

pqctoolkit-strongswan

pqctoolkit-strongswan은 양자내성 알고리즘이 적용된 IPSec 도구입니다. 이 문서는 pqctoolkit-strongswan을 설정하고 사용하는 방법을 안내합니다.

- [pqctoolkit-strongswan](#)
- [기본 설정](#)
 - [실행 권한](#)
 - [동적 라이브러리 경로 설정](#)
 - [설정 파일 수정](#)
 - [charon.conf](#)
 - [pki.conf](#)
 - [인증서 생성](#)
 - [Root](#)
 - [Root key 생성](#)
 - [Root 인증서 생성](#)
 - [Server](#)
 - [Server key 생성](#)
 - [Server 인증서 생성 및 서명](#)
 - [Client](#)
 - [Client Key 생성](#)
 - [Client 인증서 생성 및 서명](#)
 - [인증서 및 Key 파일 복사](#)
- [구동](#)
 - [Server 설정](#)
 - [Server 실행](#)
 - [Client 설정](#)
 - [Client 실행](#)
- [테스트](#)
 - [설정 스크립트](#)
 - [테스트 스크립트](#)
- [지원환경](#)

기본 설정

다운로드 받은 pqctoolkit-strongswan은 `/usr/local/pqctoolkit-strongswan`에 위치시킵니다. 이 문서에서 설정하는 pqctoolkit-strongswan 파일들의 경로의 root 디렉토리는 `/usr/local/pqctoolkit-strongswan`입니다. 예시로 `etc/strongswan.d/pki.conf`의 절대경로는 `/usr/local/pqctoolkit-strongswan/etc/strongswan.d/pki.conf`입니다.

실행 권한

pqctoolkit-strongswan은 root 권한으로 실행되어야 하므로 사용자를 root로 바꾼 후 이후의 절차를 수행해야 합니다.

동적 라이브러리 경로 설정

pqctoolkit-strongswan은 PQC(Post-Quantum-Cryptography)를 지원하기 위해 pqctoolkit-library의 shared library를 필요로 합니다. pqctoolkit-library의 shared library를 다운로드 받은 후 이르 사용할 수 있도록 라이브러리 위치를 다음과 같이 설정합니다.

```
export LD_LIBRARY_PATH=/path/to/pqctoolkit-library/lib:$LD_LIBRARY_PATH
ldconfig
```

설정 파일 수정

pqctoolkit-strongswan 관련 파일들은 pqctoolkit-strongswan 설치 디렉토리에 있는 `etc/strongswan.d` 디렉토리 내부에 종류에 따라 존재합니다.

charon.conf

```
accept_private_algs = yes # pqctoolkit-library에서 지원하는 알고리즘을 사용하기
                           # 위함
max_packet = 60000        # pqctoolkit-library의 IKEv2 KEM 최대 패킷 크기가
                           # 클 수 있음.
fragment_size = 1440
```

pki.conf

pqctoolkit-strongswan의 utility tool 중 인증서 파일 생성에 사용되는 `pki`가 `pqctoolkit-library`를 사용하도록 `pki.conf`의 내용 중 `load` 항목을 다음과 같이 수정합니다.

```
load = random drbg x509 pubkey pkcs1 pkcs8 pem openssl oqs
```

수정된 `pki.conf` 파일의 예시:

```
pki {
    # Plugins to load in the pki tool.
    load = random drbg x509 pubkey pkcs1 pkcs8 pem openssl oqs
    scep {
        # Source IP address to bind for HTTP operations.
        # http_bind =

        # Timeout for HTTP operations.
```

```
# http_timeout = 30s

# Some SCEP servers (e.g. openxpki) are incorrectly doing
certificate
# renewal via messageType PKCSReq (19) instead of RenewalReq (17).
# renewal_via_pkcs_req = no
    }
}
```

인증서 생성

pqctoolkit-strongswan이 사용할 인증서를 생성하기 위해 pqctoolkit-strongswan이 제공하는 **pki**를 이용합니다. **pki**는 pqctoolkit-strongswan 설치 디렉토리 내에 **bin/**에 존재합니다.

Root

Root key 생성

haetae2, **haetae3**, **haetae5** 등의 서명 알고리즘을 사용할 수 있으며 아래 예제는 **haetae5** 알고리즘을 이용해서 root key를 만들어 **cryptolab_ca_key.pem** 파일로 저장합니다.

```
pki --gen --type haetae5 --outform pem > cryptolab_ca_key.pem
```

Root 인증서 생성

아래 예제는 생성된 상기 항목에서 생성한 root key를 이용하여 유효기간 10년이 인증서를 생성하여 **cryptolab_ca_cert.pem** 파일로 저장합니다.

```
pki --self --type priv --in cryptolab_ca_key.pem \
--ca --lifetime 3650 \
--dn "C=KR, O=cryptolab, CN=Strongswan Test Root CA" \
--outform pem > cryptolab_ca_cert.pem
```

Server

Server key 생성

root의 경우와 마찬가지로 **haetae2**, **haetae3**, **haetae5** 등의 서명 알고리즘을 사용할 수 있으며 아래 예제는 **haetae5** 알고리즘을 이용해서 server key를 만들어 **server_key.pem** 파일로 저장합니다.

```
pki --gen --type haetae5 --outform pem > server_key.pem
```

Server 인증서 생성 및 서명

아래 예제는 상기 항목에서 생성한 root key 및 인증서를 이용하여 서명된 server 인증서를 생성합니다.

```
pki --issue \  
    --cacert cryptolab_ca_cert.pem \  
    --cakey cryptolab_ca_key.pem \  
    --type priv --in server_key.pem --lifetime 365 \  
    --dn "C=KR, O=cryptolab, CN=testserver.cryptolab.co.kr" \  
    --san testserver.cryptolab.co.kr \  
    --outform pem > server_cert.pem
```

Client

Client Key 생성

상기 항목들과 마찬가지로 **haetae2**, **haetae3**, **haetae5** 등의 알고리즘을 사용할 수 있으며 아래 예제는 **haetae5** 알고리즘을 이용해서 Client key를 만들어 **client_key.pem** 파일로 저장한다.

```
pki --gen --type haetae5 --outform pem > client_key.pem
```

Client 인증서 생성 및 서명

아래 예제는 상기 항목에서 생성한 root key 및 인증서를 이용하여 서명된 client 인증서를 생성합니다.

```
pki --issue \  
    --cacert cryptolab_ca_cert.pem \  
    --cakey cryptolab_ca_key.pem \  
    --type priv --in client_key.pem --lifetime 365 \  
    --dn "C=KR, O=cryptolab, CN=strongswantest@cryptolab.co.kr" \  
    --san strongswantest@cryptolab.co.kr \  
    --outform pem > client_cert.pem
```

인증서 및 Key 파일 복사

각종 인증서들은 pqctoolkit-strongswan 설치 디렉토리내에 **etc/swanctl** 내부에 파일 종류에 따라 서로 다른 디렉토리에 각각 저장되어야 합니다.

- Root CA 인증서: **cryptolab_ca_cert.pem**를 **etc/swanctl/x509ca**에 복사합니다.
- Server Key: **server_key.pem**을 server 역할을 하는 장비에 있는 **etc/swanctl/pkcs8**에 복사합니다.
- Client Key: **client_key.pem**을 client 역할을 하는 장비에 있는 **etc/swanctl/pkcs8**에 복사합니다.
- Server 인증서: **server_cert.pem**을 server 역할을 하는 장비에 있는 **etc/swanctl/x509**에 복사합니다.
- Client 인증서: **client_cert.pem**을 client 역할을 하는 장비에 있는 **etc/swanctl/x509**에 복사한다.

구동

Server 설정

Client의 접속을 허용할 인증 방법 및 cipher suite에 대한 설정은 pqctoolkit-strongswan 설치 디렉토리내에 있는 `etc/swanctl/swanctl.conf` 파일에 있습니다. 아래 예시에서는 Client를 인증서 기반으로 인증하며, `cryptolab_ca_cert.pem` 파일을 이용하여 Client의 인증서를 검증합니다. Server를 인증 하기 위해서는 `server_cert.pem` 파일을 사용합니다.

`swanctl.conf` 파일 예시

```
connections {
    rw {
        pools = rw_pool

        local {
            auth = pubkey
            certs = server_cert.pem
            id = testserver.cryptolab.co.kr
        }
        remote {
            auth = pubkey
            cacerts = cryptolab_ca_cert.pem
        }
        children {
            net {
                local_ts = 10.1.0.0/24
                esp_proposals = aes256-sha256-x25519-ke1_kyber5-ke2_bike3-
ke3_hqc3-ke3_none-ke4_hqc5-ke4_none
            }
            host {
                esp_proposals = aes256-sha256-modp3072-ke1_frodoa3-
ke2_bike3
            }
        }
        version = 2
        # IKEv2 협상시 사용될 cipher suites
        proposals = aes256-sha256-x25519-modp3072-ke1_kyber5,aes256-aes192-
aes128-sha512-sha256-sha1-x25519-modp3072
    }
}

pools {
    rw_pool {
        addrs = 10.3.0.0/24
    }
}
```

Server 실행

pqctoolkit-strongswan 서버를 구동하기 위해서 pqctoolkit-strongswan 설치 디렉토리내에 있는 `libexec/ipsec/charon`을 background에서 root 권한으로 실행합니다.

```
charon &
```

pqctoolkit-strongswan 서버가 설정을 로드하게 하기 위해서 pqctoolkit-strongswan 설치 디렉토리내에 있는 `sbin/swanctl`를 수행합니다.

```
swanctl -load-all
```

Client 설정

Server에 접속시 사용할 인증 방법 및 cipher suite에 대한 설정은 pqctoolkit-strongswan 설치 디렉토리내에 있는 `etc/swanctl/swanctl.conf` 파일에 있습니다. 아래 예시에서는 Server 접속 시 인증서 기반의 방식을 사용하며 `client_cert.pem`을 이용하여 server에 접속합니다.

```
connections {
    home {
        remote_addrs = 123.123.123.123      # 접속하고자 하는 서버 주소
        vips = 0.0.0.0

        local {
            auth = pubkey                    # 인증서 기반 인증
            certs = client_cert.pem          # 클라이언트 인증시 사용할 인증서
파일명
            id = strongswantest@cryptolab.co.kr # 인증서에 명시된 CN
        }
        remote {
            auth = pubkey
            id = testserver.cryptolab.co.kr   # 서버인증서에 있는 CN
        }
        children {
            net {
                esp_proposals = aes256-sha256-x25519-ke1_kyber5-ke2_bike3-
ke3_hqc3-ke3_none-ke4_hqc5-ke4_none
                rekey_time = 20m
            }
        }
        version = 2
        # IKEv2 협상시 사용될 cipher suits
        proposals = aes256-sha256-x25519-modp3072-ke1_kyber5,aes256-aes192-
aes128-sha512-sha256-sha1-x25519-modp3072
```

```
}  
}
```

Client 실행

pqctoolkit-strongswan을 Client로서 서버에 접속하기 위해서 strongSwan 설치 디렉토리내에 있는 `libexec/ipsec/charon`을 background에서 root 권한으로 실행합니다.

```
charon &
```

strongSwan client가 설정을 로드하기 위해 pqctoolkit-strongswan 설치 디렉토리내에 있는 `sbin/swanctl`을 실행합니다.

```
swanctl -load-all
```

위의 예시의 경우 children내에 존재하는 net을 이용하여 Server에 접속하기 위해서 `swanctl`을 다음과 같이 실행합니다.

```
swanctl --initiate --child net
```

테스트

테스트를 위해 `example` 내의 스크립트들을 실행할 수 있습니다.

설정 스크립트

- `generate_ca.sh`: Root CA 용 `haetae5` 기반 key 및 self-signed 인증서 생성
- `generate_server_key.sh`: Server용 `haetae5` 기반 key 생성 후 `generate_ca.sh`를 이용하여 생성된 Root CA key 및 인증서를 이용하여 서명된 server 인증서 생성
- `generate_client_key.sh`: 각 알고리즘에 따라 client key 생성 후 `generate_ca.sh`를 이용하여 생성된 Root CA key 및 인증서를 이용하여 서명된 client 인증서 생성

테스트 스크립트

`oqs_test.sh`, `swancontrol.py`, `swancontrol.sh`를 이용하여 pqctoolkit-strongswan 서버를 검증하기 위해 각 알고리즘에 해당하는 설정을 한 후 접속 시도 및 결과를 출력합니다. 성공적으로 `oqs_test.sh`를 수행 하기 위해서는 다음과 같은 조건을 만족해야 합니다.

- `charon`이 이미 수행되고 있어야 함
- `swanctl`의 경로가 `PATH` 환경변수에 설정되어져야 함
- `python3` 및 `python vici`가 설치되어 있어야 함

지원환경

pqctoolkit-strongswan는 x86_64 프로세서의 Ubuntu 22.04에서 구현되고 테스트되었습니다.