Vue.js에 SSL 인증 추가하기(https)

일요일, 7월 4, 2021 2:40 오후

클리핑 출처: https://deep-dive-dev.tistory.com/26

작업은 ROOT CA 인증서 발급한 후 SSL 인증서 발급, Vue.js에 인증서 정보 추가 순으로 진행된다.

1. SSL 인증서 발급

1.0. Terms

SSL 인증 관련 용어 정리

PKI

- Public Key Infrastructure
- Asymmetric Key Algorithm을 사용
- Key 한 쌍 - Private, Public Key 생성
- Private Key는 개인만이 소유
- Public Key는 공개
- RFC 2459 표준문서 참고
- (참고) X.509는 인증서 포맷을 의미함

CA

- Certificate Authority, 공인 인증 기관
- 일반적으로 우리가 발급 받는 인증서는 신뢰할 수 있는 기관이 발급하고, 상위 기관의 서명을 받음
- 인증 기관들도 상하위 관계가 있고, 최상위 기관을 ROOT CA라고 함

Certificate

- 인증서
- CA는 Private, Public Key 쌍을 만들고, 그 Private Key를 공인 인증 기관에서 적법한 절차를 통해 만들었다고 인증서를 발행
- 인증서 안에 Public Key가 포함됨

Self-signed Certificate

- 최상위 인증기관의 인증서는 누군가 더 상위에서 서명을 해 줄수가 없으므로 스스로 서명을 하여 인증서를 발급
- 개인이 직접 발급하고 스스로 서명하여 ROOT CA와 같이 행동할 수 있음 → **개발/테스트를 위해 ROOT CA 인증서 사용, 하지만 정보가 기관에 등록되있지 않으므로** 신뢰할 수 없는 인증서로 경고가 발생

CSR

- Certificate Signing Request, 인증서 서명 요청
- Private Key, 인증서에 포함되야 하는 식별정보를 담은 인증서 발급 요청 정보

1.1. ROOT CA 인증서 발급

1. CA가 사용할 RSA key(Private key) 생성

```
$ openssl genrsa -aes256 -out rootca.key 2048
```

1. CSR(Certificate Signing Request) 생성

rootca_openssl.conf 작성, CSR 파일을 생성할 때 자동으로 공개키가 생성되어 포함된다.

```
[ req ]
default_bits            = 2048
default_md              = sha1
default_keyfile         = rootca.key
distinguished_name      = req_distinguished_name
extensions              = v3_ca
req_extensions          = v3_ca

[ v3_ca ]
basicConstraints        = critical, CA:TRUE, pathlen:0
subjectKeyIdentifier    = hash
##authorityKeyIdentifier = keyid:always, issuer:always
keyUsage                = keyCertSign, cRLSign
nsCertType              = sslCA, emailCA, objCA

[req_distinguished_name ]
countryName                     = Country Name (2 letter code)
countryName_default             = KR
countryName_min                 = 2
countryName_max                 = 2

organizationName                = Organization Name (eg, company)
organizationName_default        = company

commonName              = Common Name (eg, your name or your servers
hostname)
commonName_default      = Test Self Signed CA
commonName_max          = 64
```

```
 $ openssl req -new -key rootca.key -out rootca.csr -config
rootca_openssl.conf
Enter pass phrase for rootca.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
```

There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.

Country Name (2 letter code) [KR]:KR
Organization Name (eg, company) [company]:
Common Name (eg, your name or your servers hostname) [Test Self Signed CA]:
$ ll
total 20
drwxrwxr-x 2 kronos kronos 4096 7월 24 09:43 ./
drwxr-xr-x 16 kronos kronos 4096 7월 24 09:31 ../
-rw-rw-r-- 1 kronos kronos 1064 7월 24 09:42 rootca_openssl.conf
-rw-rw-r-- 1 kronos kronos 1086 7월 24 09:43 rootca.csr
-rw-rw-r-- 1 kronos kronos 1766 7월 24 09:42 rootca.key

1. self-signed 인증서 생성(기간은 10년으로 설정)

$ openssl x509 -req ₩
-days 3650 ₩
-extensions v3_ca ₩
-set_serial 1 ₩
-in rootca.csr ₩
-signkey rootca.key ₩
-out rootca.crt ₩
-extfile rootca_openssl.conf
 $ cat rootca.crt
-----BEGIN CERTIFICATE-----
MIIDUDCCAjigAwIBAgIBATANBgkqhkiG9w0BAQsFADBAMQswCQYDVQQGEwJL
UjEM
...
v4sPkxXRk1kZupLnRC33aufsVMbmiASbcUwbZttXICmRycbK
-----END CERTIFICATE-----

1.2. SSL 인증서 발급

위에서 생성한 ROOT CA 서명키로 SSL 인증서 발급

1. SSL Host에서 사용할 RSA Key(private key) 생성

$ openssl genrsa -aes256 -out domain.com.key 2048

1. Remove Passphrase from key

개인키를 보호하기 위해 Key-Derived Function 으로 개인키 자체가 암호화되어 있다. 인터넷 뱅킹등에 사용되는 개인용 인증서는 당연히 저렇게 보호되어야 하지만 SSL 에 사용하려는 키가 암호가 걸려있으면 https 구동때마다 pass phrase 를 입력해야 하므로 암호를 제거한다.

```
$ cp domain.com.key  domain.com.key.enc
$ openssl rsa -in domain.com.key.enc -out domain.com.key
```

1. CSR(Certificate Signing Request) 생성

host_openssl.conf 작성

```
[ req ]
default_bits            = 2048
default_md              = sha1
default_keyfile         = rootca.key
distinguished_name      = req_distinguished_name
extensions              = v3_user
## 인증서 요청시에도 extension 이 들어가면 authorityKeyIdentifier 를 찾지 못해 에러가 나므로 막아둔다.
## req_extensions = v3_user

[ v3_user ]
# Extensions to add to a certificate request
basicConstraints = CA:FALSE
authorityKeyIdentifier = keyid,issuer
subjectKeyIdentifier = hash
keyUsage = nonRepudiation, digitalSignature, keyEncipherment
## SSL 용 확장키 필드
extendedKeyUsage = serverAuth,clientAuth
subjectAltName   = @alt_names

[ alt_names]
# DNS.1  = www.tachometer.com
# DNS.2  = tachometer.com
# DNS.3  = *.tachometer.com
DNS  = localhost

[req_distinguished_name ]
countryName                 = Country Name (2 letter code)
countryName_default             = KR
countryName_min                 = 2
countryName_max                 = 2

organizationName            = Organization Name (eg, company)
organizationName_default        = company

organizationalUnitName          = Organizational Unit Name (eg, section)
organizationalUnitName_default  = test SSL Project

# SSL 서비스할 domain 명 입력
commonName                  = Common Name (eg, your name or your servers hostname)
commonName_default              = localhost # ex)domain.com
```

commonName_max = 64

별도로 발급받은 도메인이 없는 경우 host ip 또는 localhost로도 사용 가능

$ openssl req -new -key domain.com.key -out domain.com.csr -config host_openssl.conf
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.

Country Name (2 letter code) [KR]:KR
Organization Name (eg, company) [company]:
Organizational Unit Name (eg, section) [test SSL Project]:
Common Name (eg, your name or your servers hostname) [domain.com]:

1. 5년짜리 domain.com 용 SSL 인증서 발급 (서명시 ROOT CA 개인키로 서명)

$ openssl x509 -req -days 1825 -extensions v3_user -in domain.com.csr ₩
-CA rootca.crt -CAcreateserial ₩
-CAkey rootca.key ₩
-out domain.com.crt -extfile host_openssl.conf
$ cat domain.com.crt
-----BEGIN CERTIFICATE-----
MIIDuDCCAqCgAwIBAgIJAO1RpQKoH0hVMA0GCSqGSIb3DQEBCwUAMEAxCzA
JBgNV
...
oqRqKufadzIMKRs6OeYZgSziAAYfZ0U8AduFVmf5rfX6gteUkxbBIYFcNpM=
-----END CERTIFICATE-----

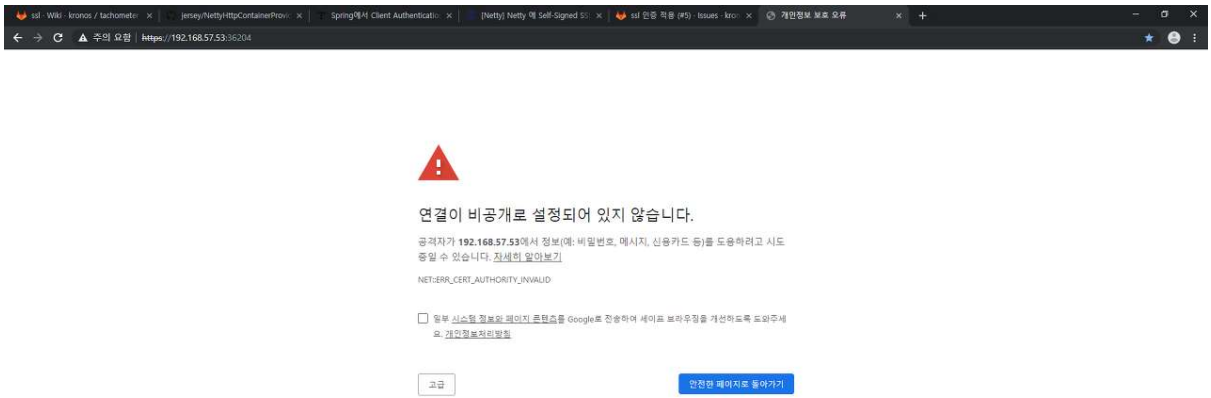
2. Vue.js 설정

webpack.config.js에 아래 내용을 추가

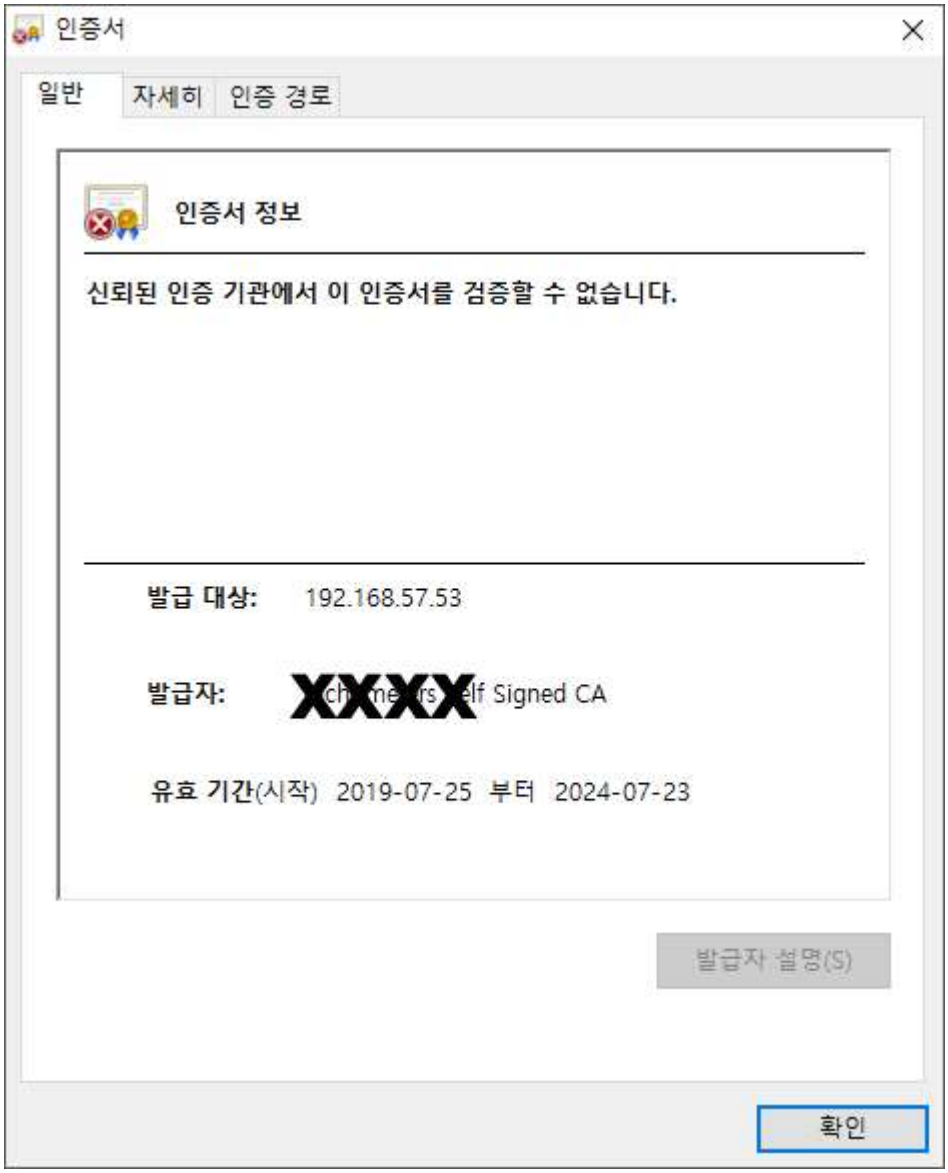
```
module.exports = {
  //...
  devServer: {
    https: {
      key: fs.readFileSync('/path/to/server.key'),
      cert: fs.readFileSync('/path/to/server.crt'),
      ca: fs.readFileSync('/path/to/ca.pem'),
    }
  }
};
```

- key: 서버쪽 비공개키(여기서는 domain.com.key)
- cert: 디지털 인증서(여기서는 domain.com.crt)
- ca: ROOT CA 인증서(여기서는 rootca.crt)

재실행 한 후 페이지에 접속하면 아래와 같은 화면을 확인할 수 있다.



https 설정 후 접속 화면



인증서 정보를 브라우저에서 확인할 수 있다.

References