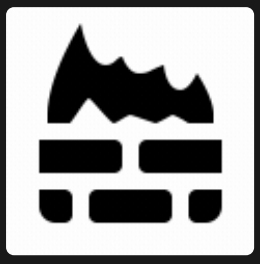


Network Policies(Firewall)

1. NetworkPolicy 이해

NetworkPolicy

- Kubernetes가 지원하는 Pod 통신 접근 제한. 일종의 방화벽으로 Pod로 트래픽이 들어오고(Inbound), 나가는(Outbound)것을 설정하는 정책
- https://kubernetes.io/ko/docs/concepts/services-networking/network-policies/

- NetworkPolicy : IP 주소 또는 포트 수준(OSI 계층 3 또는 4)에서 트래픽 흐름을 제어
  - 파드가 통신할 수 있는 엔티티는 다음 3개의 식별자 조합을 통해 식별된다.
    1. 허용되는 다른 파드(예외: 파드는 자신에 대한 접근을 차단할 수 없음)
    2. 허용되는 네임스페이스
    3. IP 블록(예외: 파드 또는 노드의 IP 주소와 관계없이 파드가 실행 중인 노드와의 트래픽은 항상 허용됨)
  - Ingress 트래픽 컨트롤 : Inbound 정책. 즉, **들어오는 트래픽(기준: 특정 Pod:port)**을 허용할 것 인지를 정의
    - 기본적으로, 파드는 Ingress에 대해 비격리되어 있다. 즉, 모든 인바운드 연결이 허용된다.
    - 해당 파드에 적용되면서 `policyTypes` 에 "Ingress"가 있는 NetworkPolicy가 존재하는 경우, 파드가 인그레스에 대해 격리된다.
  - Egress 트래픽 컨트롤 : Outbound 정책. 즉, **트래픽이 나갈 수(기준: 특정 Pod:port)** 있는 허용 범위 정의
    - 기본적으로, 파드는 Egress에 대해 비격리되어 있다. 즉, 모든 아웃바운드 연결이 허용된다.
    - 해당 파드에 적용되면서 `policyTypes` 에 "Egress"가 있는 NetworkPolicy가 존재하는 경우, 파드가 Egress에 대해 격리된다.

Ingress 트래픽 컨트롤 정의

- 어디서 들어오는 트래픽을 허용 할 것 인지를 정의하는 방법은 여러 가지가 있다.
  - **ipBlock** : CIDR IP 대역으로, 특정 IP 대역에서만 **트래픽이 들어오도록 지정**할 수 있다.
  - **podSelector** : label을 이용하여, 기준이 되는 Pod가 있는 네임스페이의 **특정 label을 가지고 있는 Pod들에서 들어오는 트래픽만 받을 수 있다.**
  - **namespaceSelector** : 특정 namespace로 부터 들어오는 트래픽만을 받는 기능이다. 운영 로깅 서버의 경우에는 운영 환경 namespace에서만 들어오는 트래픽을 받거나, 특정 서비스 컴포넌트의 namespace에서의 트래픽만 들어오게 컨트롤이 가능하다. 내부적으로 새로운 서비스 컴포넌트를 오픈했을 때, 베타 서비스를 위해서 특정 서비스나 팀에게만 서비스를 오픈하고자 할 때 유용하게 사용할 수 있다.
  - **Protocol & Port** : 특정 Protocol 또는 Port로 설정된 트래픽만 허용되는 포트를 정의할 수 있다.

- 예제(https://kubernetes.io/docs/concepts/services-networking/network-policies/#networkpolicy-resource)

```
apiVersion: networking.k8s.io/v1 kind: NetworkPolicy metadata:
name: test-network-policy namespace: default spec: podSelector:
matchLabels: ## default 네임스페이스에서 동작중인 role=db 레이블을 가진 파드들에 Ingress(inbound),Egress(outbound)허용/금지 role: db
policyTypes: - Ingress - Egress ingress: - from: - ipBlock:
cidr: 172.17.0.0/16 ## 이 네트워크로만 인바운드 허용 or except: -
172.17.1.0/24 - namespaceSelector: matchLabels: ## 레이블을 가지는 네임스페이스에 있는 모든 pod들은 허용 or project: myproject - podSelector:
matchLabels: ## 동일네임스페이스(default)에 있는 role: frontend 레이블을 가지는 포드만 인바운드 허용 or role: frontend ports: - protocol: TCP
port: 6379 egress: - to: - ipBlock: cidr: 10.0.0.0/24 ## 이 네트워크로만 아웃바운드 허용 ports: - protocol: TCP ## 이 포트로만 아웃바운드 허용
port: 5978
```

- 예: 특정 namespace에 있는 특정 파드들만 허용

```
cat web-allow-prod.yaml kind: NetworkPolicy apiVersion: networking.k8s.io/v1 metadata: name: web-allow-prod spec: podSelector: matchLabels: app: web ## default 네임스페이스에 있는 app=web 레이블을 가진 POD가 기준 policyTypes: - Ingress ingress: - from: # purpose: production - namespaceSelector: matchLabels: purpose: production ## label을 가진 네임스페이스에 있는 모든 Pod는 허용 or - podSelector: matchLabels: ## default 네임스페이스에 있는 role=frontend 레이블을 가지는 pod만 인바운드 허용 role: frontend
```

- 예: 특정 namespace에 있는 특정 파드들만 허용

```
cat web-allow-prod.yaml kind: NetworkPolicy apiVersion: networking.k8s.io/v1 metadata: name: web-allow-prod spec: podSelector: matchLabels: app: web ## default 네임스페이스에 있는 app=web 레이블을 가진 POD가 기준 policyTypes: - Ingress ingress: - from: # purpose: production - namespaceSelector: matchLabels: purpose: production ## label을 가진 네임스페이스에 있는 모든 Pod는 허용 and podSelector: matchLabels: ## default 네임스페이스에 있는 role=frontend 레이블을 가지는 pod만 인바운드 허용 role: frontend
```

- 예: 특정 namespace와 all namespace의 파드 허용

```
cat web-allow-prod.yaml kind: NetworkPolicy apiVersion: networking.k8s.io/v1 metadata: name: web-allow-prod spec: podSelector: matchLabels: app: web ## default 네임스페이스에 있는 app=web 레이블을 가진 POD가 기준 policyTypes: - Ingress ingress: - from: # purpose: production - namespaceSelector: matchLabels: purpose: production ## purpose: production이라는 label을 가진 네임스페이스에 있는 모든 Pod는 허용 or - podSelector: matchLabels: ## default 네임스페이스에 있는 role=frontend 레이블을 가지는 pod만 인바운드 허용 role: frontend
```

- 모든 Inbound 트래픽 거부

```
# default namespace에 있는 모든 pod 에는 모두 접근 거부 apiVersion: networking.k8s.io/v1 kind: NetworkPolicy metadata: name: default-deny-ingress spec: podSelector: {} policyTypes: - Ingress
```

- 모든 Inbound 트래픽 허용

```
# default namespace에 있는 모든 pod 에는 모두 접근 허용 apiVersion: networking.k8s.io/v1 kind: NetworkPolicy metadata: name: allow-all-ingress spec: podSelector: {} policyTypes: - Ingress ingress: - {}
```

- 예: devops namespace에 모든 Pod(80) 에 대해서 특정 네임스페이스에 pod들이 허용

```
cat web-allow-prod.yaml kind: NetworkPolicy apiVersion: networking.k8s.io/v1 metadata: name: web-allow-prod namespace: devops spec: podSelector: {} ## devops 네임스페이스에 있는 모든 POD가 기준 policyTypes: - Ingress ingress: - from: - namespaceSelector: matchLabels: purpose: production ## label을 가진 네임스페이스에 있는 모든 Pod는 허용 - podSelector: matchLabels: ## devops 네임스페이스에 있는 role=frontend 레이블을 가지는 pod만 인바운드 허용 role: frontend ports: - protocol: TCP port: 80
```

Network Policy Egress 설정

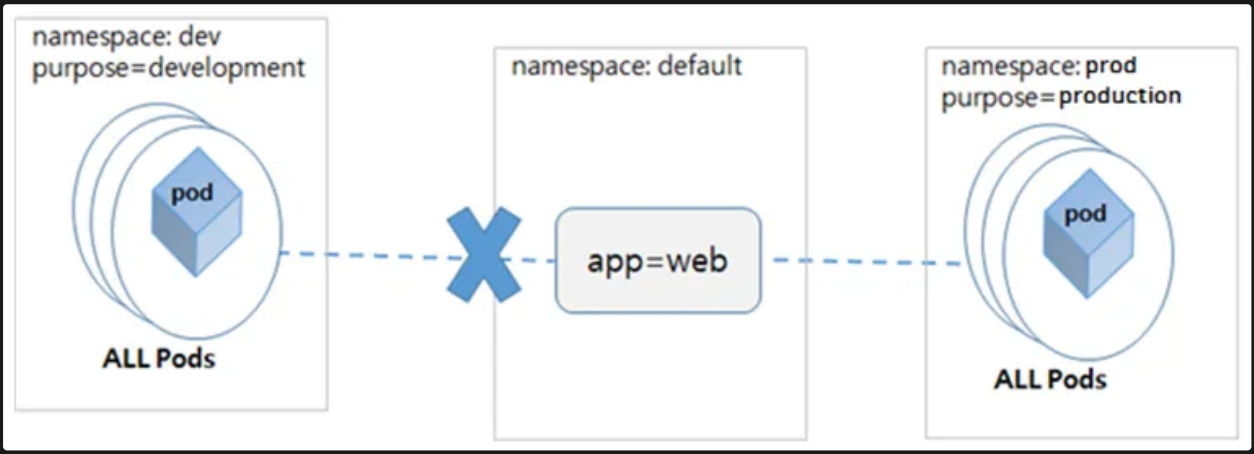
- Network Outbound. 특정 Pod에서 밖으로 나가는 트래픽에 대한 방화벽 설정으로 어떠한 트래픽이 Pod로 부터 나가는 것을 허용할 것인지 설정함.
- ipBlock
  - 특정 IP 대역만 트래픽이 Outbound 될 수 있도록 허용할 수 있음.
  - ipBlock에 설정된 대역만 허용이고 그 외는 모두 차단.
  - IP대역 설정은 CIDR IP 대역으로 설정되야 함.
- Protocol & Port
  - 특정 Protocol 또는 port로 설정된 트래픽만 나가는 것을 허용할 수 있음.

2. 실습

namespace ACL

- 특정 네임 스페이스에서 Ingress 허용

  `purpose=production` 레이블을 사용하는 namespace에 있는 모든 Pod 들만 app=web 레이블을 사용하는 pod에 80포트 접근 허용




```
kubectl create namespace dev kubectl create namespace prod kubectl
```