

6장 서비스 API 카테고리

2025.11.28 구피(김은빈)

목차

1. 서비스란?
2. 서비스 디스커버리
3. 서비스 타입
4. 서비스 설정 속성들
 - Session Affinity
 - externalTrafficPolicy
 - Topology aware Routing
5. 인그레스란?
6. 게이트웨이 API란?

서비스란?

“서비스”에 요청하면,
엔드포인트들에 트래픽을 균등분배해줌
(L4 로드밸런싱)

서비스에 요청: IP가 대상
서비스의 IP일 수도 아닐 수도 있음

엔드포인트들: Endpoints 리소스
SVC에 라벨 Selector 지정하면: 매칭 Pod의 IP 목록으로 자동생성
SVC에 Selector 지정하지 않으면: IP 목록 지정한 리소스 생성해야함

서비스 위치 찾는 법(디스커버리)

- **Pod 환경변수로 주소 얻기**
 - 같은 네임스페이스에 있는 서비스의 IP, port들을 환경변수로 가져올 수 있음
- **A레코드 도메인:** {service명}.{namespace명}.svc.cluster.local
ex. _http-port._tcp.default.svc.cluster.local
- **SRV레코드 도메인:** _{port명}._{protocol}.{service명}.{namespace명}.svc.cluster.local
ex. _http-port._tcp.default.svc.cluster.local
 - SRV 레코드를 조회하면 **포트 번호와 타겟 호스트**를 함께 얻을 수 있어서,
클라이언트가 서비스의 포트를 하드코딩하지 않아도 됨

서비스 타입

type	서버스 도메인 resolve 결과	외부 접근 (external IP)
clusterIP	클러스터 내부 IP	외부 IP (노드 IP) (선택, 있으면 내부 통신)
NodePort	클러스터 내부 IP	모든 노드 IP
LoadBalancer	클러스터 내부 IP	외부 LB → 모든 노드 IP (NodePort)
Name	내부 IP 할당 X, Pod IP 변환	X
ExternalName	내부 IP 할당 X, 특정 도메인의 CNAME	X

서비스 타입

Q. 이 두 유형의 서비스는
Pod 환경변수로 어떤 값이 주입될까?

A. 환경변수로 주입되지 않는다. IP가 없으므로.

<https://stackoverflow.com/questions/69611827/kubernetes-is-there-any-way-to-get-headless-service-endpoint-info-in-container>

<https://github.com/kubernetes/kubernetes/issues/60535>

Name	내부 IP 할당 X, Pod IP 반환	X
ExternalName	내부 IP 할당 X, 특정 도메인의 CNAME	X

```
/ # env | sort
CLUSTERIP_SVC_PORT=tcp://192.168.194.151:80
CLUSTERIP_SVC_PORT_80_TCP=tcp://192.168.194.151:80
CLUSTERIP_SVC_PORT_80_TCP_ADDR=192.168.194.151
CLUSTERIP_SVC_PORT_80_TCP_PORT=80
CLUSTERIP_SVC_PORT_80_TCP_PROTO=tcp
CLUSTERIP_SVC_SERVICE_HOST=192.168.194.151
CLUSTERIP_SVC_SERVICE_PORT=80
HOME=/root
HOSTNAME=nodepool-runner-alpine-411
KUBERNETES_PORT=tcp://192.168.194.129:443
KUBERNETES_PORT_443_TCP=tcp://192.168.194.129:443
KUBERNETES_PORT_443_TCP_ADDR=192.168.194.129
KUBERNETES_PORT_443_TCP_PORT=443
KUBERNETES_PORT_443_TCP_PROTO=tcp
KUBERNETES_SERVICE_HOST=192.168.194.129
KUBERNETES_SERVICE_PORT=443
KUBERNETES_SERVICE_PORT_HTTPS=443
NODEPORT_SVC_PORT=tcp://192.168.194.242:80
NODEPORT_SVC_PORT_80_TCP=tcp://192.168.194.242:80
NODEPORT_SVC_PORT_80_TCP_ADDR=192.168.194.242
NODEPORT_SVC_PORT_80_TCP_PORT=80
NODEPORT_SVC_PORT_80_TCP_PROTO=tcp
NODEPORT_SVC_SERVICE_HOST=192.168.194.242
NODEPORT_SVC_SERVICE_PORT=80
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin
PWD=/
SHLVL=2
TERM=xterm
```

서비스 타입

Q2. kubernetes라는 이름의 서비스는
왜 test namespace에 없는데도 환경변수에 있는지?

```
~/personal/k8stest > kg svc -n test
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
clusterip-svc	ClusterIP	192.168.194.151	<none>	80/TCP	11m
externalname-svc	ExternalName	<none>	example.com	<none>	11m
headless-svc	ClusterIP	None	<none>	80/TCP	11m
nodeport-svc	NodePort	192.168.194.242	<none>	80:30080/TCP	11m

A. kube-apiserver를 가리키는 서비스
어느 네임스페이스에 있든 API 서버와 통신할 수 있어야 하므로,
kubelet이 예외적으로 모든 Pod에 주입함

<https://kubernetes.io/docs/concepts/services-networking/service/#discovering-services>

```
/ # env | sort
CLUSTERIP_SVC_PORT=tcp://192.168.194.151:80
CLUSTERIP_SVC_PORT_80_TCP=tcp://192.168.194.151:80
CLUSTERIP_SVC_PORT_80_TCP_ADDR=192.168.194.151
CLUSTERIP_SVC_PORT_80_TCP_PORT=80
CLUSTERIP_SVC_PORT_80_TCP_PROTO=tcp
CLUSTERIP_SVC_SERVICE_HOST=192.168.194.151
CLUSTERIP_SVC_SERVICE_PORT=80
HOME=/root
HOSTNAME=nodepool-runner-alpine-411
KUBERNETES_PORT=tcp://192.168.194.129:443
KUBERNETES_PORT_443_TCP=tcp://192.168.194.129:443
KUBERNETES_PORT_443_TCP_ADDR=192.168.194.129
KUBERNETES_PORT_443_TCP_PORT=443
KUBERNETES_PORT_443_TCP_PROTO=tcp
KUBERNETES_SERVICE_HOST=192.168.194.129
KUBERNETES_SERVICE_PORT=443
KUBERNETES_SERVICE_PORT_HTTPS=443
NODEPORT_SVC_PORT=tcp://192.168.194.242:80
NODEPORT_SVC_PORT_80_TCP=tcp://192.168.194.242:80
NODEPORT_SVC_PORT_80_TCP_ADDR=192.168.194.242
NODEPORT_SVC_PORT_80_TCP_PORT=80
NODEPORT_SVC_PORT_80_TCP_PROTO=tcp
NODEPORT_SVC_SERVICE_HOST=192.168.194.242
NODEPORT_SVC_SERVICE_PORT=80
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin
PWD=/
SHLV=2
TERM=xterm
```

Session Affinity

```
apiVersion: v1
kind: Service
▼ metadata:
  name: my-service
▼ spec:
  selector:
    app: my-app
  ports:
    - port: 80
      targetPort: 8080
  sessionAffinity: ClientIP
  sessionAffinityConfig:
    clientIP:
      timeoutSeconds: 10800
```

이전에 요청이 전달된 노드로 전달

주의할 점: NodePort에선

externalTrafficPolicy: Cluster이면 NodeIP에 따라 동작

구현은 kube-proxy 동작에 따라 다름

Iptables

- recent 모듈로 pod별로 방문자들 테이블을 저장하고 체크

<https://blog.rlaisqls.com/til/devops/kubernetes/개념/session-affinity/#동작-원리>

<https://ipset.netfilter.org/iptables-extensions.man.html>

IPVS

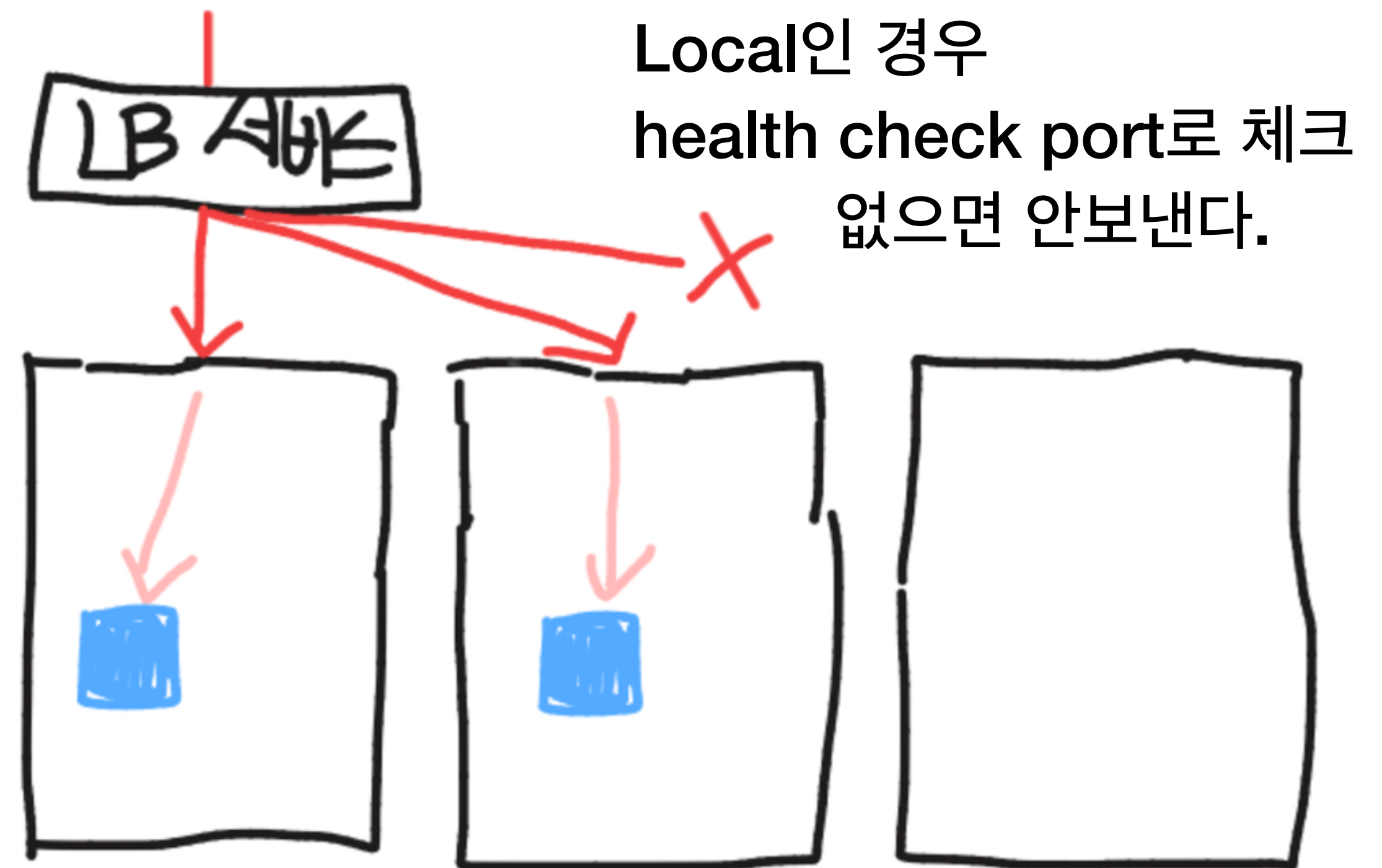
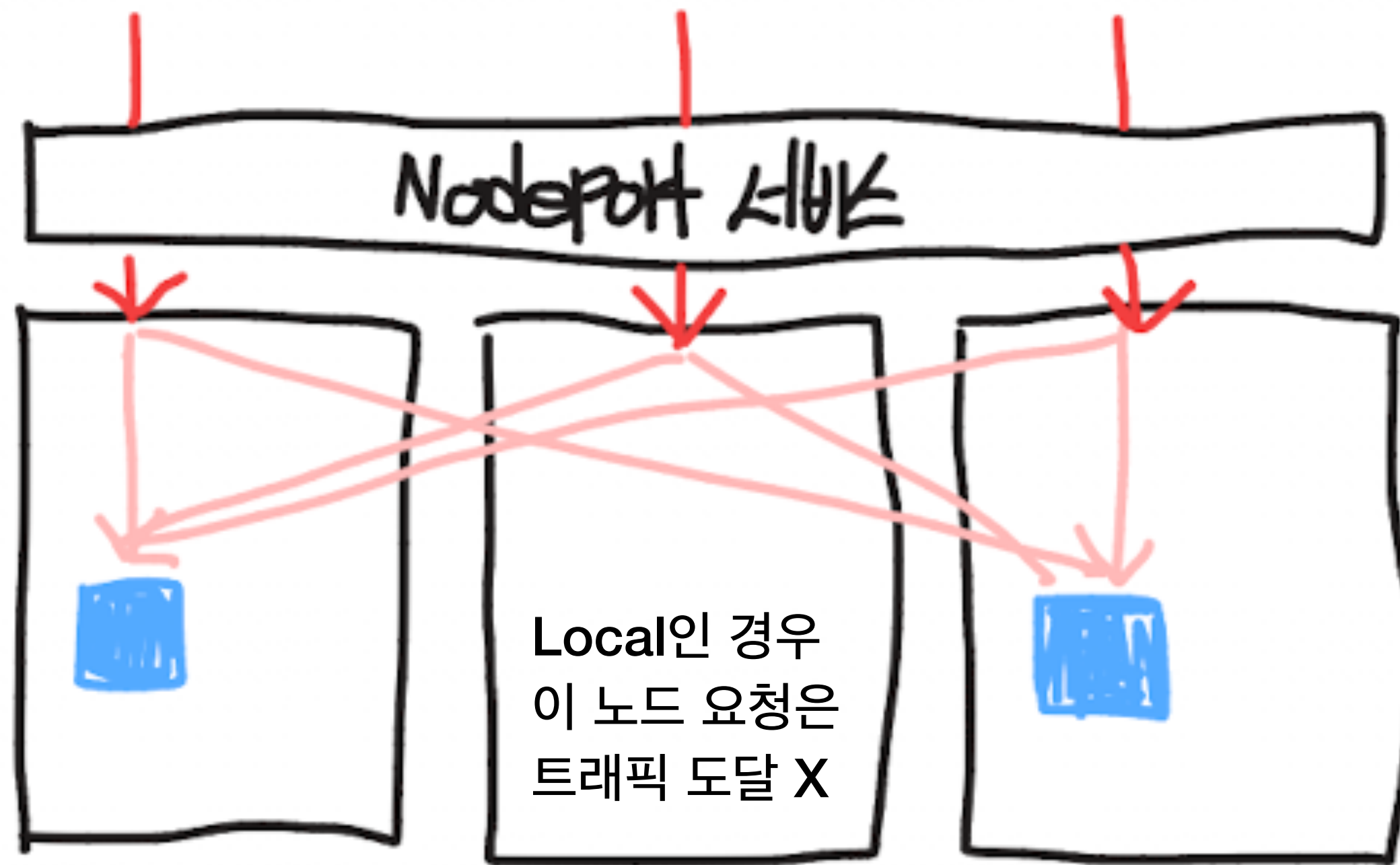
- 동일한 역할의 Persistence 기능 사용

<https://kubernetes.io/blog/2018/07/09/ipvs-based-in-cluster-load-balancing-deep-dive/>

externalTrafficPolicy

Cluster (기본값): 다른 노드에 있는 Pod를 포함해 로드밸런싱, 분홍색 선

Local: 노드간 로드밸런싱하지 않음



Cluster인 경우 Source IP가 손실됨

- Node A가 패킷을 받아서 Node B의 Pod로 전달. Pod는 원본 Client IP로 직접 응답하려 함
- 하지만 Client는 Node A를 통해 왔으므로, Node B에서 직접 응답하면 비대칭 라우팅 발생
- 이를 방지하기 위해 SNAT(Source NAT)을 수행 → Source IP를 Node A의 IP로 변경

Topology Aware Routing

kube-proxy가 가능한 한 같은 Zone 내의 엔드포인트로 트래픽을 라우팅한다.
EndpointSlice 리소스에 설정 가능, Endpoints 리소스에는 설정할 수 없음

```
apiVersion: v1
kind: Service
▼ metadata:
  name: my-service
  annotations:
    service.kubernetes.io/topology-mode: Auto
▼ spec:
  selector:
    app: my-app
  ports:
▼    - port: 80
      targetPort: 8080
```

```
apiVersion: discovery.k8s.io/v1
kind: EndpointSlice
▼ metadata:
  name: my-service-abc
  labels:
    kubernetes.io/service-name: my-service
addressType: IPv4
▼ endpoints:
▼   - addresses:
      - "10.0.1.5"
      conditions:
        ready: true
        zone: zone-a
      hints:
        forZones:
          - name: zone-a
▼   - addresses:
      - "10.0.2.10"
      conditions:
        ready: true
        zone: zone-b
      hints:
        forZones:
          - name: zone-b
```

Endpoints vs EndpointSlice

Endpoints의 한계

- 단일 객체에 모든 엔드포인트 저장 (etcd 1MB 제한)
- Pod 하나가 변경되어도 전체 Endpoints 객체 전송
- 대규모 클러스터에서 성능 저하

EndpointSlice: 100개씩 나눠 저장

Kubernetes 1.21+는 EndpointSlice가 기본적으로 사용됨

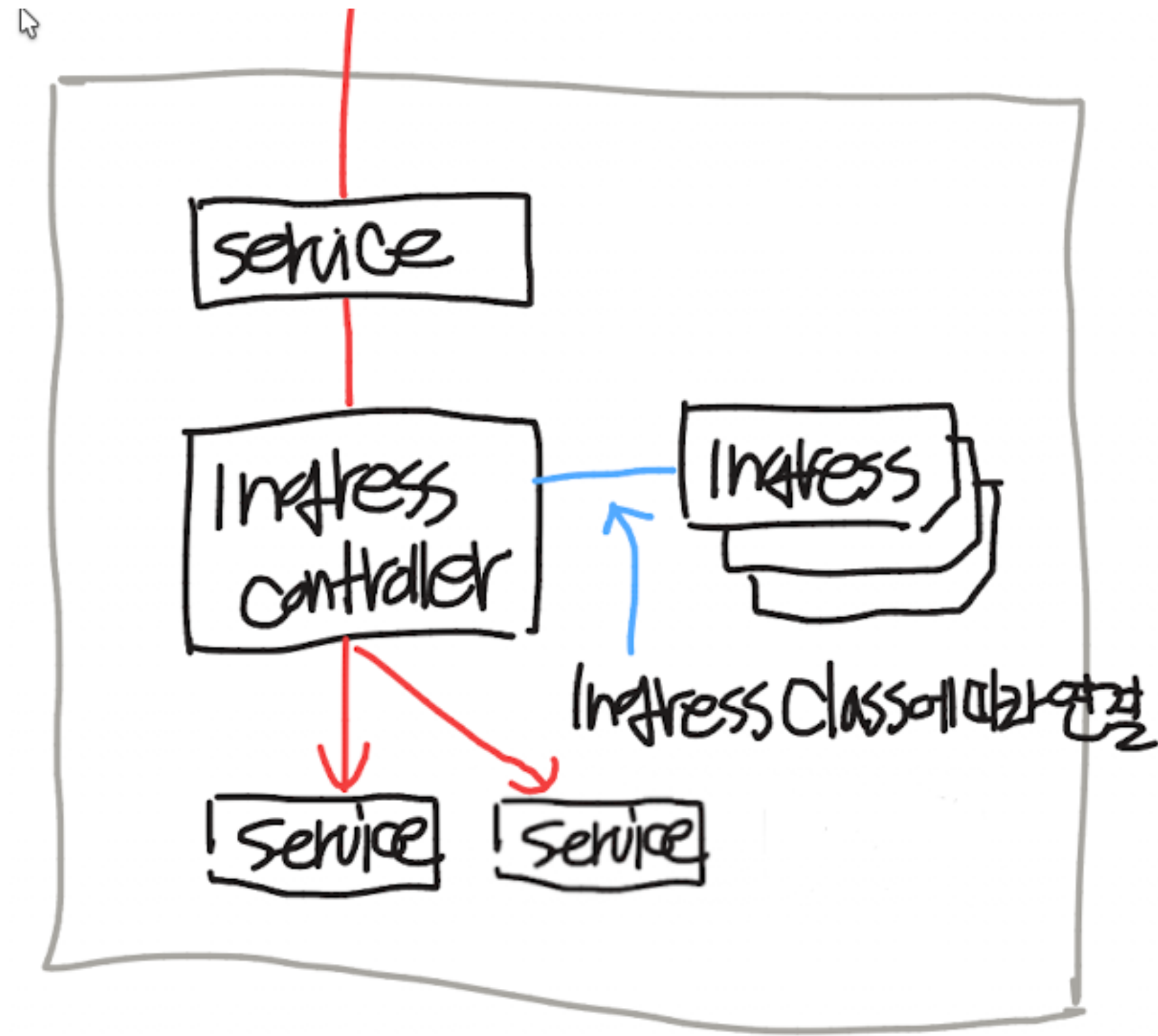
Topology 등 기능, EndpointSlice에만 지원되는 경우가 많음

```
apiVersion: v1
kind: Endpoints
metadata:
  name: my-service
subsets:
  - addresses:
      - ip: 10.1.1.1
      - ip: 10.1.1.2
    ports:
      - port: 80
```

```
apiVersion: discovery.k8s.io/v1
kind: EndpointSlice
metadata:
  name: my-service-abc12
  labels:
    kubernetes.io/service-name: my-service
addressType: IPv4
ports:
  - name: http
    port: 80
    protocol: TCP
endpoints:
  - addresses:
      - "10.1.1.1"
    conditions:
      ready: true
      serving: true
      terminating: false
      nodeName: node-1
      zone: ap-northeast-2a
    hints:
      forZones:
        - name: ap-northeast-2a
```

인그레스란?

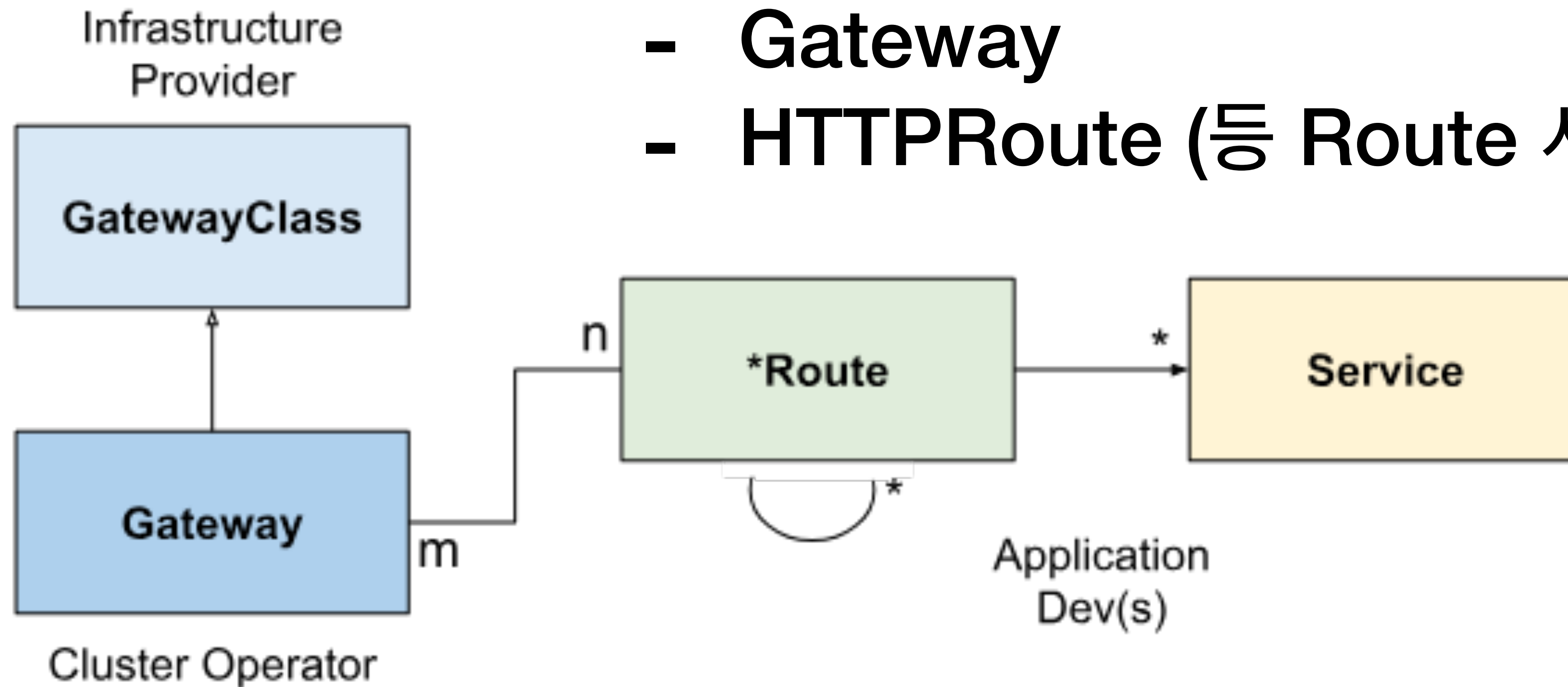
L7 (HTTP/HTTPS) 트래픽을 라우팅



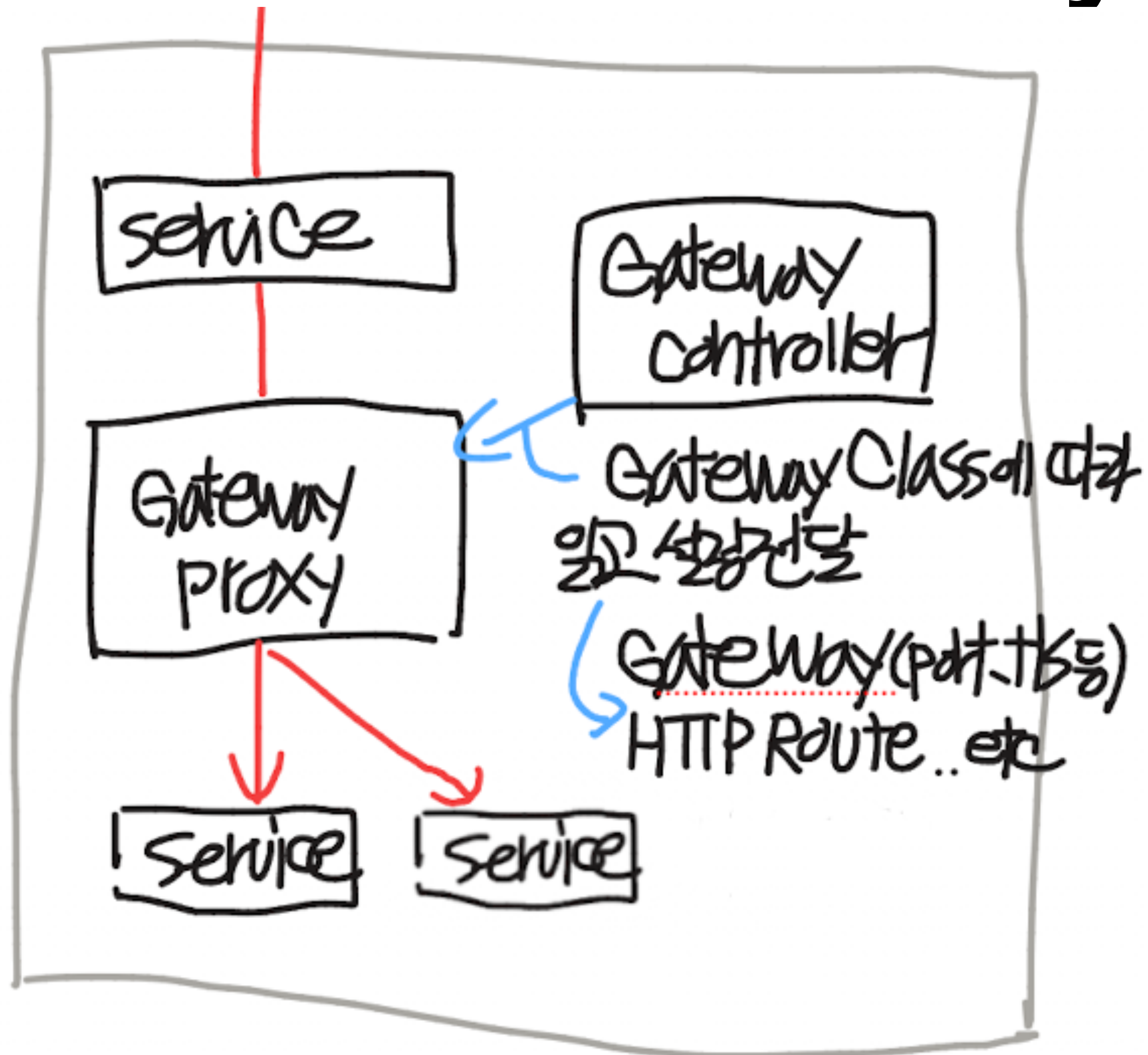
- IngressController (라우팅 수행)
- IngressClass
- Ingress (규칙 정의)

Gateway API란?

- Gateway Controller
- GatewayClass
- 기존 Ingress 역할
 - Gateway
 - HTTPRoute (등 Route 시리즈)



Gateway API란?



- Ingress와 기본적으로 같은데, 아키텍처를 이렇게 하는 경우가 종종 있다

정리

Service
Ingress
Gateway API

감사합니다.