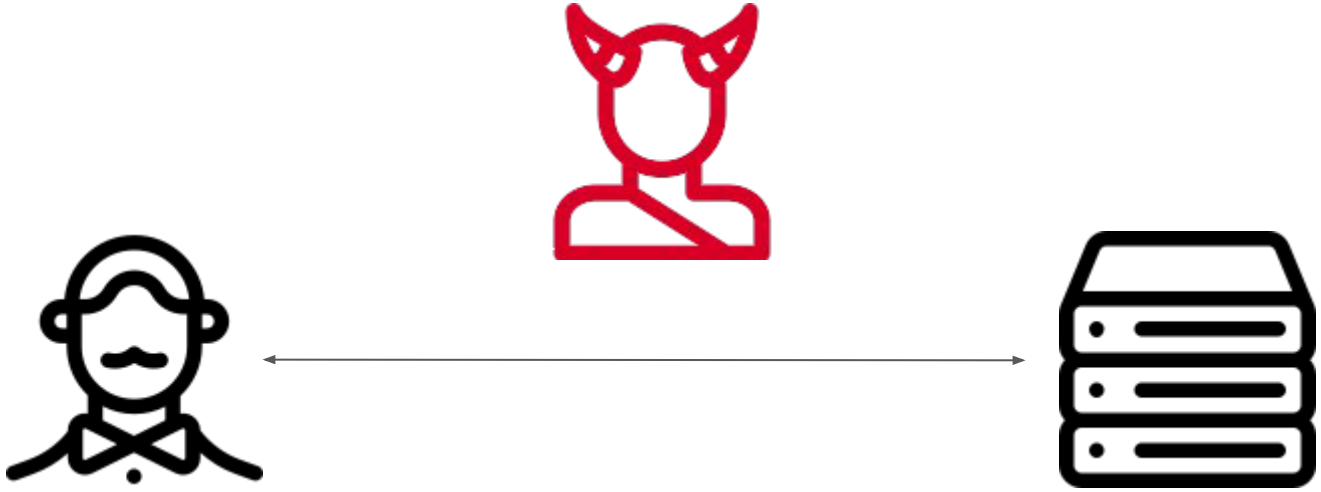
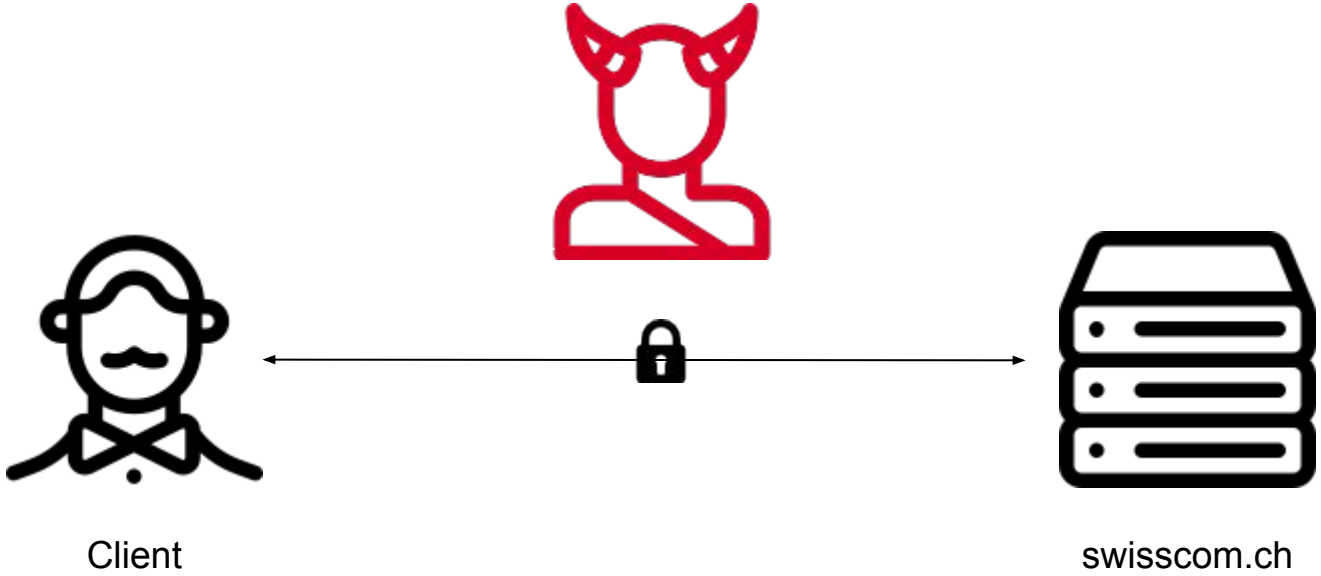
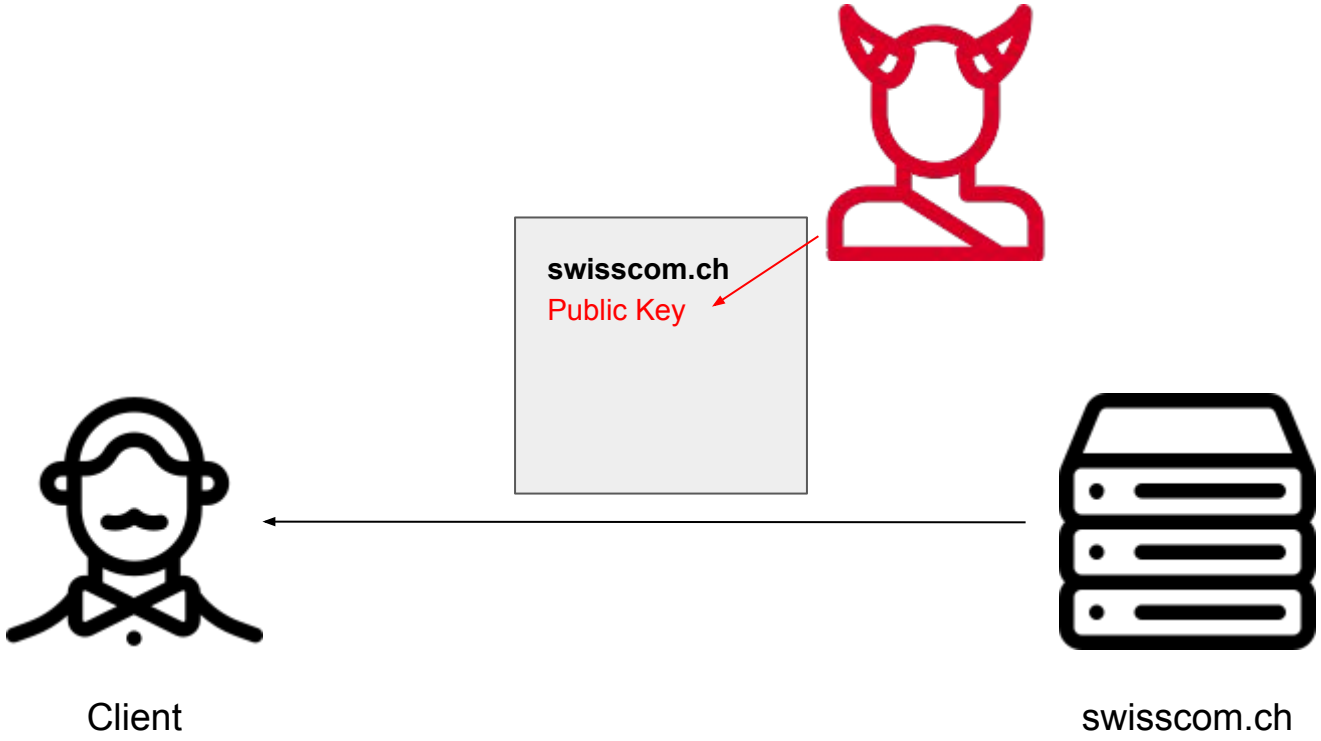
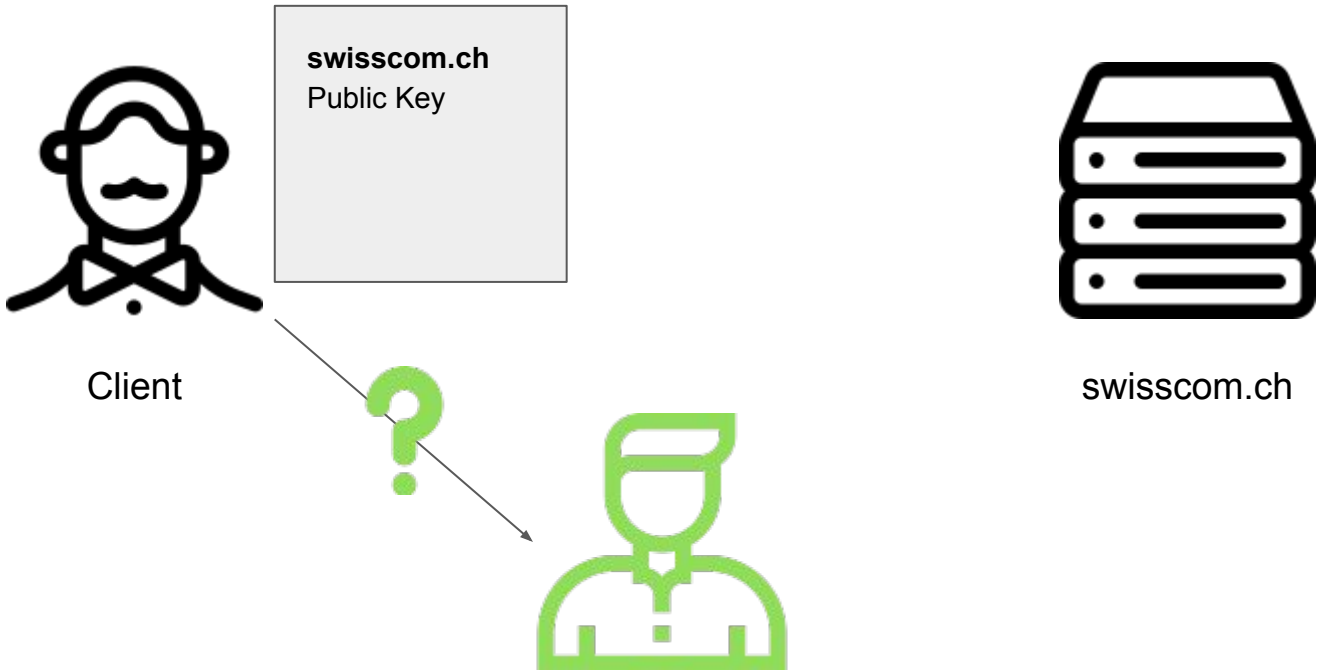
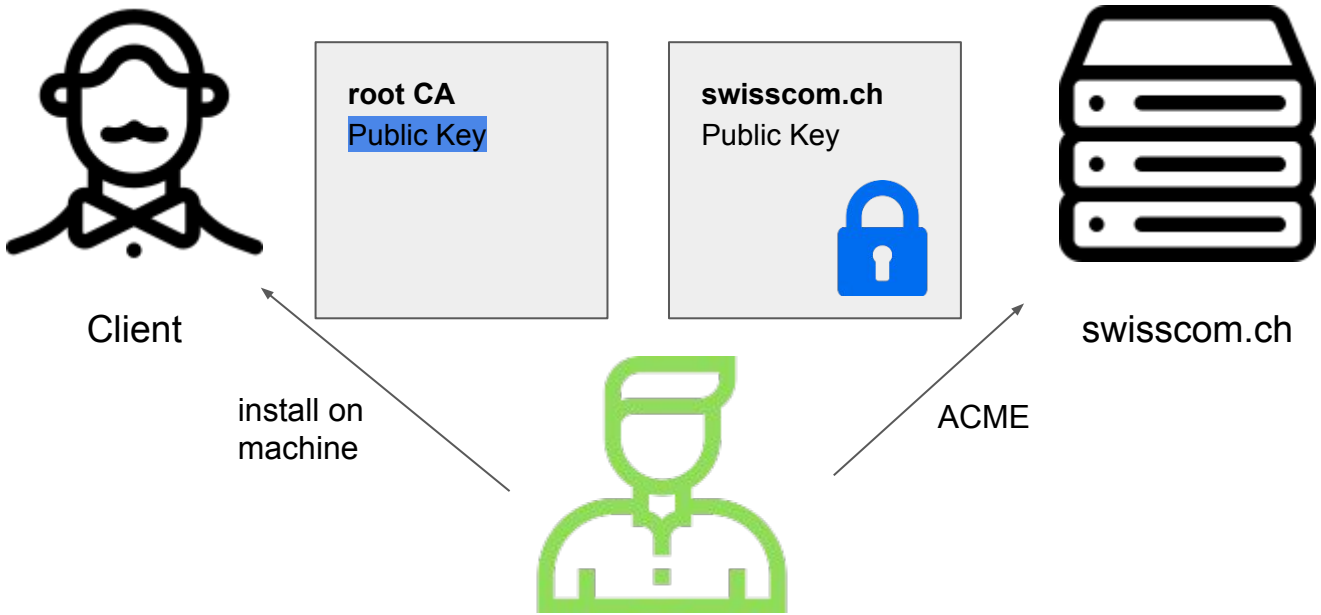

Client
swisscom.ch


Client
swisscom.ch


swisscom.ch
Public Key
Client
swisscom.ch


swisscom.ch
Public Key
Client
swisscom.ch

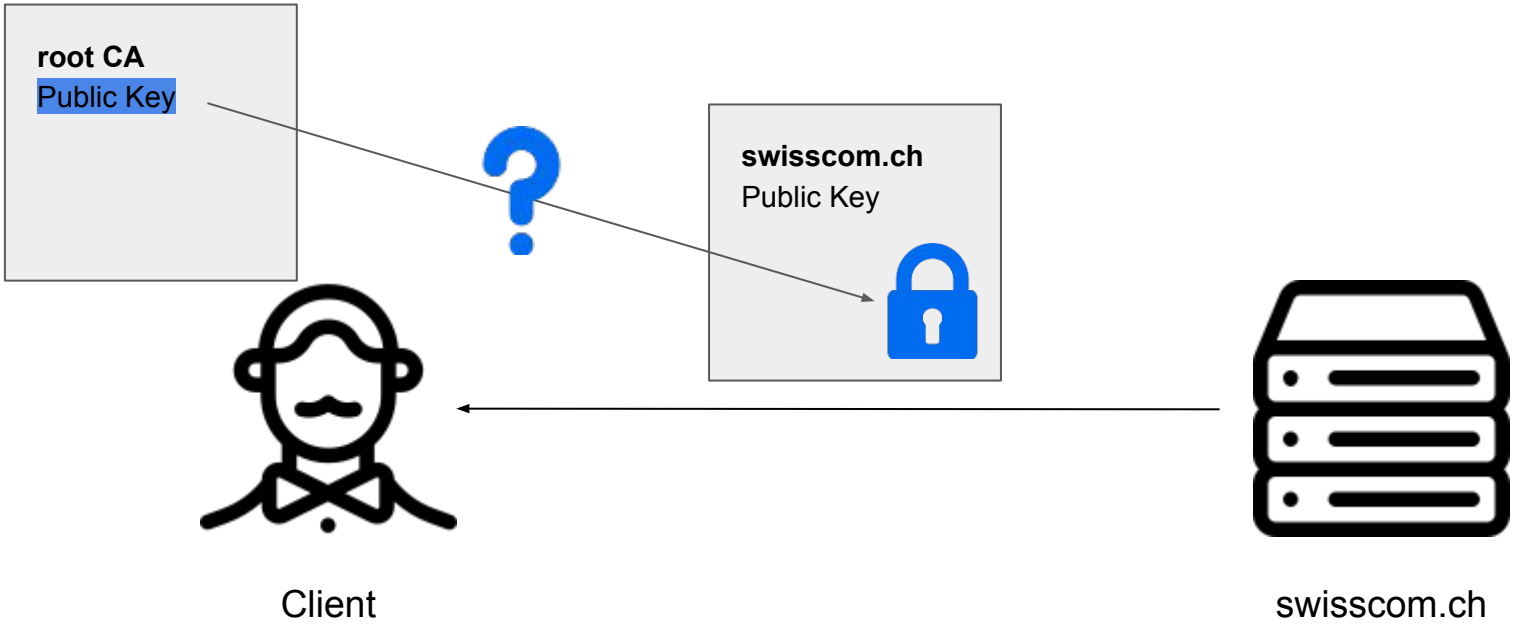

swisscom.ch
Public Key
Client
swisscom.ch


root CA
Public Key
swisscom.ch
Public Key
Client
swisscom.ch
install on
machine
ACME

root CA
Public Key

Client

swisscom.ch


root CA
Public Key
swisscom.ch
Public Key
Client
swisscom.ch

Task

1. Retrieve a TLS certificate for a freely choosable URL.
2. Present the most important information to the user.

Tools to retrieve a certificate

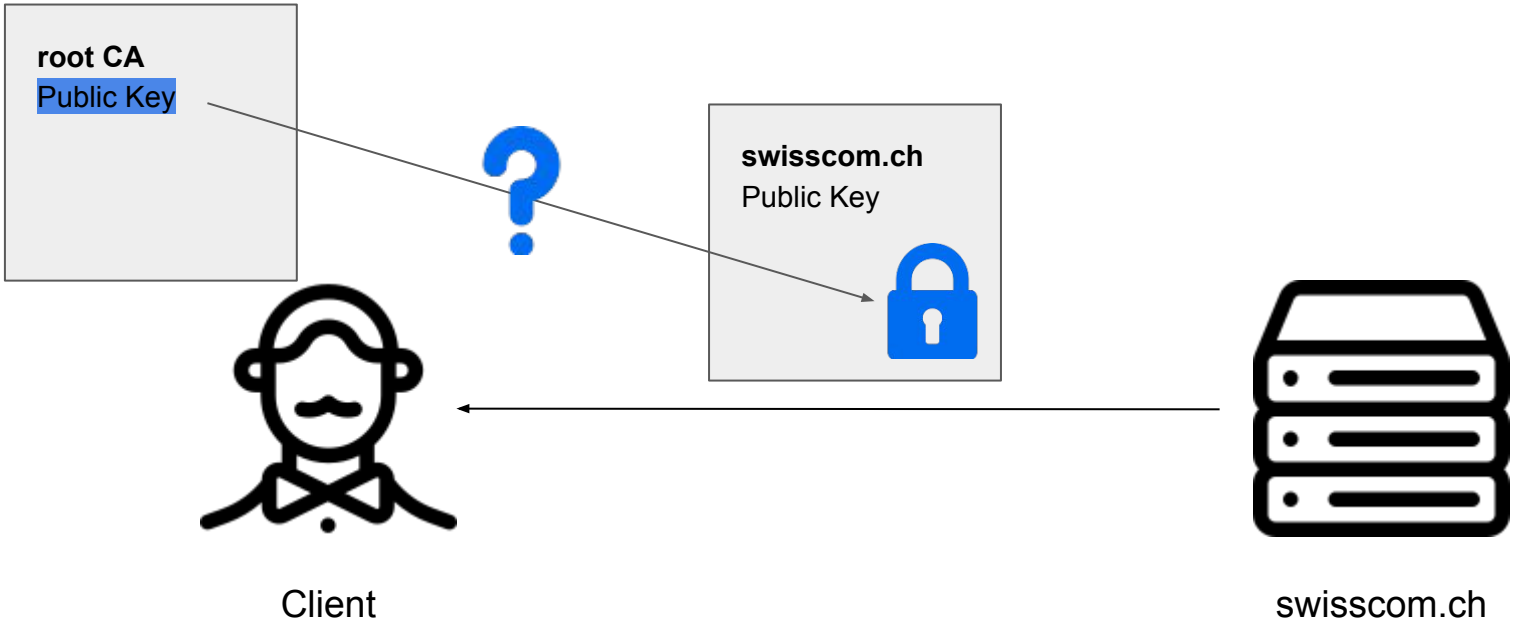
- Python: libraries *ssl* and *OpenSSL*
  - Problem: doesn’t support manual signature verification
- Golang: library *crypto/tls*
- Command line tool *openssl*
  - widely used (= support)

Task

1. Retrieve a TLS certificate for a freely choosable URL. →*openssl*
2. Present the most important information to the user.

Task

1. Retrieve a TLS certificate for a freely choosable URL. →*openssl*
2. Present the most important information to the user.


root CA
Public Key
swisscom.ch
Public Key
Client
swisscom.ch

Most important information

- To whom was the certificate issued?
- Who issued it?
- Is it not yet/still valid?
- Signature
  - certificate digest and root CA’s public key is used to verify
- Public key

Outline

1. Retrieve the certificate.
2. Print the issuer.
3. Print the subject.
4. Print the SANs.
5. Print the issue date.
6. Print the expiration date.
7. Print the certificate digest (SHA256 and SHA1).
8. Print the public key.

- User can verify the public key’s integrity.
  - verify the signature of the digest using root’s public key
  - check its validity
  - (omitted: CRL)
- Use public key to encrypt traffic in TLS Handshake protocol.
  - i.e., set up symmetric key for TLS Record protocol