안전한 backend 통신을 위한

TLS 인증서 통합 방안

CK 1-1 팀 윤진수

개요

- 현 인증서 상황
- 인증서 통합의 필요성 & 방안 제시
    - 인증서는 생성: cert-manager
    - 도메인 네임 통합, gateway 하나의 인증서로 사용: api-gateway (with traefik)
- 통합된 인증서 구조
- **각 팀별 인증서 생성 방법**
    - **Certificate 생성**
    - **생성된 TLS 인증서 확인**
    - **ingress 객체 생성 & 확인**

현 인증서 상황

- 각 팀 마다 다른 방식으로 인증서 생성
  - 쉘 스크립트로 openssl 이용해 self-signed 된 인증서 생성하여 사용
  - 인증서 생성해 주는 이미지 이용하여 생성
  - 오픈 소스 제품 설치 시 알아서 인증서 까지 만들어서 인증서 자체를 모름
  - ETC …

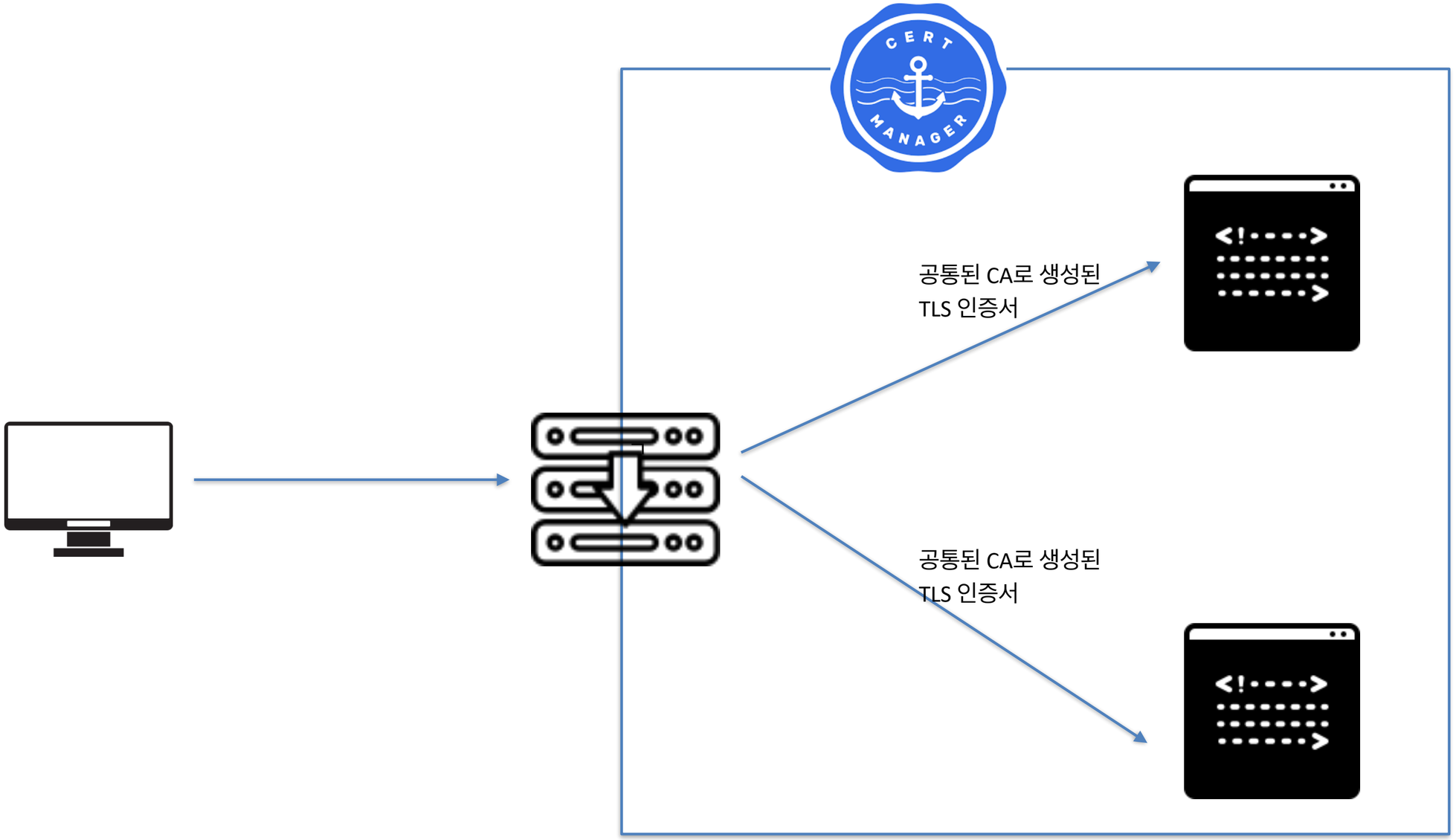
인증서 통합의 필요성 & 방안 제시

- 인증서 통합의 필요성
    - gateway와 backend 간의 https 통신의 문제
        - 각자의 방식으로 만든 인증서 이기 때문에 CA가 다름
            - 현재까진 console에서 insecure 모드로 backend간 통신을 함
                - Insecure 모드란 CA 검증을 하지않고 https 통신을 함
                - CA 검증을 하지 않는다고 해서 https 통신을 안하는 건 아니니 안전하긴 하나 바람직한 모습은 아님
    - Backend 간의 https 통신 문제
        - backend 간의 통신에서 동일한 CA를 가지고 있어야 https 통신이 원할하게 가능
        - 해결방안으로 openssl로 CA 만든 후에 각 제품들이 CA를 통해 인증서를 만들어서 해결
            - 같은 팀내에선 의사소통의 원할함으로 CA를 공유할 수 있으나 팀 간에서 공유하긴 힘듬.
            - tls 인증성 생성하고 사용하기 위해선 동작방식을 이해하고 인증서 구조도 알고 있어야함

인증서 통합의 필요성 & 방안 제시

- 방안 제시
  - cert-manager로 CA를 통합
    - 담당자가 CA 생성 및 관리
  - 각 팀별 담당자는 CA를 기반으로 필요한 인증서 생성
    - 인증서는 cert-manager가 만듬. 필요한 certificate 객체 생성만 하면 됨
  - 인그레스 생성 시 따로 TLS 설정 X
    - annotation에 entrypoint만 지정

통합된 인증서 구조



각 팀별 인증서 생성 방법

- Certificate 생성
  - Certificate 을 생성하면 cert-manager에서 secretName과 동일한 kubernetes.io/tls 시크릿을 생성 해줌

```
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: test
  namespace: test
spec:
  # 원하는 시크릿 이름
  secretName: webserver-secret
  isCA: false
  usages:
    - digital signature
    - key encipherment
    - server auth
    - client auth
  dnsNames:
    # 원하는 dns 이름 설정
    - "webserver.tmaxcloud.org"
  issuerRef:
    kind: ClusterIssuer
    group: cert-manager.io
    # issuer 네임은 tmaxcloud-issuer로 고정
    name: tmaxcloud-issuer
```

각 팀별 인증서 생성 방법

```
cat <<EOF > test-tls.yaml
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: test
  namespace: test
spec:
  # 원하는 시크릿 이름
  secretName: webserver-secret
  isCA: false
  usages:
    - digital signature
    - key encipherment
    - server auth
    - client auth
  dnsNames:
    # 원하는 dns 이름 설정
    - "webserver.tmaxcloud.org"
  issuerRef:
    kind: ClusterIssuer
    group: cert-manager.io
    # issuer 네임은 tmaxcloud-issuer로 고정
    name: tmaxcloud-issuer
EOF
```

```
kubectl apply -f test-tls.yaml
```

```
kubectl get certificate -n test test
```

```
NAME   READY   SECRET             AGE
test   True    webserver-secret   27s
```

```
kubectl get secret-n test webserver-secret
```

```
NAME                TYPE                DATA   AGE
webserver-secret    kubernetes.io/tls   3      101s
```

```
cat <<EOF > test-ingress.yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  labels:
    app.tmaxcloud.org/ingress: test-nginx
  annotations:
    traefik.ingress.kubernetes.io/router.entrypoints: websecure
  name: test-nginx
  namespace: test
spec:
  ingressClassName: tmax
  rules:
    ## nginx.{{ gateway에서 사용하는 기본 domain 주소 }}
    - host: nginx.tmaxcloud.org
      http:
        paths:
          - backend:
              service:
                name: test-nginx
                port:
                  number: 80
            path: /
            pathType: Prefix
```

```
kubectl run —image nginx test-nginx —expose —port 80 -n test
```

```
kubectl get ingress -n test test-nginx
```

```
NAME         CLASS   HOSTS                 ADDRESS         PORTS   AGE
test-nginx   tmax    nginx.tmaxcloud.org   192.168.9.144   80      27m
```

```
kubectl get ingress -A -l tmaxcloud.org/ingress=test-nginx —show-labels
```