Ingress Resource

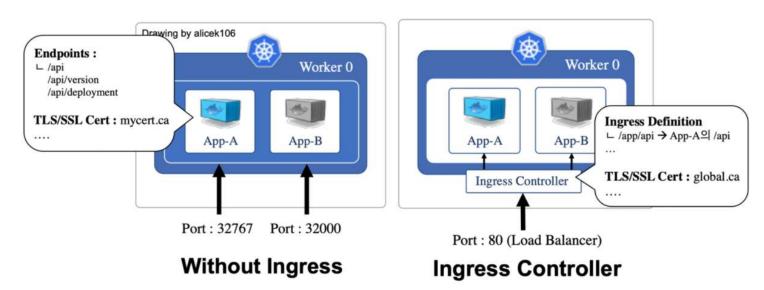
2021. 07. 18

k8s study w8

Yunjay Hong

Ingress

- Service Resource로 k8s pods의 네트워킹을 설정 할 수 있지만, 세부 적인 네트워크 처리 로직을 구현하기에는 한계가 있음
- NodePort와 같은 Resource로는 Layer4 (Transport Layer. e.g. TCP, UDP)에서의 요청을 처리한다



Ingress (Cont'd)

- Ingress Resource는 Layer7(Application Layer)의 요청을 처리
- 로드 밸런싱, TLS/SSL 인증서 처리, HTTP 경로 라우팅
- NodePort등으로 구현이 가능하지만, 각 application 별로 구현하게 되면 서비스 운영 측면에서 복잡성이 발생
- Ingress Resource에서 통합해 관리함으로서 라우팅 규칙 등을 쉽게 관리한다.

Ingress Object

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
    annotations:
    kubernetes.io/ingress.class: nginx
    name: mynginx
spec:
    rules:
    - host: 10.0.1.5.sslip.io
    http:
    paths: 3
    - path: /
    backend:
    serviceName: mynginx
    servicePort: 80

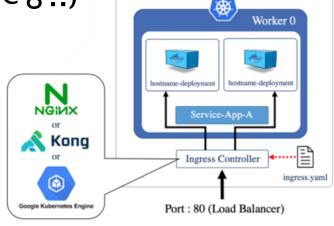
5
```

- 이 Ingress object는 10.0.1.5.sslip.io로 접속한 트래픽 중 '/' 의 path를 가진 트래픽을 mynginx 서비스의 80 port로 라우팅을 하겠다는 규칙을 정의한다
- Annotation[1]: Ingress를 처리하는 Ingress Controller에게 주는 정보. Metadata와 다 르게 Label Filtering이 불가능하다.

Ingress Controller

- Ingress Object는 Ingress(들어오는 Traffic)에 대한 규칙을 정의할 뿐이고, 실제 그 규칙을 실행하지 않음.
- Ingress Controller는 규칙을 실행하는 웹 서버.

• 다른 Controller와는 다르게 명시적으로 설치해주어야 하고, 종류가 다양하다. (NGINX, Kong, Google k8s engine등..)



Domain/Path based Routing

- subdomain domain/path
- apache.10.0.1.5.sslip.io로 접속한 트래픽은 apache:80로 라우팅 되고, apache라는 이름의 서비스에 라우팅 된다.
- 10.0.1.5.sslip.io/apache로 접속한 트래픽은 nginx ingress controller에 의해 apache 서비스의 80번 포트로 라우팅된다.
- 이때 *nginx,ingress,kubernetes,io/rewrite target*은 프록시되는 서버에 전달되는 path를 '/' 로 재정의하게 도와준다.

```
1 apiVersion: extensions/v1beta1
2 kind: Ingress
3 metadata:
4 annotations:
5 kubernetes.io/ingress.class: nginx
6 name: apache-domain
7 spec:
8 rules:
9 - host: apache.10.0.1.5.sslip.io
10 http:
11 paths:
12 - path: /
13 backend:
14 serviceName: apache
15 servicePort: 80
```

```
1 apiVersion: extensions/v1beta1
2 kind: Ingress
3 metadata:
4 annotations:
5 kubernetes.io/ingress.class: nginx
6 nginx.ingress.kubernetes.io/rewrite-target: /
7 name: apache-path
8 spec:
9 rules:
10 host: 10.0.1.5.sslip.io
11 http:
12 paths:
13 - path: /apache
14 backend:
15 serviceName: apache
16 servicePort: 80
```

Basic Authentication

- nginx에서 기본적으로 제공하는 auth 기능으로, 똑같이 nginx ingress controller에서도 사용가능함.
- HTTP Authentication 방법 중 하나로, 인코딩 된 ID/PW를 header로 받아 검증하는 방식.

• Secret object를 생성하고, 이 secret을 ingress의 annotation에 지

정해 authentication을 알려준다.

```
kind: Ingress
metadata:
annotations:
kubernetes.io/ingress.class: nginx
nginx.ingress.kubernetes.io/auth-type: basic
nginx.ingress.kubernetes.io/auth-secret: basic-auth
nginx.ingress.kubernetes.io/auth-realm: 'Authentication Required - foo'
name: apache-auth
spec:
rules:
- host: apache-auth.10.0.1.5.sslip.io
http:
paths:
- path: /
backend:
serviceName: apache
servicePort: 80
```

TLS(Transport Layer Security)

- SSL(Secure Socket Layer)의 차세대 버전으로, 일반적으로 생각하는 SSL이라고 보면 된다.
- https 프로토콜, 이메일, 유즈넷 등의 연결에 사용되고 있으며, 암호화 되는 인증서를 기반으로 트래픽의 보안을 유지하는 프로토콜이다.
- Self-signed Certificate은 공인된 인증 기관을 통해 인증받은 인증서 가 아닌 일반적인 인증서이며, openssl을 이용해 생성할 수 있다.

Certificate 생성

- openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout tls.key -out tls.crt -subj "/CN=apache-tls.10.0.5.1.sslip.io"
- req: 생성을 위한 openss! 유틸리티
- -x509: x509 structure로 key를 생성하는 옵션
- -newkey rsa:2048 : 2048bit rsa 개인키를 생성하는 옵션
- -keyout tls.key : 개인키 생성을 위한 옵션
- -out tls.crt : certificate 저장을 위한 옵션

Secret & Ingress TLS

```
master@master-jay:~/kubernetes-start-up-study/09_k8s_ingress$ cat << EOF | kubectl apply -f -
apiVersion: v1
kind: Secret
metadata:
   name: my-tls-certs
   namespace: default
data:
   tls.crt: $(cat tls.crt | base64 | tr -d '\n')
   tls.key: $(cat tls.key | base64 | tr -d '\n')
type: kubernetes.io/tls
EOF
secret/my-tls-certs created</pre>
```

master@master-jay:~/kubernetes-start-up-study/09_k8s_ingress\$ cat tls.crt | base64 | tr -d '\n'
LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSURMekNDQWhlZ0F3SUJBZ0lVU3JvSGRNckJXQTEVMkFLRklyTWozai9oZmdVd0RRWUpLb1pJaHZjTkFRRUwKQlFBd0p6RWxNQ01HQ
TFVRUF3d2NZWEJoNTJobExYUnNjeTR4TUM0deXqRXVOUZV6YZJ4cGNDNXBiekFlRncweQpNVEEZTVRnd05qSTJNVEphRncweU1qQTNNVCd3TmpJMk1USnFNQ2N4SlRBakJnTlZCQU1NSE
dGd1lXTm9aUZEwCmJJTXVNVEF1TUM0eExqVXVjM05zYVhBdWFXOHdnZ0VpTUEwR0NTcUdTSW1zRFFFQkFRVUPBNElCRHMdB2dnRUSKQW9JQkFRQzhxbTfinCtOTUpwdnVoL280eG1HWkR
kdllhamYyU2xmazArdWVld1dtTVArWmtoenBYd1dmU2FucwpoUFZDLzhkRGFGTkJkNHIyMG9yeUs4dXJvY1VTU0cxNWQ2TFJWejQrYmNQb0Y4RGljZG1CdDRSdVJQUmw3RWNOcnV4WmFv
RHLKWmdLZVl3M2RZNWszZVVMZVlZYmJnK21GeU1rOXFlaTVhQwZTN1hyeE1UVkJwU2ZxL2dWdjBEcW0K0FRTQn1BvEVBUnozTDczaitxNENMbXFWpjB6dEpvMLZHR2M5ektGT0k0V3yV0
DFRQUg1czFmR2d1Sw41NWhvMgppc3BtazMrNld3MVMrVUZPbDhYSEw0eGt5ckZkcU10ZzBjQm9YekZ1d2NTT2FqQXBHd0tUZzZvbDVDeU9HcwZNCnJVbzk5bjkyeUZBbjVaZlgvOVpoK3
BFa3l3NlRBZ01CQUFHalV6QlJNQjBHQTFVZERnUVdCQlRrUGZhRUIvSEEKM25LRmdLQ1FhSk9nYXozbjBqQWZCZ05WSFNNRUdEQVdnQlRrUGZhRUIvSEEzbktGZ0tDUWFKT2dhejNuMGp
BUApCZ05WSFJNQkFm0EVCVEFEQVFIL01BMEdDU3FHU0liM0RRRUJDd1VBQTRJQkFRQmhPe0lPMXV6TDZOSFI1azhCcmhieEthV1FZeDQyemRVRWYXUGpjUUlXNnJsb2ZscmhzN2R5SEsr
ZUtBBUZIdnJZYWxVM3ZxKzlaMTVmZlNtagCkEXASJUUtaxij9xTEx0ND10UlBIb2NKLysobzBybG0ybkN4SVFXWUHRQ1kxdkN0ZG9USNniv70ZJREQ1U2kv0WIXNwpJ3elgrU0dHyTqUURJZ
Vp0VU8vVGI1eWZ4WFF1cmt1MkhJdDvSM1pYcFhPV3d2YmY4cU9V0Gpx0HJWUXM1TVg0CjJMQUh5UzVRCSthTVhoKzBTUGLtK2Q5ajh0bUJ3a1R2WEhqRFFS0XR4TE0wRlB4RFdkY3Bway
912lhNNzFhbosK51IvdmN5aw9PactvUmt4d3V6NERzVjNmKzRVZ0xDVz0rVDISUZR1aG5SczFCVGxCay9swXBPMYY2NkZXXAVUwpqZVBaciotLS0tLVStEENFULRJRklDQVRFLS0tLS0

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
   name: apache-tls
spec:
  tls:
   - hosts:
         apache-tls.10.0.1.5.sslip.io
     secretName: my-tls-certs
   host: apache-tls.10.0.1.5.sslip.io
     http:
       paths:
       - path: /
         backend:
           serviceName: apache
           servicePort: 80
```