

## Class activity

A hands-on exercise from the class-activities appendix. The full text also appears in Appendix A of the book.

---

Accompanies *Computer Security: The Foundations*, Krerk Piromsopa, Ph.D., 2026.  
<https://www.cp.eng.chula.ac.th/~krerk/books/ComputerSecurity/>

### Authorized use only

These exercises involve real attack techniques. Carry them out only against systems you own, or that you have written permission from the owner to test. Unauthorized access, scanning or interception is a criminal offence under Thailand's Computer Crime Act and its equivalents elsewhere, whether or not any damage results. If you are unsure whether a target is in scope, ask your instructor before you start.

## Activity 6b: Public Key Infrastructure

### Overview

In this activity you will learn the fundamentals of Public Key Infrastructure (PKI).

### Tools required:

- A. **OpenSSL**. Pre-installed on Linux and macOS. For Windows, download from <https://wiki.openssl.org/index.php/Binaries>.
- B. **Python** with pyopenssl and pem. Install with pip if not already present:  

```
$ pip install pyopenssl  
$ pip install pem
```
- C. The **CA certificate bundle** from your OS (e.g. `/etc/ssl/certs/ca-certificates.crt` on Linux, `/etc/ssl/cert.pem` on macOS).

### Exercise

Issue the following command:

```
$ openssl s_client -connect www.chula.ac.th:443
```

Once connected, you may try:

```
GET / HTTP/1.0  
[press Enter twice]
```

(The server may answer with an error such as HTTP 404 or HTTP 503: the request carries no Host header, so a host that serves several sites cannot tell which one you meant. This is normal, and the certificate has already been exchanged by then.)

Repeat the step with the CA file:

```
$ openssl s_client -connect www.chula.ac.th:443 \  
-CAfile ca-certificates.crt
```

1. What is the difference between the two commands above?  
*Note:* If your OS shows no error in the first command, try `-CApath /dev/null`.
2. What does the verify error in the first command mean?
3. Copy the server certificate (from -----BEGIN CERTIFICATE----- to -----END CERTIFICATE-----) and save it as `chula_ac_th.cert`. Use the command below to display its contents, then briefly explain the meaning of each field:

```
$ openssl x509 -in chula_ac_th.cert -text
```

4. From the information in Exercise 3, is there an intermediate certificate? If yes, what purpose does it serve?  
*Hint:* Look for the Issuer field and download the intermediate certificate. Use the following to inspect it:

```
$ openssl x509 -inform der -in intermediate.cert -text
```

5. Is there an intermediate CA — i.e. is more than one organization involved in certification? Explain your reasoning.
6. What is the role of `ca-certificates.crt`?
7. Explore `ca-certificates.crt`. How many certificates does it contain? State the command/method you used to count.
8. Extract a root certificate from `ca-certificates.crt` and inspect it with OpenSSL. Do you see any Issuer information? Compare the root certificate details with those of the server certificate you saved and the intermediate certificate.
9. If the intermediate certificate is not in PEM format, convert it:

```
$ openssl x509 -inform der -in certificate.cer \
-out certificate.pem
```

10. Using the Python code below, implement certificate validation. Verify the certificates of: `www.google.com`, `www.cp.eng.chula.ac.th`, and `classdeedee.cloud.cp.eng.chula.ac.th`.

```
from OpenSSL import crypto
import pem

def verify():
    with open('./target.cert', 'r') as cert_file:
        cert = cert_file.read()
    with open('./intermediate.cert', 'r') as int_cert_file:
        int_cert = int_cert_file.read()

    pems = pem.parse_file('./ca-certificates.cert')
    trusted_certs = []
    for mypem in pems:
        trusted_certs.append(str(mypem))
    trusted_certs.append(int_cert)

    verified = verify_chain_of_trust(cert, trusted_certs)
    if verified:
        print('Certificate verified')

def verify_chain_of_trust(cert_pem, trusted_cert_pems):
    certificate = crypto.load_certificate(
        crypto.FILETYPE_PEM, cert_pem)
    store = crypto.X509Store()
```

```
for trusted_cert_pem in trusted_cert_pems:
    trusted_cert = crypto.load_certificate(
        crypto.FILETYPE_PEM, trusted_cert_pem)
    store.add_cert(trusted_cert)
store_ctx = crypto.X509StoreContext(store, certificate)
result = store_ctx.verify_certificate()
return result is None
```

11. Today there are root certificates for Class 1 and Class 3. What uses would a Class 1 signed certificate have that Class 3 would not, and vice versa?
12. Assume that a root CA in your root store is compromised and under attacker control for several months without anyone noticing.
  - (a) What further attacks can the attacker stage? Draw a possible attack setup.
  - (b) In the attack you described, can CRLs or OCSP provide protection? Explain.

---

From *Computer Security: The Foundations*, Kerk Piromsopa, Ph.D.  
Reproduce and adapt freely for non-commercial classroom instruction, with attribution. Full terms: LICENSE.txt in the student package.