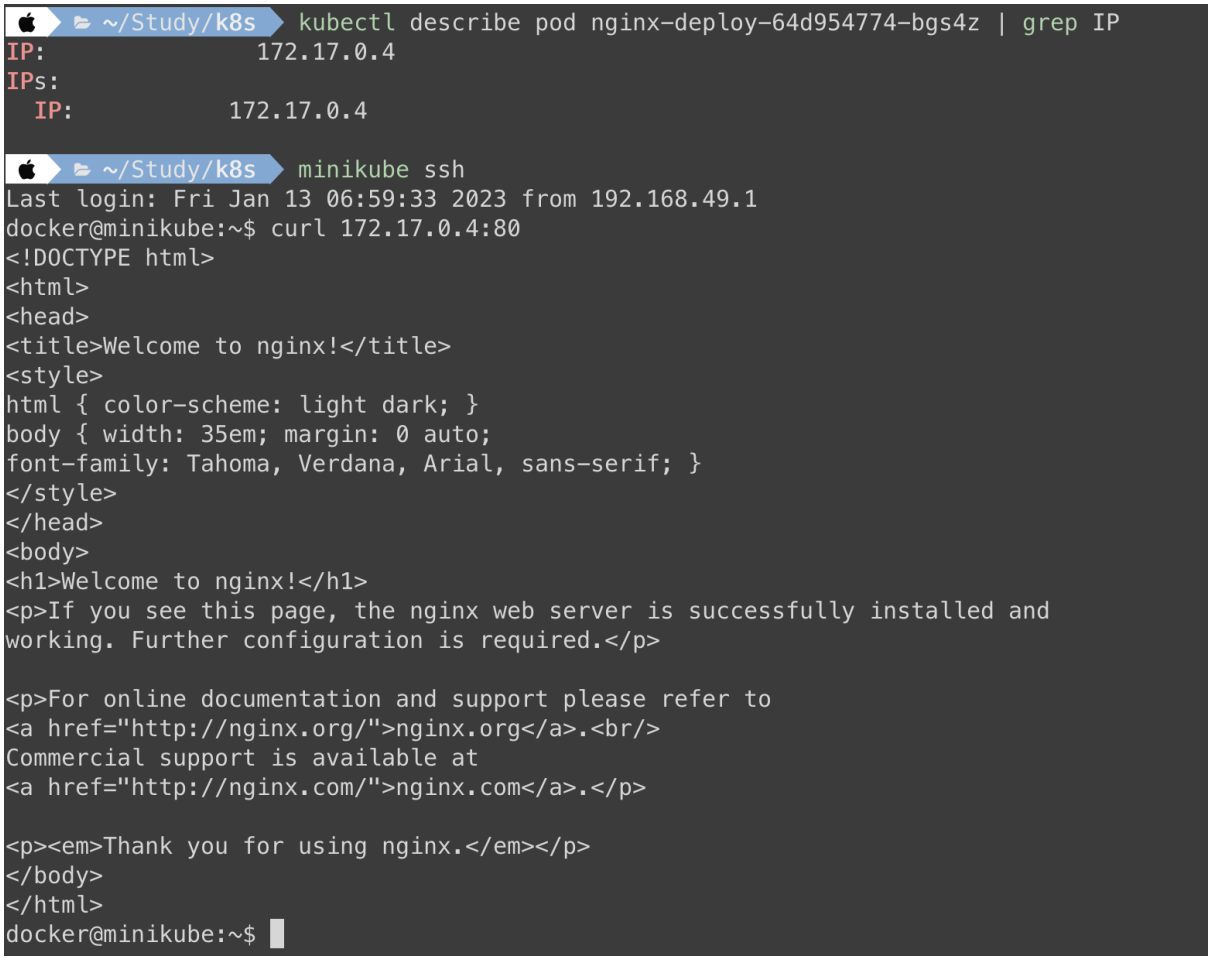
쿠버네티스 클러스터에 웹서버 팟을 배포했다고 가정해봅시다. 우리는 어떻게 브라우저를 통해 이 웹페이지에 접근할 수 있을까요?

먼저 지난번에 배포했던 nginx 컨테이너를 포트에 바인딩 시켜보겠습니다.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deploy
```

```
  spec:
    replicas: 3
    template:
      metadata:
        name: nginx-pod
        labels:
          app: nginx-demo
      spec:
        containers:
          - name: nginx
            image: nginx
            ports:
            - containerPort: 80
    selector:
      matchLabels:
        app: nginx-demo
```

팟들은 아이피를 가지고 있습니다. 하지만 우리는 팟의 내부 아이피로는 ping을 할 수 없습니다. 네트워크가 다르기 때문이죠.



팟의 내부 아이피 172.17.0.4 로 ping을 하려면 위 캡쳐처럼 쿠버네티스 노드에 직접 연결해서 해야합니다.

서비스의 유스케이스 중 하나는 노드의 포트를 리스닝해서, 요청을 팟의 포트로 포워딩 시켜주는 것입니다. 이런 종류의 서비스를 NodePort 서비스라고 합니다. 이것 포함 4가지 종류의 서비스가 있지만 나중에 설명하겠습니다.

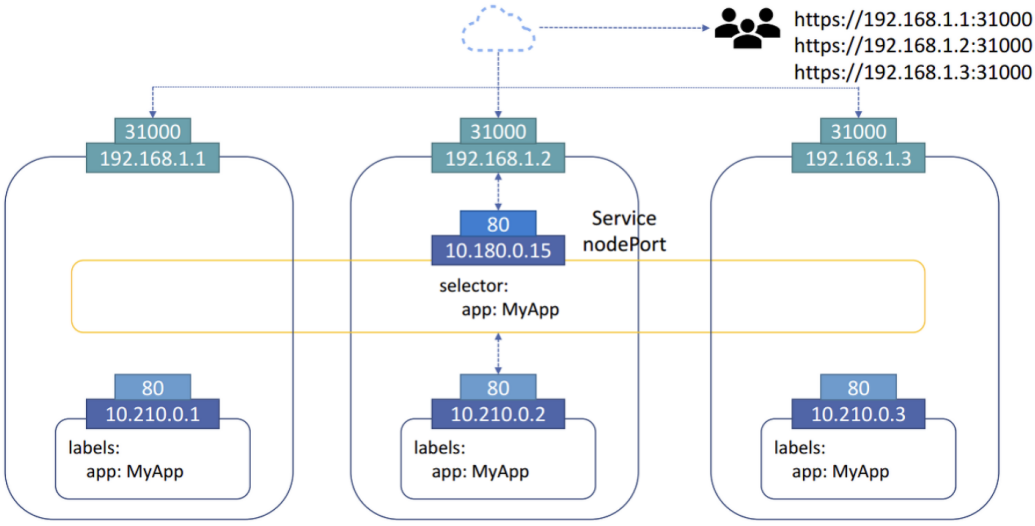
일단은 서비스를 생성해보겠습니다.

```
apiVersion: v1
kind: Service
metadata:
  name: nginx-service
spec:
  type: NodePort
  ports:
    - targetPort: 80
      port: 80
      nodePort: 31000
  selector:
    app: nginx-demo
```

- TargetPort (팟의 포트) : 실제 웹 서버가 동작하고 있는 포트
- Port (서비스의 포트) : 서비스는 사실 쿠버네티스 노드 안의 (일종의) Virtual 서버이기 때문에 클러스터 안에서 자체 아이피를 가집니다. 이것을 Cluster IP라고 부릅니다. `<ClusterIP>:<Port>` 를 통해서 TargetPort로 연결이 됩니다.
- NodePort (노드의 포트) : 웹서버를 외부에서 접속할 수 있게 하는 포트로, Valid Range는 30000~32767 입니다. 만약 각 노드들이 publicIP를 가지고 있다면, `<각 노드의 publicIP>:<NodePort>` 를 통해서 접근할 수 있습니다.

여기서 필수 포트는 포트 (서비스의 포트) 만 입니다. TargetPort 는 기재하지 않으면 Port랑 똑같은 값이, NodePort 는 30000~32767 범위중 가능한 포트를 자동으로 할당합니다.

서비스는 자신이 관리할 대상을 selector - label 방식으로 관리하기 때문에, selector를 YAML에 기입해두었습니다. 팟이 한개든, 혹은 deployment를 통해 여러개가 생성이 되었든 상관이 없습니다. selector만 잘 기입해주었다면 됩니다. 그럼 로드밸런싱 같은 것들은 어떻게 되는걸까요? 기본적으로 Random 알고리즘을 사용해서 로드밸런싱을 해줍니다. 만약 총 3개의 노드에 걸쳐 팟이 배포가 되어있다면 - `<노드1아이피>:<NodePort>` 를 통해 요청이 들어왔어도 트래픽 상황이 가장 좋은 노드로 접속하게 됩니다. (selector - label 을 통해 논리적으로 묶여있기 때문입니다.)



위와 같은 구성이 됩니다.

그럼 이제 `localhost:30008` 로 접속하면 nginx에 접속될까요? 아닙니다. 우리는 minikube를 통해 단일 노드 쿠버네티스 클러스터를 띄웠는데요, 해당 노드에는 외부 IP가 매핑되어 있지 않습니다.

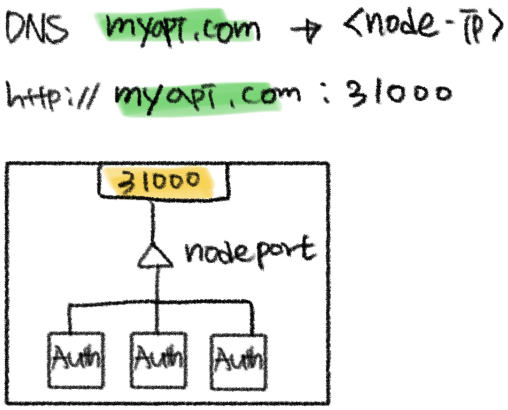
```
~/Study/k8s kubectl get nodes | grep ExternalIP -C 1
~/Study/k8s
```

만약 외부 IP가 매핑되어있는 노드가 있었다면 결과물이 출력되었을 것입니다.

서비스의 종류

- ClusterIP : 쿠버네티스 클러스터안에 virtual IP(클러스터 전용 내부 IP)를 생성함으로서 팟들이 클러스터 내부의 다른 리소스들과 통신할 수 있도록 하는 서비스. (External access는 없다.) 만약 서비스를 생성하는 yaml을 작성할때 type을 명시해주지 않으면 이것으로 생성된다.
- NodePort : 외부에서 팟에 접근할 수 있도록 해주는 서비스.
- LoadBalancer : 별도의 외부 로드 밸런서를 제공하는 클라우드 환경을 고려한 서비스로, 외부 로드밸런서를 이 서비스로 프로비저닝 해서 사용한다.
- ExternalName: selector 대신 DNSname을 직접 명시하고자 할 때 쓰는 서비스

Ingress

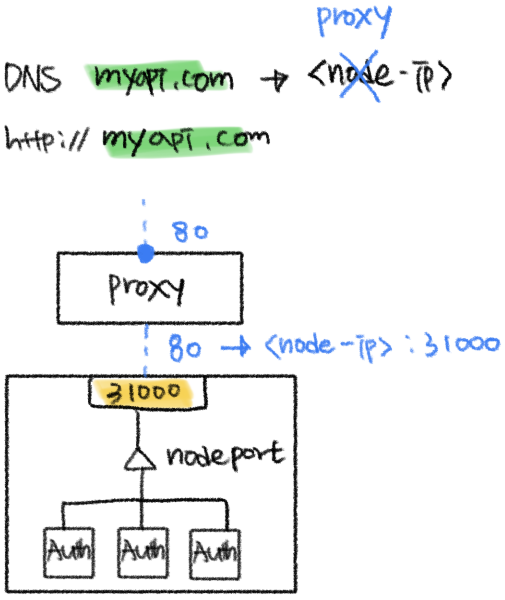


Auth api를 만들고 외부에서 접근가능하도록 해봅시다.

Nodeport 서비스를 통해 31000 포트에 노출합니다. `http://<node-ip>:31000` 을 통해 브라우저에서 접근 가능해집니다.

`<node-ip>` 대신 URL 으로 접근 가능하도록 DNS에 `myapi.com` 에 `<node-ip>` 를 등록합니다.

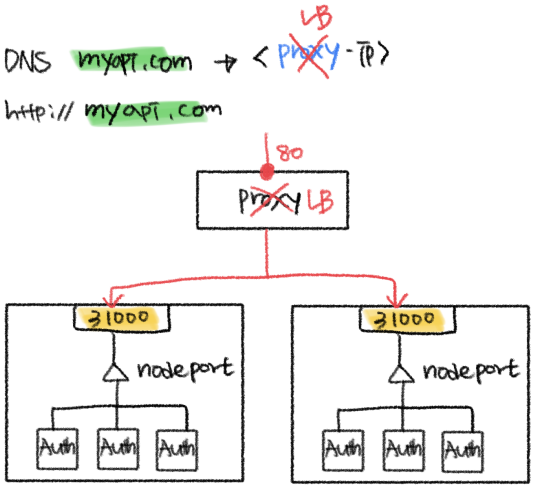
이제 사용자는 `http://myapi.com:31000` 으로 접근할 수 있습니다. 하지만 DNS에는 포트까진 등록할 수 없으므로 사용자는 여전히 포트 번호를 알고 있어야 합니다. `http://myapi.com` 만 치고도 접근할 수 있도록 하고 싶습니다.



프록시를 도입합니다. 프록시 서버의 80포트로 오는 요청을 `<node-ip>:31000` 으로 리버스 프록싱하도록 등록해줍니다. 이제 DNS에는 `<node-ip>` 가 아닌 `<proxy-ip>` 를 등록합니다. 사용자는 `http://myapi.com` 으로 Auth api에 접근할 수 있게 되었습니다.

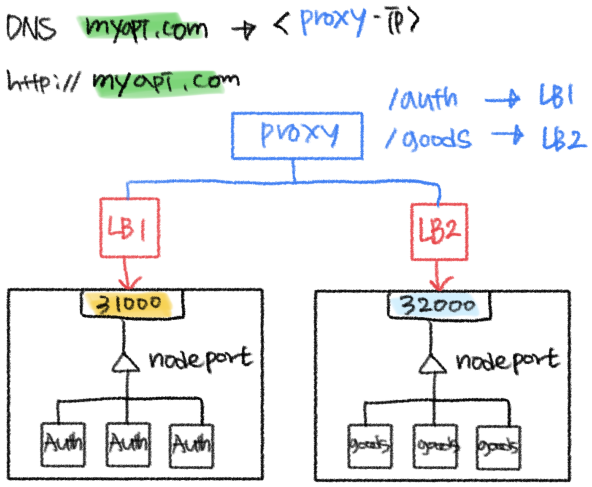
하지만, 여기서 클러스터가 2개 이상이 되면 문제가 생깁니다. 만약 노드가 2개가 된다면 노드의 IP도 두개가 될 것 입니다. Nodeport 서비스는 `노드의 IP:Nodeport` 로의 요청만 처리할 수 있습니다.

즉, `<node-ip 1번>:31000` `<node-ip 2번>:31000` 으로 앞에서 로드밸런싱 해줄 무언가 - 로드밸런서가 필요합니다.



노드 앞에 로드밸런서를 둡시다. 이제 DNS에는 `<proxy-ip>` 가 아닌 `<loadbalancer-ip>` 를 등록합니다.

하지만, 여기서 deployment가 여러 종류가 되면 문제가 생깁니다. 만약 Goods api를 추가한다고 생각해봅시다. 앞으로 `myapi.com/auth` 로 오는 요청은 Auth api로, `myapi.com/goods` 로 오는 요청은 Goods api로 보내고 싶습니다. 로드밸런서는 이걸 처리하지 못합니다.



로드밸런서는 (보통) L4 레벨이라 각 서비스마다 로드밸런서를 달아줍니다. 이러면 구성이 너무 복잡해졌습니다. 그래서 등장한것이 바로 Ingress입니다.

Ingress는 간단히 말해서 쿠버네티스 빌트인 L7 레벨 로드밸런서라고 생각하면 됩니다.

> 💡 Ingress는 사용자가 URL을 통해 어플리케이션에 접근할 수 있게 합니다.

URL 경로를 기반으로 클러스터 안의 서비스에 라우팅할 수 있도록 합니다.

단, 인그레스를 도입하더라도 여전히 클러스터 외부에서 (인그레스에) 접속할 수 있도록 인그레스와 연결된 서비스가 필요합니다.

참고 자료

https://seongjin.me/kubernetes-service-types/

https://yoonchang.tistory.com/49

https://medium.com/google-cloud/kubernetes-nodeport-vs-loadbalancer-vs-ingress-when-should-i-use-what-922f010849e0