**Connessione tls post-quantum**

# Introduzione

OQS abilita connessioni di tipo TLS v1.3 (ma non versioni inferiori) che fanno uso di algoritmi post-quantum.

Il progetto complessivo si articola in due progetti più piccoli:

* Il primo si occupa di generare i certificati necessari;
* il secondo effettua la connessione TLS vera e propria.

# Primo progetto: Generazione dei certificati

Per la generazione dei certificati, è stato creato un primo progetto basato su un'[immagine](https://hub.docker.com/r/openquantumsafe/oqs-ossl3) Docker contenente liboqs, OpenSSL e OQS-Provider già preinstallati.

Per essere precisi:

* openssl è il tool che consente di usare gli strumenti crittografici classici come RSA o le curve ellittiche per cifratura e firma digitale;
* liboqs è la libreria scritta in C per l’implementazione degli algoritmi post quantum finora approvati dal NIST;
* oqs-provider consiste principalmente nell’integrazione tra openssl e liboqs.

L’immagine di cui sopra, tuttavia, è consigliata esclusivamente per scopi di testing, poiché include certificati precaricati forniti da OQS. Tali certificati servono a semplificare il test di connessioni TLS utilizzando le direttive di OpenSSL.

La generazione dei certificati richiede un file di configurazione specifico, denominato *oqs\_openssl.cnf*. Per ottenere un certificato server firmato da una CA fittizia, vengono creati due certificati: uno per la CA fittizia e uno per il server. Tuttavia, questa configurazione risulta superflua se non si intende verificare la validità del certificato server; in tal caso, è sufficiente un unico certificato per il server, necessario per il corretto funzionamento di TLS.

La CA utilizzata è fittizia, nel senso che i certificati da essa forniti sono assolutamente analoghi a quelli forniti dal server, ma con impostazioni specifiche che ne definiscono il ruolo di CA. Tali impostazioni sono indicate nella sezione [v3\_ca] del file di configurazione “oqs\_openssl.cnf”. In questo caso i comandi da lanciare (da shell del container Docker generato “cert-generator”) sono i seguenti tre:

* **openssl req -x509 -new -newkey dilithium5 -keyout /certs/CA.key -out /certs/CA.crt -nodes -days 365 -config certs/oqs\_openssl.cnf -subj "/CN=oqstest CA" -extensions v3\_ca**
* **openssl req -new -newkey dilithium5 -keyout /certs/server.key -out /certs/server.csr -nodes -config certs/oqs\_openssl.cnf -subj "/CN=nginx\_pq" -extensions v3\_req**
* **openssl x509 -req -in /certs/server.csr -out /certs/server.crt -CA /certs/CA.crt -CAkey /certs/CA.key -CAcreateserial -days 365**

Se invece si opta per la creazione di un solo certificato, la sezione finale del file di configurazione (da [req] a linea 22 in poi) può essere eliminata, e il comando per generare il certificato del server si modifica nel seguente modo:

* **openssl req -x509 -new -newkey dilithium5 -keyout /certs/server.key -out /certs/server.crt -nodes -days 365 -config certs/oqs\_openssl.cnf -subj "/CN=nginx\_pq"**.

Il Common Name (CN) per la CA può essere scelto arbitrariamente, mentre per il server deve corrispondere al link a cui curl si deve connettere. In un progetto docker due container possono comunicare utilizzando il nome di quel container. Se il nome del container cambia, il CN del certificato del server deve essere aggiornato di conseguenza.

Una volta creati, i certificati della CA e del server vengono concatenati e passati manualmente al secondo progetto. Tuttavia, i certificati generati presentano restrizioni di utilizzo che devono essere rimosse. Per praticità, questa operazione viene eseguita nel secondo progetto.

# Secondo progetto: Costruzione app

Questo progetto prevede l’uso di un container per rimuovere le restrizioni dai certificati generati e un altro container dedicato alla configurazione di un server basato su Flask. La gestione della comunicazione TLS è affidata a una versione modificata di Nginx, appositamente progettata per supportare algoritmi di crittografia post-quantum. La configurazione di Nginx è controllata tramite un file specifico (“nginx.conf”) che consente di definire in modo chiaro e agevole i parametri necessari al suo funzionamento.

Attualmente, i browser tradizionali non supportano certificati con firme basate su algoritmi post-quantum, rendendo impossibile accedere al servizio tramite i normali link generati. Per verificare il corretto funzionamento della connessione TLS invece, si utilizza una versione modificata di curl, sviluppata dal team OQS, in grado di gestire la crittografia post-quantum. Questo strumento permette di specificare opzioni come la verifica del certificato o la selezione di un particolare KEM (Key Encapsulation Mechanism). In assenza di una configurazione esplicita, curl utilizza di default le curve ellittiche per la cifratura.