[secret을 활용한 https 구성]

```
kevin@k8s-master:~/LABs$ mkdir sec-https && cd $_
kevin@k8s-master:~/LABs/sec-https$

# 인증서 환경 구성
kevin@k8s-master:~/LABs/sec-https$ mkdir -p ./secret/cert
kevin@k8s-master:~/LABs/sec-https$ mkdir -p ./secret/config
kevin@k8s-master:~/LABs/sec-https$ mkdir -p ./secret/kubetmp
kevin@k8s-master:~/LABs/sec-https$ cd ./secret/cert
kevin@k8s-master:~/LABs/sec-https/secret/cert$ openssl genrsa -out https.key 2048
Generating RSA private key, 2048 bit long modulus (2 primes)
..........................+++++
.............+++++
e is 65537 (0x010001)

kevin@k8s-master:~/LABs/sec-https/secret/cert$ openssl req -new -x509 -key https.key -out https.cert
-days 360 -subj /CN=*.kakao.io

kevin@k8s-master:~/LABs/sec-https/secret/cert$  kubectl  create  secret  generic  dshub-https
--from-file=https.key --from-file=https.cert
secret/dshub-https created

kevin@k8s-master:~/LABs/sec-https/secret/cert$ kubectl get secret/dshub-https
NAME          TYPE     DATA   AGE
dshub-https   Opaque   2      35s

kevin@k8s-master:~/LABs/sec-https/secret/cert$ cd ../config/
kevin@k8s-master:~/LABs/sec-https/secret/config$ vi custom-nginx-config.conf
server {
        listen                  8080;
        listen                  443 ssl;
        server_name             www.kakao.io;
        ssl_certificate         certs/https.cert;
        ssl_certificate_key     certs/https.key;
        ssl_protocols           TLSv1 TLSv1.1 TLSv1.2;
        ssl_ciphers             HIGH:!aNULL:!MD5;
        gzip on;
        gzip_types text/plain application/xml;
        location / {
                root    /usr/share/nginx/html;
                index   index.html index.htm;
        }
}

kevin@k8s-master:~/LABs/sec-https/secret/config$ vi sleep-interval
5

kevin@k8s-master:~/LABs/sec-https/secret/config$ cd ..
kevin@k8s-master:~/LABs/sec-https/secret$ kubectl create cm dshub-config --from-file=./config
configmap/dshub-config created

kevin@k8s-master:~/LABs/sec-https/secret$ cd ..
kevin@k8s-master:~/LABs/sec-https$ vi https-pod.yaml
apiVersion: v1
kind: Pod
metadata:
  name: dshub-https
spec:
  containers:
```

```
  - image: dbgurum/k8s-lab:env
    env:
    - name: INTERVAL
      valueFrom:
        configMapKeyRef:
          name: dshub-config
          key: sleep-interval
    name: html-generator
    volumeMounts:
    - name: html
      mountPath: /var/htdocs
  - image: nginx:alpine
    name: web-server
    volumeMounts:
    - name: html
      mountPath: /usr/share/nginx/html
      readOnly: true
    - name: config
      mountPath: /etc/nginx/conf.d
      readOnly: true
    - name: certs
      mountPath: /etc/nginx/certs/
      readOnly: true
    ports:
    - containerPort: 80
    - containerPort: 443
  volumes:
  - name: html
    emptyDir: {}
  - name: config
    configMap:
      name: dshub-config
      items:
      - key: custom-nginx-config.conf
        path: https.conf
  - name: certs
    secret:
      secretName: dshub-https


kevin@k8s-master:~/LABs/sec-https$ kubectl apply -f https-pod.yaml
pod/dshub-https created

kevin@k8s-master:~/LABs/sec-https$ kubectl get po dshub-https -o wide
NAME         READY   STATUS    RESTARTS   AGE     IP              NODE       NOMINATED NODE   READINESS GATES
dshub-https   2/2     Running   0          2m37s   10.111.156.116   k8s-node1   <none>           <none>

# 1번 터미널에서 서비스 게시
kevin@k8s-master:~/LABs/sec-https$ kubectl port-forward dshub-https 8443:443 &
[1] 618204
kevin@k8s-master:~/LABs/sec-https$ Forwarding from 127.0.0.1:8443 -> 443
Forwarding from [::1]:8443 -> 443
Handling connection for 8443
Handling connection for 8443
Handling connection for 8443

-- 2번 터미널: https를 수행하면 1번 터미널에 접속 로그(Handling connection for 844)를 확인할 수 있다.

[root@k8s-master ~]# curl https://localhost:8443 -k -v
*   Trying 127.0.0.1:8443...
* TCP_NODELAY set
* Connected to localhost (127.0.0.1) port 8443 (#0)
* ALPN, offering h2
* ALPN, offering http/1.1
* successfully set certificate verify locations:
```

```
*   CAfile: /etc/ssl/certs/ca-certificates.crt
  CApath: /etc/ssl/certs
* TLSv1.3 (OUT), TLS handshake, Client hello (1):
* TLSv1.3 (IN), TLS handshake, Server hello (2):
* TLSv1.2 (IN), TLS handshake, Certificate (11):
* TLSv1.2 (IN), TLS handshake, Server key exchange (12):
* TLSv1.2 (IN), TLS handshake, Server finished (14):
* TLSv1.2 (OUT), TLS handshake, Client key exchange (16):
* TLSv1.2 (OUT), TLS change cipher, Change cipher spec (1):
* TLSv1.2 (OUT), TLS handshake, Finished (20):
* TLSv1.2 (IN), TLS handshake, Finished (20):
* SSL connection using TLSv1.2 / ECDHE-RSA-AES256-GCM-SHA384
* ALPN, server accepted to use http/1.1
* Server certificate:
*  subject: CN=*.kakao.io
*  start date: Oct  5 15:50:53 2022 GMT
*  expire date: Sep 30 15:50:53 2023 GMT
*  issuer: CN=*.kakao.io
*  SSL certificate verify result: self signed certificate (18), continuing anyway.
> GET / HTTP/1.1
> Host: localhost:8443
> User-Agent: curl/7.68.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Server: nginx/1.23.1
< Date: Wed, 05 Oct 2022 15:58:32 GMT
< Content-Type: text/html
< Content-Length: 50
< Last-Modified: Wed, 05 Oct 2022 15:58:31 GMT
< Connection: keep-alive
< ETag: "633da9a7-32"
< Accept-Ranges: bytes
<
Don't get stuck in a closet -- wear yourself out.
* Connection #0 to host localhost left intact
```

```
root@k8s-master:~
[root@k8s-master ~]# curl https://localhost:8443 -k -v
* About to connect() to localhost port 8443 (#0)
*   Trying ::1...
* Connected to localhost (::1) port 8443 (#0)
* Initializing NSS with certpath: sql:/etc/pki/nssdb
* skipping SSL peer certificate verification
* SSL connection using TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
* Server certificate:
*       subject: CN=*.k8s.com
*       start date: May 02 05:13:28 2021 GMT
*       expire date: Apr 27 05:13:28 2022 GMT
*       common name: *.k8s.com
*       issuer: CN=*.k8s.com
> GET / HTTP/1.1
> User-Agent: curl/7.29.0
> Host: localhost:8443
> Accept: */*
>
< HTTP/1.1 200 OK
< Server: nginx/1.19.10
< Date: Sun, 02 May 2021 05:16:46 GMT
< Content-Type: text/html
< Content-Length: 142
< Last-Modified: Sun, 02 May 2021 05:16:43 GMT
< Connection: keep-alive
< ETag: "608e35bb-8e"
< Accept-Ranges: bytes
<
Perilous to all of us are the devices of an art deeper than we ourselves
possess.
               -- Gandalf the Grey [J.R.R. Tolkien, "Lord of the Rings"]
* Connection #0 to host localhost left intact
```

```
root@k8s-master:~/LABs/secret_https
[root@k8s-master cert]# kubectl create secret generic dshub-https --from-file=ht
cert
secret/dshub-https created
[root@k8s-master cert]# kubectl get secret/dshub-https
NAME          TYPE     DATA   AGE
dshub-https   Opaque   2      6s
[root@k8s-master cert]# cd config/
-bash: cd: config/: No such file or directory
[root@k8s-master cert]# cd ..
[root@k8s-master secret]# cd config/
[root@k8s-master config]# vi custom-nginx-config.conf
[root@k8s-master config]# vi sleep-interval
[root@k8s-master config]# cd ..
[root@k8s-master secret]# kubectl create cm dshub-config --from-file=./config
configmap/dshub-config created
[root@k8s-master secret]# cd ..
[root@k8s-master secret_https]# vi secret-pod.yaml
[root@k8s-master secret_https]# kubectl apply -f secret-pod.yaml
pod/dshub-https created
[root@k8s-master secret_https]# kubectl get pod/dshub-https
NAME          READY   STATUS    RESTARTS   AGE
dshub-https   2/2     Running   0          5s
[root@k8s-master secret_https]# kubectl port-forward dshub-https 8443:443 &
[1] 15940
[root@k8s-master secret_https]# Forwarding from 127.0.0.1:8443 -> 443
Forwarding from [::1]:8443 -> 443
Handling connection for 8443
Handling connection for 8443
Handling connection for 8443
Handling connection for 8443
```