SENG2250/6250 System and Network Security

School of Electrical Engineering and Computing

Semester 2, 2020

Lab 5: PKI

Objectives

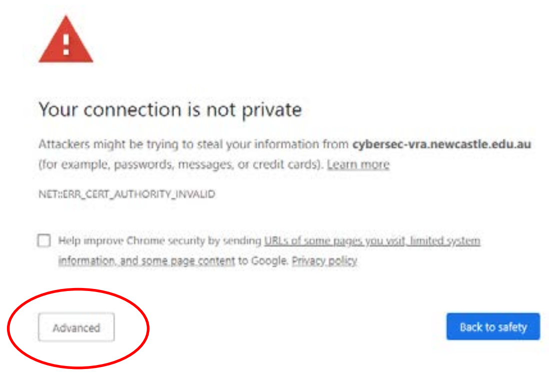
1) Setup virtual lab environment.
2) Learn how to create public key certificates.
3) Implement cross-certification for a given hierarchy.

Notes

- The virtual lab is accessible in the **lab time and room only**.
- This lab's required system and software are listed here.
- The lab initialisation may take around 20 minutes for the first access. The lab can be recreated/resumed quickly later on.
- The virtual lab will be automatically turned off at the end of the lab.
- Virtual lab is isolated to the host machine and Internet. You cannot copy in/out between the host and virtual machines.

Part 1 Setup Virtual Lab Environment

Read the instructions CyberSecurity-Teaching-UserGuide.pdf and set up your personal virtual lab environment. (The instruction file is available on the Blackboard). If the following waring page prompts (e.g., in Chrome): Please ignore the warning and click "Advanced" → "Proceed to cybersec-vra.newcastle.edu.au (unsafe)".

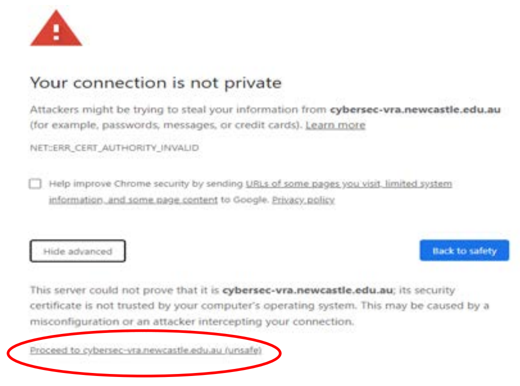


Your connection is not private

Attackers might be trying to steal your information from **cybersec-vra.newcastle.edu.au** (for example, passwords, messages, or credit cards). Learn more

NET::ERR_CERT_AUTHORITY_INVALID

☐ Help improve Chrome security by sending URLs of some pages you visit, limited system information, and some page content to Google. Privacy policy



Your connection is not private

Attackers might be trying to steal your information from **cybersec-vra.newcastle.edu.au** (for example, passwords, messages, or credit cards). Learn more

NET::ERR_CERT_AUTHORITY_INVALID

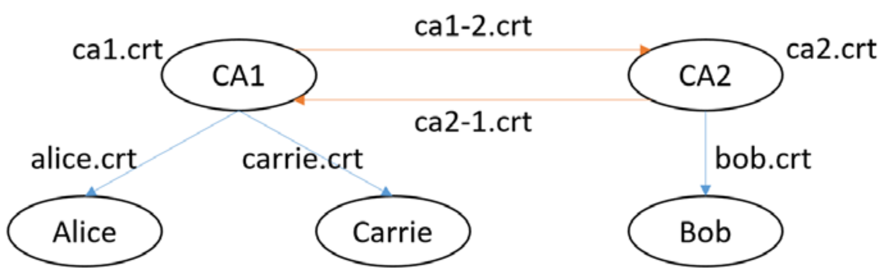
☐ Help improve Chrome security by sending URLs of some pages you visit, limited system information, and some page content to Google. Privacy policy

This server could not prove that it is **cybersec-vra.newcastle.edu.au**; its security certificate is not trusted by your computer's operating system. This may be caused by a misconfiguration or an attacker intercepting your connection.

Proceed to cybersec-vra.newcastle.edu.au (unsafe)

Part 2 Cross-Certification

In this exercise, you will create certificates as the following structure.



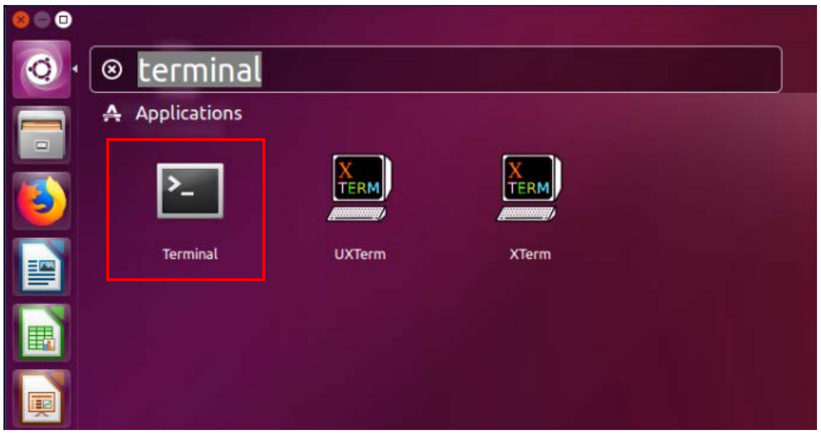
Here is the List of Commands needed in this lab.

Start an Ubuntu Virtual Machine

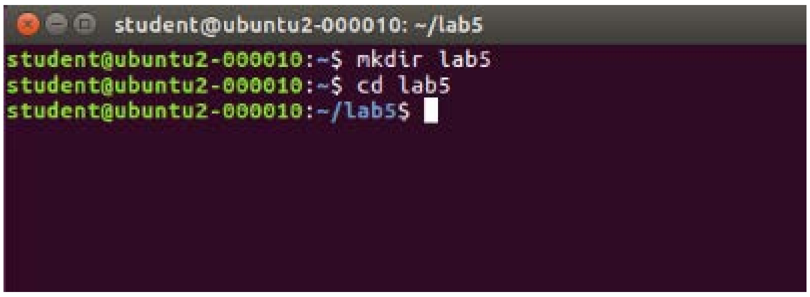
Follow the "CyberSecurity-Teaching-UserGuide" instructions to start an Ubuntu virtual machine. It can be any of three listed Ubuntu virtual machines.

Create lab directory

1. Find and open a "terminal" as following.



2. Create lab directory, then go to the directory
   ```
   mkdir lab5
   cd lab5
   ```



```
student@ubuntu2-000010: ~/lab5
student@ubuntu2-000010:~$ mkdir lab5
student@ubuntu2-000010:~$ cd lab5
student@ubuntu2-000010:~/lab5$
```

Create a self-signed certificate for CA1

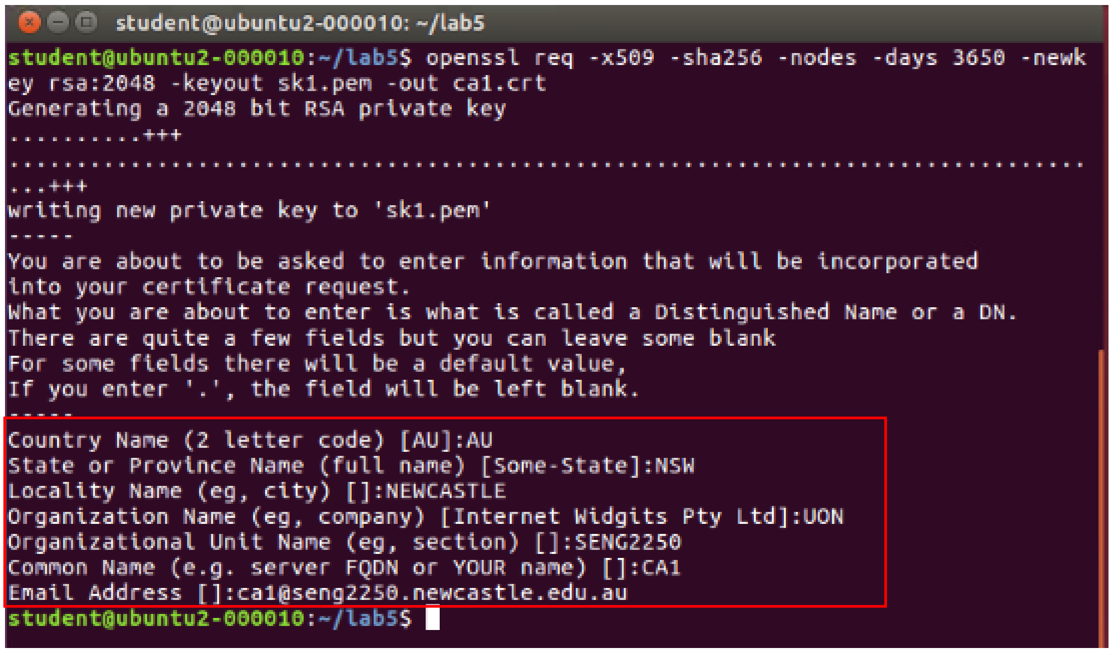
1. Create CA1's self-signed certificate:

   ```
   openssl req -x509 -sha256 -nodes -days 3650 -newkey rsa:2048 -keyout
   sk1.pem -out ca1.crt
   ```

   - `sk1.pem`: the file name of CA1's RSA private key. A generated private key will output to the file.
   - `ca1.crt`: the file name of CA1's self-signed certificate. A generated certificate will output to the file.

   You are required to fill out **CA1's identify information** (case-sensitive) as follows.

   - Write down the identity information (what you typed in), you may need it later for cross-certification generation.

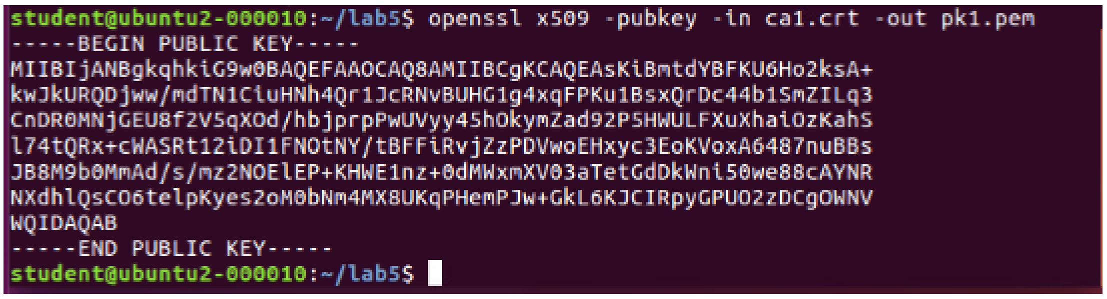


```
student@ubuntu2-000010: ~/lab5
student@ubuntu2-000010:~/lab5$ openssl req -x509 -sha256 -nodes -days 3650 -newk
ey rsa:2048 -keyout sk1.pem -out ca1.crt
Generating a 2048 bit RSA private key
..........+++
...............................................................................
...+++
writing new private key to 'sk1.pem'
-----
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) [AU]:AU
State or Province Name (full name) [Some-State]:NSW
Locality Name (eg, city) []:NEWCASTLE
Organization Name (eg, company) [Internet Widgits Pty Ltd]:UON
Organizational Unit Name (eg, section) []:SENG2250
Common Name (e.g. server FQDN or YOUR name) []:CA1
Email Address []:ca1@seng2250.newcastle.edu.au
student@ubuntu2-000010:~/lab5$
```

2. Extract CA1's public key from its certificate

```
openssl x509 -pubkey -in ca1.crt -out pk1.pem
```

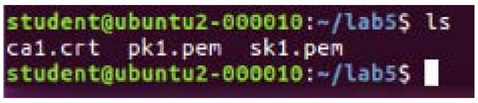
- `ca1.crt`: the file name of CA1's certificate.
- `pk1.pem`: the file name of CA1's public key. An extracted public key will output to the file.



```
student@ubuntu2-000010:~/lab5$ openssl x509 -pubkey -in ca1.crt -out pk1.pem
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAsKiBmtdYBFKU6Ho2ksA+
kwJkURQDjww/mdTN1CiuHNh4Qr1JcRNvBUHG1g4xqFPKu1BsxQrDc44b1SmZILq3
CnDR0MNjGEU8f2V5qXOd/hbjprpPwUVyy45hOkymZad92P5HWULFXuXhaiOzKahS
l74tQRx+cWASRt12iDI1FNOtNY/tBFFiRvjZzPDVwoEHxyc3EoKVoxA6487nuBBs
JB8M9b0MmAd/s/mz2NOElEP+KHWE1nz+0dMWxmXV03aTetGdDkWni50we88cAYNR
NXdhlQsCO6telpKyes2oM0bNm4MX8UKqPHemPJw+GkL6KJCIRpyGPUO2zDCgOWNV
WQIDAQAB
-----END PUBLIC KEY-----
student@ubuntu2-000010:~/lab5$ 
```

Now you should have **ca1.crt, pk1.pem** and **sk1.pem** files in "lab5". To check it, use command

```
ls
```



```
student@ubuntu2-000010:~/lab5$ ls
ca1.crt  pk1.pem  sk1.pem
student@ubuntu2-000010:~/lab5$ 
```

Create Alice's certificate from CA1

1. Generate Alice's RSA public and private keys

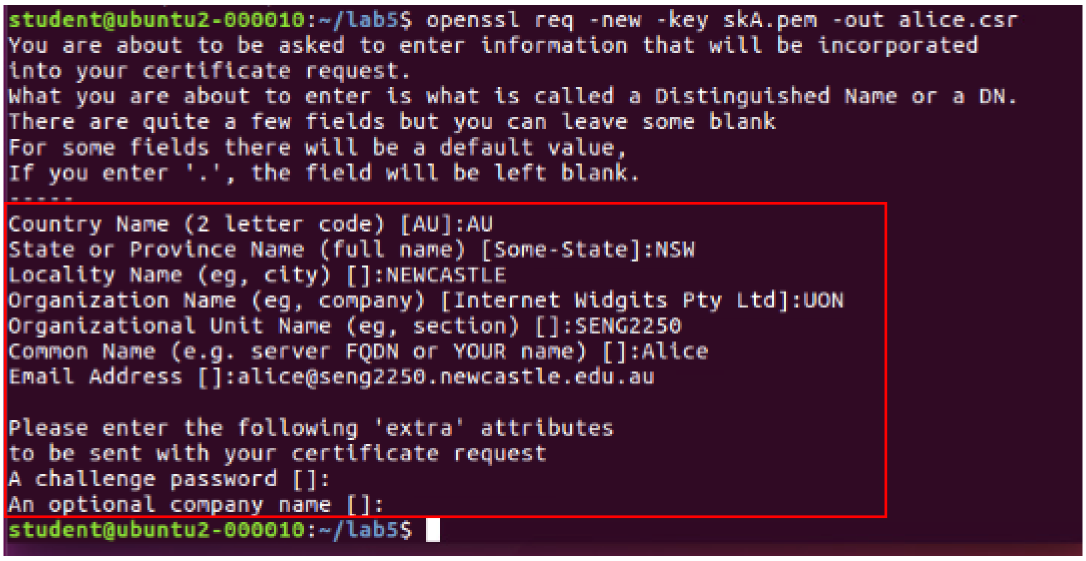
```
openssl genrsa -out skA.pem 2048
```

- `skA.pem`: the file name of Alice's private key. A generated private key will output to the file.
- What is the public key **e** showed on the screen? Check the value **e** later, when you generate public and private keys for other users. What's happened?

2. Generate Alice's Certificate Signing Request (CSR) file.
   A CSR file is to present the information of the requester's (here, Alice's) keys and identity

```
openssl req -new -key skA.pem -out alice.csr
```

- You will be required to type Alice's identity information (case-sensitive), see the figure below.
- Press **Enter** for "A challenge password []" and "An optional company name []" items.



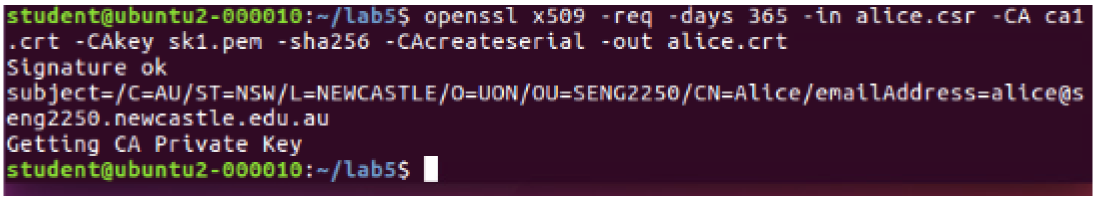
```
student@ubuntu2-000010:~/lab5$ openssl req -new -key skA.pem -out alice.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can 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) [AU]:AU
State or Province Name (full name) [Some-State]:NSW
Locality Name (eg, city) []:NEWCASTLE
Organization Name (eg, company) [Internet Widgits Pty Ltd]:UON
Organizational Unit Name (eg, section) []:SENG2250
Common Name (e.g. server FQDN or YOUR name) []:Alice
Email Address []:alice@seng2250.newcastle.edu.au

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
student@ubuntu2-000010:~/lab5$
```

3. Generate certificate for Alice by CA1

```
openssl x509 -req -days 365 -in alice.csr -CA ca1.crt -CAkey sk1.pem
-sha256 -CAcreateserial -out alice.crt
```

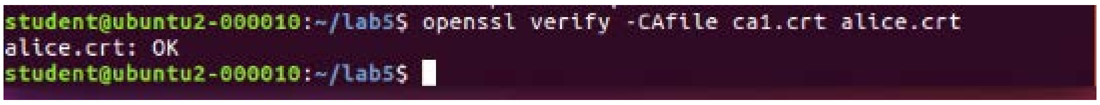
- `alice.csr`: Alice’s CSR file generated before.
- `ca1.crt`: CA1’s (self-signed) certificate.
- `sk1.pem`: CA1’s private key for signing.
- `alice.crt`: file name of Alice’s certificate. A generated certificate will output to the file.



```
student@ubuntu2-000010:~/lab5$ openssl x509 -req -days 365 -in alice.csr -CA ca1
.crt -CAkey sk1.pem -sha256 -CAcreateserial -out alice.crt
Signature ok
subject=/C=AU/ST=NSW/L=NEWCASTLE/O=UON/OU=SENG2250/CN=Alice/emailAddress=alice@s
eng2250.newcastle.edu.au
Getting CA Private Key
student@ubuntu2-000010:~/lab5$
```

4. Verify Alice’s certificate

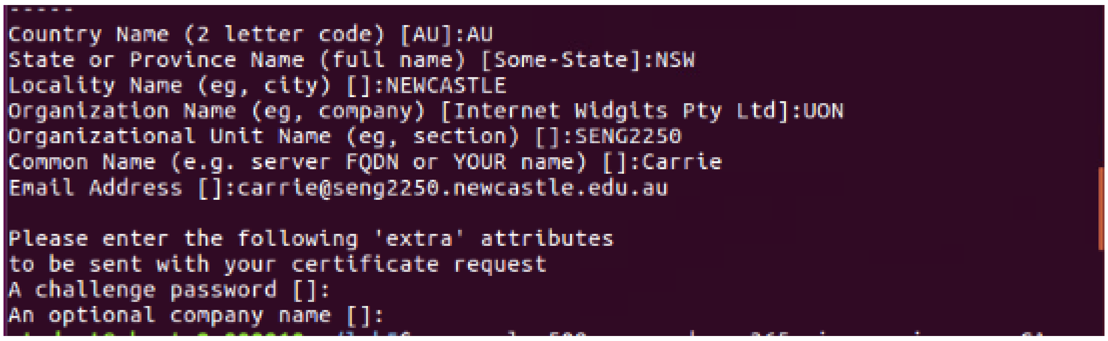
```
openssl verify -CAfile ca1.crt alice.crt
```



```
student@ubuntu2-000010:~/lab5$ openssl verify -CAfile ca1.crt alice.crt
alice.crt: OK
student@ubuntu2-000010:~/lab5$
```

Create Carrie’s certificate from CA1

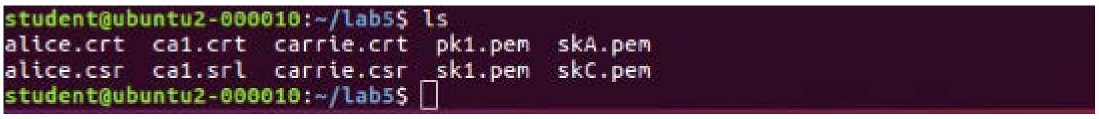
- Refer to the steps in Create Alice’s certificate from CA1 to generate a certificate for Carrie.
- Remember to change the **Common Name** and **Email Address** items accordingly using Carrie’s name, while keep other information the same as Alice. E.g.,



```
-----
Country Name (2 letter code) [AU]:AU
State or Province Name (full name) [Some-State]:NSW
Locality Name (eg, city) []:NEWCASTLE
Organization Name (eg, company) [Internet Widgits Pty Ltd]:UON
Organizational Unit Name (eg, section) []:SENG2250
Common Name (e.g. server FQDN or YOUR name) []:Carrie
Email Address []:carrie@seng2250.newcastle.edu.au

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

Now you should have files as follows:



```
student@ubuntu2-000010:~/lab5$ ls
alice.crt  ca1.crt  carrie.crt  pk1.pem  skA.pem
alice.csr  ca1.srl  carrie.csr  sk1.pem  skC.pem
student@ubuntu2-000010:~/lab5$
```

Create a self-signed certificate for CA2

Refer to the steps in Create a self-signed certificate for CA1 to setup CA2.

- Remember to change the **Common Name** and **Email Address** items accordingly using CA2’s name, while keep other information the same as CA1.
- Write down the identity information (what you typed in), you may need it later for cross-certification generation.

Create Bob’s certificate from CA2

- Refer to the steps in Create Alice’s certificate from CA1 to generate a certificate for Bob.
- Remember to change the **Common Name** and **Email Address** items accordingly using Bob’s name, while keep other information the same as Alice.

Cross-certification from CA1 to CA2

CA1 generates a certificate for CA2.

1. Generate CA2’s Certificate Signing Request (CSR) file.

   ```
   openssl req -new -key sk2.pem -out ca2.csr
   ```

   - `ca2.csr`: the name of CA2’s CSR file. A generated CSR will output to the file.
   - `sk2.pem`: CA2’s private key.
   - You MUST use CA2’s private key “sk2.pem” (generated before).
   - The CA2’s identify information MUST be the same as your input in Create a self-signed certificate for CA2.

2. Generate a certificate for CA2 by CA1

   ```
   openssl x509 -req -days 365 -in ca2.csr -CA ca1.crt -CAkey sk1.pem -sha256 -CAcreateserial -out ca1-2.crt
   ```

Verify Bob’s certificate

1. Use CA2’s certificate directly if a user trust it.

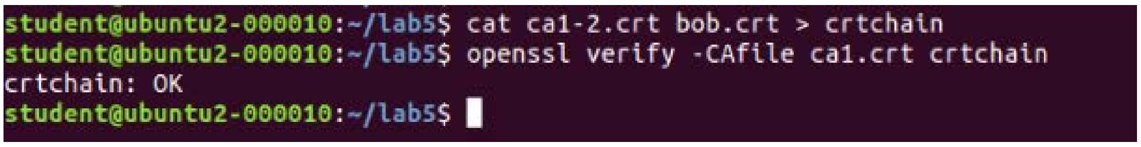
   ```
   openssl verify -CAfile ca2.crt bob.crt
   ```

2. Use cross-certification to verify if a user does not trust CA2 (i.e. you cannot use ca2.crt)
   - Combine all certificates in a certification path from CA1-2 (cross-certification) to Bob.

     ```
     cat ca1-2.crt bob.crt > crtchain
     ```

   - `crtchain`: combined certificates.
   - Verify Bob’s certificate using `crtchain`.

     ```
     openssl verify -CAfile ca1.crt crtchain
     ```



```
student@ubuntu2-000010:~/lab5$ cat ca1-2.crt bob.crt > crtchain
student@ubuntu2-000010:~/lab5$ openssl verify -CAfile ca1.crt crtchain
crtchain: OK
student@ubuntu2-000010:~/lab5$
```

Cross-certification from CA2 to CA1

Refer to the steps in Cross-certification from CA1 to CA2.

Verify Alice’s certificate in two ways.

Refer to the steps in Verify Bob’s certificate.

List of Commands

`<XXX>` indicates a file/directory name which may be changed in different cases. See the examples in the above steps.

- Create a directory.
  ```
  mkdir <XXX>
  ```
- Change directory.
  ```
  cd <XXX>
  ```
- Create self-signed certificate.
  ```
  openssl req -x509 -sha256 -nodes -days 3650 -newkey rsa:2048 -keyout
  <XXX> -out <XXX>
  ```
- Extract public key from a certificate.
  ```
  openssl x509 -pubkey -in <XXX> -out <XXX>
  ```
- List files.
  ```
  ls
  ```
- Generate an RSA private key.
  ```
  openssl genrsa -out <XXX> 2048
  ```
- Generate a CSR file.
  ```
  openssl req -new -key <XXX> -out <XXX>
  ```
- Generate a certificate by a CA.
  ```
  openssl x509 -req -days 365 -in <XXX> -CA <XXX> -CAkey <XXX> -sha256
  -CAcreateserial -out <XXX>
  ```
- Verify a certificate.
  ```
  openssl verify -CAfile <ca-certificate> <certificate>
  ```
- Combine two certificates.
  ```
  cat <certificate1> <certificate2> > <outputfile>
  ```

Lab environment settings

- OS: Ubuntu 16.04 LTS
- Software: OpenSSL 1.0.2g