TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

MÃ HÓA ỨNG DỤNG

LAB 03

CRYPTO PKI

Giảng viên: Trần Minh Triết

Lớp: 19TN

Người thực hiện:

Họ và tên	MSSV
Đặng Thái Duy	19120491
Hà Chí Hào	19120219

MỤC LỤC

I. Cấu trúc chung

Trong thư mục Source gồm có các file quan trọng sau đây:

- Thư mục Task chứa mã nguồn theo từng thư mục Task tương ứng
- Thư mục Labsetup được cung cấp sẵn
- Makefile: chứa lệnh để chạy các file .sh tương ứng theo từng task. Để chạy task chỉ cần gõ lệnh sau: make <task_name>

II. Task

1. Task 1: Becoming a Certificate Authority (CA)

Trong Task này, ta cần tạo ra một chứng chỉ kỹ thuật số. Ở đây ta sẽ tự tạo ra một root CA và sau đó sử dụng CA này để cấp chứng chỉ cho những nơi khác sử dụng như là server. Chứng chỉ của root CA là tự ký và được xem là hoàn toàn đáng tin cậy

Trước hết, ta cần một file cấu hình để có thể dùng OpenSSL để tạo ra chứng chỉ. Và ở SeedLab đã có sẵn file này là openssl.cnf, ta copy nó sang thư mục làm việc hiện tại bằng lệnh sau

```
cp /usr/lib/ssl/openssl.cnf ./openssl.cnf
```

Tiếp theo ta sẽ tạo ra thư mục mặc định để làm việc và dùng cho các task sau, cụ thể như sau:

- Thư mục demoCA: nơi cấu trúc file cho CA được thiết lập
- Thư mục certs: nơi lưu trữ các chứng chỉ đã cấp
- Thư mục crl: nơi lưu trữ các crl đã cấp
- Thư mục newcerts: nơi lưu trữ các chứng chỉ mới
- index.txt: lưu các cơ sở dữ liệu
- serial: số seri hiện tại

```
mkdir demoCA
cd demoCA
mkdir certs crl newcerts
touch index.txt
echo "1000" > serial
```

Bây giờ ta sẽ tạo ra chứng chỉ tự ký cho CA, tức là CA này hoàn toàn đáng tin cậy và chứng chỉ này sẽ đóng vai trò là chứng chỉ gốc. Ta có thể tạo được chứng chỉ này bằng câu lệnh dưới đây,

ngoài ra ta thêm vào trường thông tin subject (-subj) và mật khẩu (-passout pass) để bỏ qua việc nhập dữ liệu mỗi khi chạy chương trình

```
# Subject input
# PEM pass phrase               = 123456
# Verify PEM pass phrase        = 123456
# Country Name                  = VN
# State Or Province Name        = HCM
# Locality Name                 = HCM
# Organization Name             = HCMUS
# Organizational Unit Name      = HCMUS
# Common Name                   = www.hcmusCA.com
openssl req -new -x509 -keyout ca.key -out ca.crt -config openssl.cnf \
        -subj "/C=VN/ST=HCM/L=HCM/O=HCMUS/OU=HCMUS/CN=www.hcmusCA.com" \
        -passout pass:123456
```

Kết quả sẽ được lưu vào 2 file ca.key và ca.crt, trong đó ca.key lưu private key của CA và ca.crt lưu chứng chỉ public key. Cụ thể như sau

ca.crt

```
-----BEGIN CERTIFICATE-----
MIIDpzCCAo+gAwIBAgIURvarSYGWTzyyJ/zHXFs0TdDExWYwDQYJKoZIhvcNAQEL
BQAwYzELMAkGA1UEBhMCVk4xDDAKBgNVBAgMA0hDTTEMMAoGA1UEBwwDSENNMQ4w
DAYDVQQKDAVIQ01VUzEOMAwGA1UECwwFSENNVVMxGDAWBgNVBAMMD3d3dy5oY211
c0NBLmNvbTAeFw0yMjA1MjUxNzE0MTdaFw0yMjA2MjQxNzE0MTdaMGMxCzAJBgNV
BAYTAlZOMQwwCgYDVQQIDANIQ00xDDAKBgNVBAcMA0hDTTEOMAwGA1UECgwFSENN
VVMxDjAMBgNVBAsMBUhDTVVTMRgwFgYDVQQDDA93d3cuaGNtdXNDQS5jb20wggEi
MA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDOnX0FtZ1Zb7BmgragQuxuwajs
Y2zmi6gamveeKBUGB/uCErCNcqDvFsluEo5SxqnvQa8CLUQC7yFh+1F5iMvUY4bj
K3BXdJL8PWLzbsasRcFnym73TeZdnoy4LGJRAz2garO7WWn2vfuAsiPdWRPDWPEt
u8mC3JV8v2faNGWOfBVw0KwW950fdRv6YH1JgGBGdxH+pQHw32wlAEsq0MfGpzCF
+X0R1noC9yz835dIAcCDlNZ1YSNz7vp3PFolw//8x86f3bg4tcCeyp+QLSj+bQG4
YuWG9Fr/u12ijZAf7ycS3UFI5hiXL2pMRmNoHYcBtpiE9qlvgacIBTnTaXt3AgMB
AAGjUzBRMB0GA1UdDgQWBBQjUZqNtjMNFNovO3YWHyhGmZ+fkDAfBgNVHSMEGDAW
gBQjUZqNtjMNFNovO3YWHyhGmZ+fkDAPBgNVHRMBAf8EBTADAQH/MA0GCSqGSIb3
DQEBCwUAA4IBAQBl/YwirfMO1QWfjH4tPTC6kedgAu/yYYH0ydw14m4h6X5DjOOh
Ln95sz87YKjT/wgGqiDkoT92aBkf3Wyke6rTviFXB3yWSQZXImWl76nCsRqnyCwG
reLjj+ndUpvxNKVHIcvT4V2cWt7OgsseRzXGsi2JraDT682Q9hDsV3zq75NvJaB9
xrIEWriOBIqlC4HzEbWweGHN+WPtbSgwI1dgkGSMUPnUFWqeOdboZWIT+U8A8MAs
w7oMN5w78sqsNRrz/0yThuukZrVqQ7YJGRyiezVJzxIoF9m1kZFoesYjVtJG5IVA
VKISPKj57tuJ11wv3B5Jjc/0qJAmenc2bg1K
-----END CERTIFICATE-----
```

ca.crt

ca.key

```
-----BEGIN ENCRYPTED PRIVATE KEY-----
MIIFHDBOBgkqhkiG9w0BBQ0wQTApBgkqhkiG9w0BBQwwHAQII3ycer5C06gCAggA
MAwGCCqGSIb3DQIJBQAwFAYIKoZIhvcNAwcECGgLttJbzEAhBIIEyKkGYEN0l5Z6
mb+y46sa0SzzqekKu2Mm1ERyRGsrFv3z6b9tzOwFjnoCLimqFbM43+lIKg4pj4oA
zHJ9X7viwoggZlBdTuMhok4Y/YB6sFmB+8/R5HsZI/vWmuZxqxWh4kb3RjLnE+VO
ySsMIy/MnPsEofJ/c03JSnMuz0/eiGjL/ls8H/2XDhcUfUAHRiCaOkpsy3c6Yoj+
PBIOJyJfKybcPd91JIdFGUpNXL2vH2dbivBKZZ69v4Rcn2QAD/9v34KBuhN9YZmP
AQM2DozP9qo0RrpvbIEAVk/pqxbeVziHX0jkgrtwt5Yztgjbpvdj767XeKNYm0Z+
m70QfQd01xt03SxbJzQFQn4g4I8++4Z0yPngAxLlKWhm4b6FnjT2m3OQZ3jKCSVJ
5oYzP+HbjMFe5DDpjs692PHTHdrAM5GcxcVD5YjaOUg55vYCzgGBPJdl0hTEYvCs
Q5IlNYUY4uEgYAyGP0iMEUbwXOMH9O7ZrSuy0gIOzeJalUuFNJ78mKWmzI8s1Ika
9G275HS1lqE2IAG0jAYOh4w33S7Q/ONpl2EdxBDenCCp3yPpc2pR68xoeFRDq4bm
CffZdRz+NHd/satQoAZN9h3ICyQhDW6crcSD1bsKOdblvmg70vaVXbKhpqRSdDxY
251JBCQgYW9WOw2vgOSUV+NLR/Wl48XtvEJRDoRa2PSB1V3YrpRQ1eE2Q1rdgM1H
CzTNWL21okR4KeUpyJLrzEwJ9WImucDn7+WioloekyPs9f02l42GhGTXWVJiVJlV
EoAfJu+L3beoeRsn7o9+bSLkhhmrhmVmM3CZMw7wnJOj3c/ivTmDm1owbI9Ijhij
9ffgOXv/6bXnxdFBUZgJ6kZRA4WM+yW2jb9jgCo8qYTyibbI36aBstOsnd8J6IAe
BLmsLiPrKmQR5kBqEJYg47+MCkWo7NxaxhLjYa5dHXIOsIN/pgIGXlIXilQLRBk4
F5FkgAFvsc8e/1qe0xdVXI8W1mCUWnUrEaxEAbwBSfjgy7l46f84O5U51jHe5FmI
pZwAK8QRf/FVGL0qayDnmMkretMmhmaKjz5x0oAxp+DCPBsYPD4hLQ1f5UR1NAla
bdNbXD86MRz+tD3vkKgo3+6s1r2BLFj+maoPE/FQv5ALIvuzNl2hpQ/3YI5R6ufT
7yZ5gOsmhtGzdpsOgkvXIwaaetpd/0sRzt50Z5/RnCr+mWiuL9Yw7BlrTUUfOb2r
cnyFaHPk5/bwGZOc3lxI7YP28RW0pEzpdu2O4pGh05SEV4SyXDAeu/tEz4WF0pQS
CoNjH4D6Y3+qB/hfXEZD8GpX2gNIzul6U8Na+fw7/kxSLEFLq+eXtkeSCFr5plm8
vl39Q+Jifq3p6OfeioRAReW9VXfSvSYbvZSReVxzzMNeXZaYHeiApjRAW/LUXkq+
ac/evv/1K/yCNHCkFiUb/Yi9IHKWefbJc8970+6fmtuexUhlISCHkjSxSf8oLiwK
luVVsMWnXeVi1c/3sURwQRH8fOmGkb8grk3t/LkhtrLQE4lp1QeQQtoJ0hmMzcpA
IXmQtl9fkyroY+TjRuw2YwxNWi9A203U7Hj6i1ApSy3TQJMDZ2VJLo0Sn5GBihmG
QLZqvdl3I81ToYVCnjJ/8g==
-----END ENCRYPTED PRIVATE KEY-----
```

ca.key

Để xem nội dung được giải mã của chứng chỉ X509 và khóa RSA, ta dùng câu lệnh sau đây để hiển thị toàn bộ thông tin, kết quả tương ứng sẽ lần lượt được lưu vào 2 file ***ca.crt.txt*** và ***ca.key.txt***

```
# decoded content of X509 certificate and RSA key
openssl x509 -in ca.crt -out ca.crt.txt -text
openssl rsa  -in ca.key -out ca.key.txt -text -passin pass:123456
```

Từ kết quả nhận được, ta rút ra các nhận xét và thông số sau:

- Phần nhận diện đây là một chứng chỉ CA

```
        X509v3 Basic Constraints: critical
            CA:TRUE
```

- Phần nhận diện đây là một chứng chỉ tự ký: Subject Key Identifier và Authority Key Identifier giống nhau

```
X509v3 extensions:
    X509v3 Subject Key Identifier:
        23:51:9A:8D:B6:33:0D:14:DA:2F:3B:76:16:1F:28:46:99:9F:9F:90
    X509v3 Authority Key Identifier:
        keyid:23:51:9A:8D:B6:33:0D:14:DA:2F:3B:76:16:1F:28:46:99:9F:9F:90
```

- Các thông số khi mã hóa bằng RSA:
  - public exponent e:

```
publicExponent: 65537 (0x10001)
```

  - private exponent d:

```
privateExponent:
    4b:2a:5c:2b:51:90:f6:d2:7d:18:2e:01:86:78:12:
    4f:90:c0:85:8e:fe:35:39:25:64:d8:6f:b0:e7:e0:
    0e:1b:a5:52:02:be:9c:16:d1:99:69:6e:ca:4e:91:
    dc:67:4d:b7:2e:ec:1e:1c:9d:9b:7b:a7:67:5f:e8:
    9a:10:3c:26:fe:36:3a:3a:b9:59:f7:9c:e9:8a:e3:
    8e:b0:04:32:f4:05:92:5e:c0:d3:d3:51:35:49:d2:
    e3:c5:b1:d6:cb:19:06:5d:0b:19:e3:43:7b:72:4e:
    26:d9:a7:d6:84:eb:7d:ab:c3:c1:e1:89:43:c4:3f:
    d5:cc:44:39:cb:ca:0c:75:4a:73:23:06:78:2c:cf:
    20:ae:97:83:6b:39:df:5e:8f:0d:a8:12:90:5b:17:
    c6:22:2e:47:0b:78:09:83:73:c4:88:c4:fc:38:72:
    08:90:9f:a2:f6:85:f8:a7:10:17:9d:f2:ef:13:03:
    ed:a2:90:bf:0e:31:57:8d:54:3d:14:d7:f9:c5:24:
    d2:65:0b:3a:b3:1b:02:f1:11:34:e0:fe:07:95:f8:
    5f:3d:6b:7a:ad:56:e3:f4:79:27:3a:b5:19:01:b4:
    b5:56:c7:52:d2:5e:a2:ef:68:97:a4:80:c1:b4:cb:
    82:5b:a5:71:20:81:c8:d9:c3:a0:0a:ff:f1:e9:0a:
    91
```

  - modulus n:

```
modulus:
    00:ce:9d:7d:05:b5:9d:59:6f:b0:66:82:b6:a0:42:
    ec:6e:c1:a8:ec:63:6c:e6:8b:a8:1a:9a:f7:9e:28:
    15:06:07:fb:82:12:b0:8d:72:a0:ef:16:c9:6e:12:
    8e:52:c6:a9:ef:41:af:02:2d:44:02:ef:21:61:fb:
    51:79:88:cb:d4:63:86:e3:2b:70:57:74:92:fc:3d:
    62:f3:6e:c6:ac:45:c1:67:ca:6e:f7:4d:e6:5d:9e:
    8c:b8:2c:62:51:03:3d:a0:6a:b3:bb:59:69:f6:bd:
    fb:80:b2:23:dd:59:13:c3:58:f1:2d:bb:c9:82:dc:
    95:7c:bf:67:da:34:65:8e:7c:15:70:d0:ac:16:f7:
    9d:1f:75:1b:fa:60:7d:49:80:60:46:77:11:fe:a5:
    01:f0:df:6c:25:00:4b:2a:d0:c7:c6:a7:30:85:f9:
    7d:11:d6:7a:02:f7:2c:fc:df:97:48:01:c0:83:94:
    d6:75:61:23:73:ee:fa:77:3c:5a:25:c3:ff:fc:c7:
    ce:9f:dd:b8:38:b5:c0:9e:ca:9f:90:2d:28:fe:6d:
    01:b8:62:e5:86:f4:5a:ff:bb:5d:a2:8d:90:1f:ef:
    27:12:dd:41:48:e6:18:97:2f:6a:4c:46:63:68:1d:
    87:01:b6:98:84:f6:a9:6f:81:a7:08:05:39:d3:69:
    7b:77
```

- 2 secret number p và q

```
prime1:
    00:e9:d9:98:7a:b8:f4:c0:7a:6b:fc:22:3f:7f:bd:
    4c:0c:3c:6a:60:db:c0:70:3c:08:cc:fc:d7:0d:ca:
    9f:79:c6:ed:6e:22:6f:9d:91:d1:7e:d6:3f:82:34:
    0d:36:3f:a8:f0:f0:2c:0a:67:9c:f9:12:b4:17:80:
    16:7d:c4:29:e6:b4:45:01:56:1e:26:5c:1f:a0:28:
    43:87:26:ea:cf:e6:a4:1d:f8:39:f1:d6:a7:8b:33:
    40:94:82:1e:78:2c:09:c9:ec:ab:89:8a:9d:a8:c7:
    ad:a0:ce:19:a0:50:fd:f7:95:e7:a5:b4:a8:bd:c1:
    9c:e5:0d:a9:9f:c3:f7:08:7b
prime2:
    00:e2:2f:80:96:9b:85:69:fe:cd:8d:2c:00:4a:c3:
    ab:40:2c:3f:27:b7:0c:f0:b2:85:03:6e:3c:5c:b2:
    61:bc:e4:fb:dd:17:bd:2c:63:e1:6a:b8:00:2d:5c:
    0a:92:02:55:63:f4:7f:3f:b5:a6:76:41:07:83:ca:
    d1:1f:b8:b6:95:97:ff:90:48:8c:13:d4:50:a8:c3:
    31:0b:6b:67:8d:ce:9e:d2:36:da:58:c4:e2:b0:37:
    8f:32:3a:24:03:b4:83:f5:28:4d:e2:ac:ef:47:93:
    0e:21:1e:af:c0:e5:cc:fa:18:58:63:86:b0:8b:4e:
    c3:f5:f2:e6:3e:1e:0a:0e:35
```

2. Task 2: Generating a Certificate Request for Your Web Server

Trong task này ta cần tạo một CSR (Certificate Signing Request), về cơ bản sẽ gồm thông tin nhận dạng và public key. CSR sẽ được gửi đến CA nhằm mục đích xác định thông tin nhận dạng trong yêu cầu và sau đó tạo ra chứng chỉ.

Lệnh để tạo CSR khá giống với lệnh ta dùng ở task 1 để tạo chứng chỉ CA tự ký, điểm khác biệt là tùy chọn -x509. Ở đây ngoài 2 thông số *-subj* và *-passout* thì còn thêm trường *-addtext* để thêm vào các tên miền thay thế sẽ được dùng ở task sau

```
# generate a Certificate Signing Request (CSR)
openssl req -new -keyout server.key -out server.csr -config openssl.cnf \
        -subj "/C=VN/ST=HCM/L=HCM/O=HCMUS/OU=HCMUS/CN=www.duy2022.com" \
        -passout pass:123456 \
        -addext "subjectAltName = DNS:www.duy2022.com, DNS:www.duy2022a.com,
DNS:www.duy2022b.com"
```

Để xem nội dung được giải mã của CSR và private key, ta dùng lệnh sau đây để hiển thị toàn bộ thông tin, kết quả tương ứng sẽ lần lượt được lưu vào 2 file server.crt.txt và server.key.txt

```
# decoded content of CSR and private key
openssl req -in server.csr -out server.csr.txt -text
openssl rsa -in server.key -out server.key.txt -text -passin pass:123456
```

Kết quả ra được

server.csr.txt

```
Certificate Request:
    Data:
        Version: 1 (0x0)
        Subject: C = VN, ST = HCM, L = HCM, O = HCMUS, OU = HCMUS, CN = www.duy2022.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:d9:25:d5:63:c2:f3:83:82:27:10:69:e3:f5:00:
                    51:b1:7e:77:4e:a0:8e:81:69:85:83:5c:86:d0:6f:
                    68:59:ff:01:97:cb:ae:5d:fe:cf:40:82:e0:d0:b9:
                    ce:08:43:93:2b:8d:c8:bb:57:43:1a:e1:bd:1a:da:
                    50:ad:df:73:be:b4:94:f3:65:ba:1c:58:c4:50:bf:
                    77:2d:e2:6a:37:ce:a6:62:63:e6:c3:86:b1:1a:dc:
                    b5:a8:4d:f6:e8:76:69:ec:b7:af:f3:6b:8f:91:06:
                    2d:d8:85:d9:59:66:e8:63:e8:3b:3d:c5:df:c1:56:
                    15:5a:7c:87:a5:b8:6c:86:55:fe:b4:d1:7d:63:42:
                    a8:93:14:7b:3c:95:a6:59:c6:5d:c7:9d:b8:45:64:
                    56:1a:ed:e4:14:78:72:88:e1:4f:dc:b8:bf:ae:e4:
                    7d:2e:34:0a:2c:99:18:f6:83:86:9e:ef:23:9f:b3:
                    b8:cb:c9:c0:43:92:15:3c:2e:e4:ce:0d:56:ef:41:
                    2b:d0:bd:7e:b7:1a:23:0d:68:dd:57:c2:ce:f6:ab:
                    0f:dd:6b:e6:3f:cc:da:35:b7:54:13:18:84:8f:1a:
                    46:92:ba:5c:23:55:52:05:66:ae:b6:4e:1d:e1:2e:
                    23:ef:8b:28:20:db:0c:b2:e3:bf:7e:c4:71:50:a0:
                    4f:65
                Exponent: 65537 (0x10001)
        Attributes:
        Requested Extensions:
            X509v3 Subject Alternative Name:
                DNS:www.duy2022.com, DNS:www.duy2022a.com, DNS:www.duy2022b.com
    Signature Algorithm: sha256WithRSAEncryption
         68:68:1e:52:db:11:1f:f9:21:c6:72:b3:1f:91:e0:27:fb:a6:
```

server.csr.txt

server.key.txt

```
RSA Private-Key: (2048 bit, 2 primes)
modulus:
    00:d9:25:d5:63:c2:f3:83:82:27:10:69:e3:f5:00:
    51:b1:7e:77:4e:a0:8e:81:69:85:83:5c:86:d0:6f:
    68:59:ff:01:97:cb:ae:5d:fe:cf:40:82:e0:d0:b9:
    ce:08:43:93:2b:8d:c8:bb:57:43:1a:e1:bd:1a:da:
    50:ad:df:73:be:b4:94:f3:65:ba:1c:58:c4:50:bf:
    77:2d:e2:6a:37:ce:a6:62:63:e6:c3:86:b1:1a:dc:
    b5:a8:4d:f6:e8:76:69:ec:b7:af:f3:6b:8f:91:06:
    2d:d8:85:d9:59:66:e8:63:e8:3b:3d:c5:df:c1:56:
    15:5a:7c:87:a5:b8:6c:86:55:fe:b4:d1:7d:63:42:
    a8:93:14:7b:3c:95:a6:59:c6:5d:c7:9d:b8:45:64:
    56:1a:ed:e4:14:78:72:88:e1:4f:dc:b8:bf:ae:e4:
    7d:2e:34:0a:2c:99:18:f6:83:86:9e:ef:23:9f:b3:
    b8:cb:c9:c0:43:92:15:3c:2e:e4:ce:0d:56:ef:41:
    2b:d0:bd:7e:b7:1a:23:0d:68:dd:57:c2:ce:f6:ab:
    0f:dd:6b:e6:3f:cc:da:35:b7:54:13:18:84:8f:1a:
    46:92:ba:5c:23:55:52:05:66:ae:b6:4e:1d:e1:2e:
    23:ef:8b:28:20:db:0c:b2:e3:bf:7e:c4:71:50:a0:
    4f:65
publicExponent: 65537 (0x10001)
privateExponent:
    00:cf:8f:15:48:35:49:34:53:cc:e9:28:7e:27:6a:
    75:ab:d5:f9:4e:63:b3:b3:49:5c:c0:32:49:fe:a3:
```

server.key.txt

3. Task 3: Generating a Certificate for your server

CSR thường được gửi đến một CA đáng tin cậy để xin chữ ký xác thực. Trong task này, ta sẽ dùng CA đáng tin cậy của mình để tạo ra chứng chỉ

Câu lệnh sau đây dùng để tạo ra chứng chỉ được lưu vào file ***server.crt***

```
# generating certificate
openssl ca -in server.csr -out server.crt -cert ca.crt -keyfile ca.key -config openssl.cnf \
        -passin pass:123456 -batch
```

Sau khi ký chứng chỉ, ta sử dụng lệnh sau để in ra nội dung đã giải mã

```
# decoded content of X509 certificate
openssl x509 -in server.crt -out server.crt.txt -text
```

Kết quả ra được trong file ***server.crt.txt*** như dưới đây

server.crt.txt

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 4096 (0x1000)
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = VN, ST = HCM, L = HCM, O = HCMUS, OU = HCMUS, CN = www.hcmusCA.com
        Validity
            Not Before: May 26 11:23:39 2022 GMT
            Not After : May 26 11:23:39 2023 GMT
        Subject: C = VN, ST = HCM, O = HCMUS, OU = HCMUS, CN = www.duy2022.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:d9:25:d5:63:c2:f3:83:82:27:10:69:e3:f5:00:
                    51:b1:7e:77:4e:a0:8e:81:69:85:83:5c:86:d0:6f:
                    68:59:ff:01:97:cb:ae:5d:fe:cf:40:82:e0:d0:b9:
                    ce:08:43:93:2b:8d:c8:bb:57:43:1a:e1:bd:1a:da:
                    50:ad:df:73:be:b4:94:f3:65:ba:1c:58:c4:50:bf:
                    77:2d:e2:6a:37:ce:a6:62:63:e6:c3:86:b1:1a:dc:
                    b5:a8:4d:f6:e8:76:69:ec:b7:af:f3:6b:8f:91:06:
                    2d:d8:85:d9:59:66:e8:63:e8:3b:3d:c5:df:c1:56:
                    15:5a:7c:87:a5:b8:6c:86:55:fe:b4:d1:7d:63:42:
                    a8:93:14:7b:3c:95:a6:59:c6:5d:c7:9d:b8:45:64:
                    56:1a:ed:e4:14:78:72:88:e1:4f:dc:b8:bf:ae:e4:
                    7d:2e:34:0a:2c:99:18:f6:83:86:9e:ef:23:9f:b3:
                    b8:cb:c9:c0:43:92:15:3c:2e:e4:ce:0d:56:ef:41:
                    2b:d0:bd:7e:b7:1a:23:0d:68:dd:57:c2:ce:f6:ab:
                    0f:dd:6b:e6:3f:cc:da:35:b7:54:13:18:84:8f:1a:
                    46:92:ba:5c:23:55:52:05:66:ae:b6:4e:1d:e1:2e:
                    23:ef:8b:28:20:db:0c:b2:e3:bf:7e:c4:71:50:a0:
                    4f:65
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
```

4. Task 4: Deploying Certificate in an Apache-Based HTTPS Website

Bắt đầu từ task này, thì ta phải sử dụng docker, cụ thể là container được gửi trong file Labsetup.zip của SEEDLab. Cho nên ta cần phải setup container đó trước, ta phải vào folder Labsetup (được giải nén từ Labsetup.zip), rồi sau đó gõ lệnh dưới đây để xây dựng container:

```
docker-compose build
```

Và lệnh dưới đây để chạy container:

```
docker-compose up
```

Để chạy lệnh bên trong 1 container đang chạy, trước tiên ta phải biết được ID của container đó, ta có thể dùng lệnh sau để biết được thông tin của các container đang chạy:

```
docker ps
```

ID của container sẽ nằm ở cột đầu tiên CONTAINER ID, sử dụng ID vào lênh sau đây để bắt đầu chạy lệnh trong container:

```
docker exec -it <id> /bin/bash
```

Trong máy ảo của riêng SEEDLab có sẵn các alias để có thể viết các lệnh trên ngắn hơn, nhưng em giữ như cũ để mang tính tổng quát.

Giờ em sẽ bắt đầu thực hiện yêu cầu của task, task này ta cần phải tạo một Apache-based HTTPS Website với đầy đủ certificate. Sau khi xem qua ví dụ web www.bank32.com của file đề, em rút được các bước để thực hiện được điều đó như sau:

- Chuẩn bị các config (vị trí tài nguyên, domain website, vị trí certificate, …) của website cho Apache biết trong 1 file .conf.
- Setup tài nguyên cho website (Phần DocumentRoot trong file .conf).
- Setup certificate và private key cho server của website.
- Enable website bằng lệnh `a2ensite`.
- Setup DNS
- Nạp certificate của CA mà cung cấp certificate cho website cho browser để browser có thể tin tưởng website của ta.

4.1. Chuẩn bị config

Để chuẩn bị config cho website, em vào thư mục `/etc/apache2/sites-available` của thư mục, copy file `bank32_apache_ssl.conf` và paste thành file `duy2022_apache_ssl.conf`, và sửa thông tin file thành như sau:

```
<VirtualHost *:443>
    DocumentRoot /var/www/duy2022
    ServerName www.duy2022.com
    ServerAlias www.duy2022a.com
```

```
    ServerAlias www.duy2022b.com
    DirectoryIndex index.html
    SSLEngine On
    SSLCertificateFile /certs/server.crt
    SSLCertificateKeyFile /certs/server.key
</VirtualHost>

<VirtualHost *:80>
    DocumentRoot /var/www/duy2022
    ServerName www.duy2022.com
    DirectoryIndex index_red.html
</VirtualHost>

# Set the following gloal entry to suppress an annoying warning message
ServerName localhost
```

Vì file certificate request và certificate được tạo ra từ Task 2, Task 3 chúng em sử dụng trang www.duy2022.com, nên file .conf cũng phải config như vậy.

4.2. Setup tài nguyên

Tài nguyên của website chỉ đơn giản là các file HTML, CSS và Javascript nếu có. Cho thuận tiện thì em sẽ copy lại các file tài nguyên từ bank32 bằng cách vào folder `/var/www` và copy folder `bank32` và paste thành `duy2022` (trùng với `DocumentRoot` của file config).

4.3. Setup certificate và private key

Certificate và private key thì em sử dụng lại từ task 2, 3, nhưng 2 file đó đang ở bên ngoài máy ảo, chứ không bên trong container. Vì vậy SEEDLab đã setup trước folder `volumes` trong thư mục Labsetup để ta có thể chuyển tập tin giữa máy ảo và container. Các tập tin, thư mục xuất hiện ở thư mục `Labsetup/volumes` đều xuất hiện ở thư mục `/volumes` của container.

Như vậy em chỉ cần copy 2 file certificate và private key vào trong `Labsetup/volumes`, và trong container em chỉ cần di chuyển lại 2 file đó từ `/volumes` vào `/certs` (vị trí được định nghĩa trong file config)

4.4. Enable website

Sau khi đã có đủ tài nguyên và certificate, giờ ta có thể enable website của chúng ta bằng cách vào lại trong thư mục `/etc/apache2/sites-available` và chạy lệnh:

```
a2ensite duy2022_apache_ssl
```

4.5. Setup DNS

Để có thể truy cập được trang web, thì ta phải setup DNS để có thể dẫn từ domain www.duy2022.com về IP của server (10.9.0.80) bằng cách sửa file /etc/hosts:

```
sudo nano /etc/hosts
```

Ta thêm 1 dòng dưới đây vào /etc/hosts như sau:

```
10.9.0.80       www.duy2022.com
```

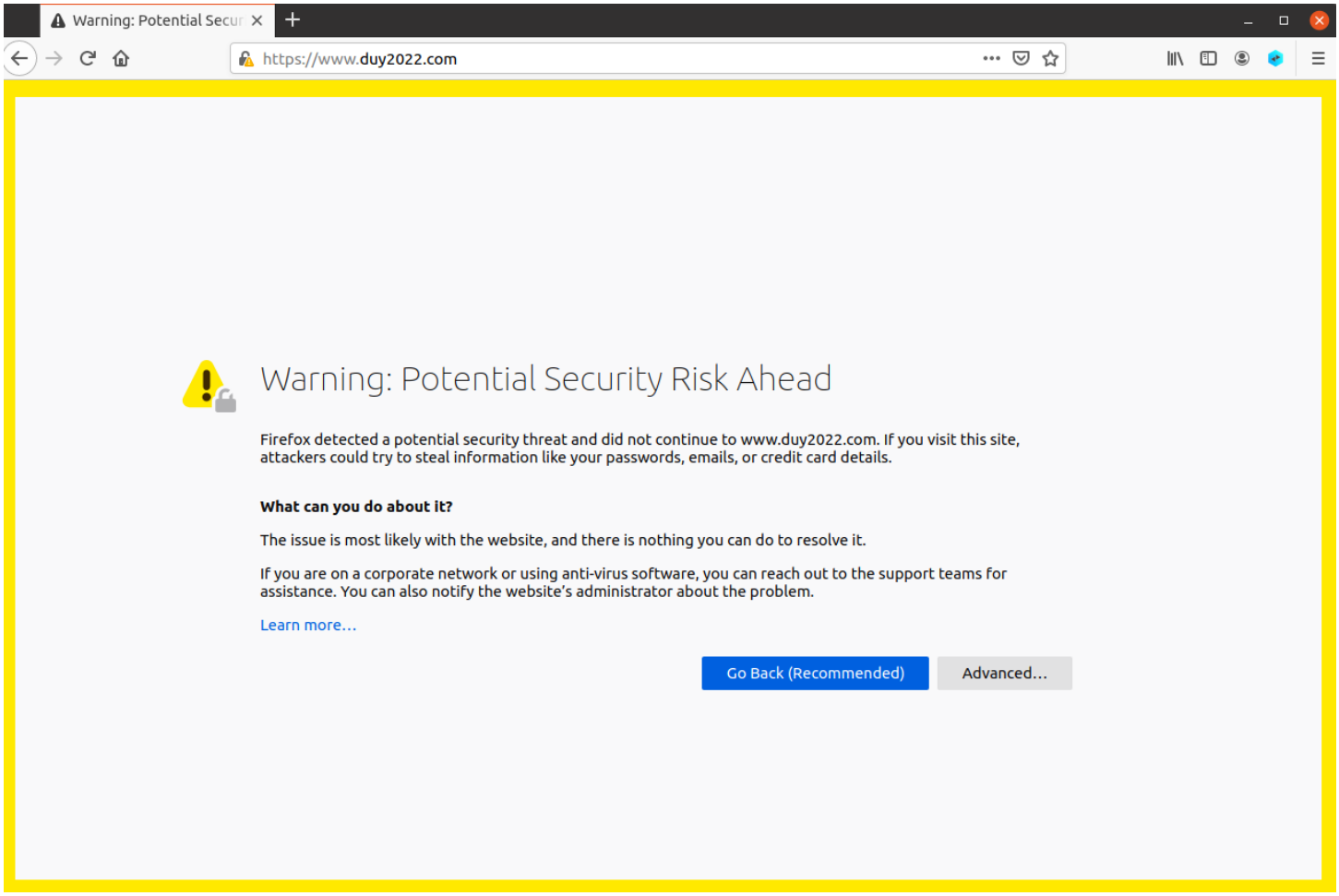
4.6. Nạp certificate của CA cho browser

Các việc ta cần làm gần như đã xong, nhưng nếu ta thử khởi chạy server và truy cập thử trang web thì ta sẽ gặp một sự cố. Để khởi chạy server ta dùng lệnh sau:

```
service apache2 start
```

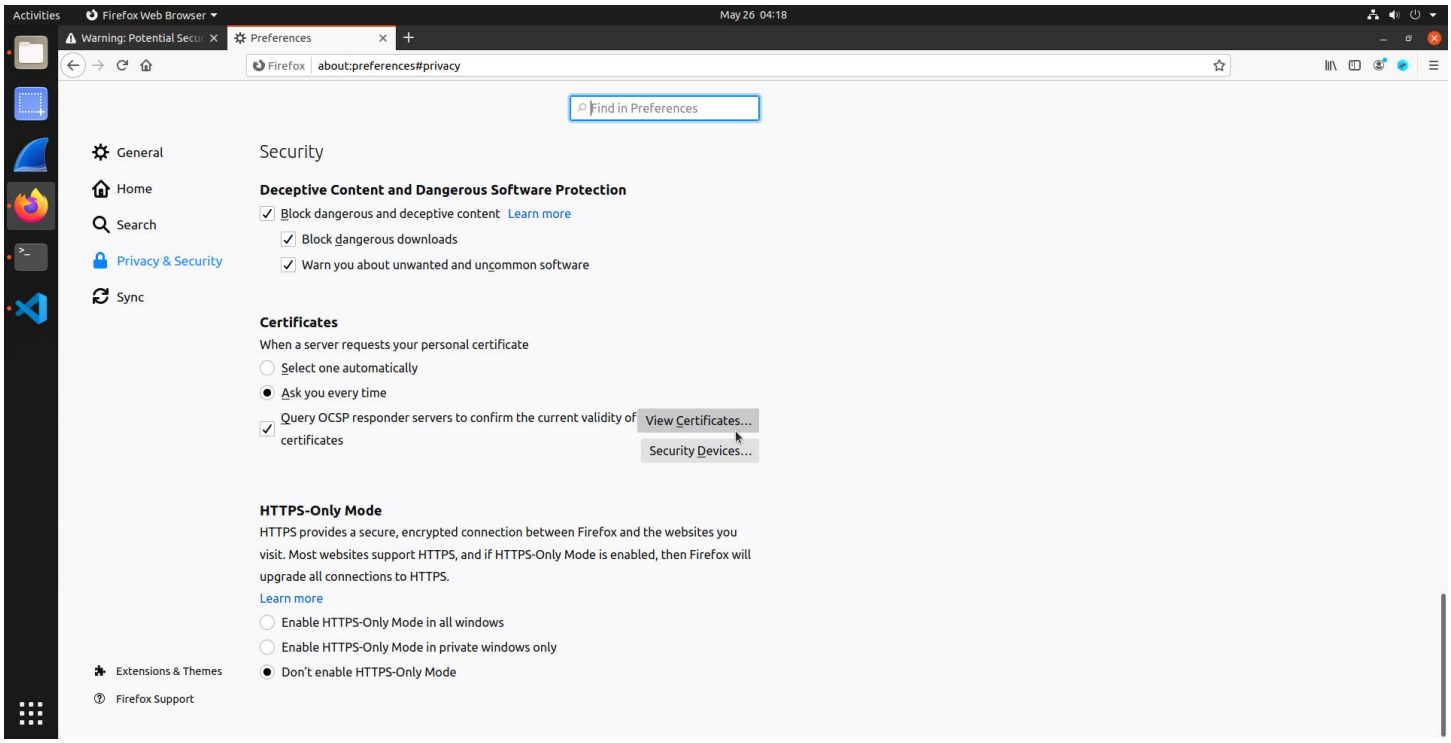
Apache sẽ hỏi ta passphrase của www.duy2022.com và www.bank32.com, chúng lần lượt là 123456 và dees.

Và khi ta truy cập thử trang web bằng firefox:

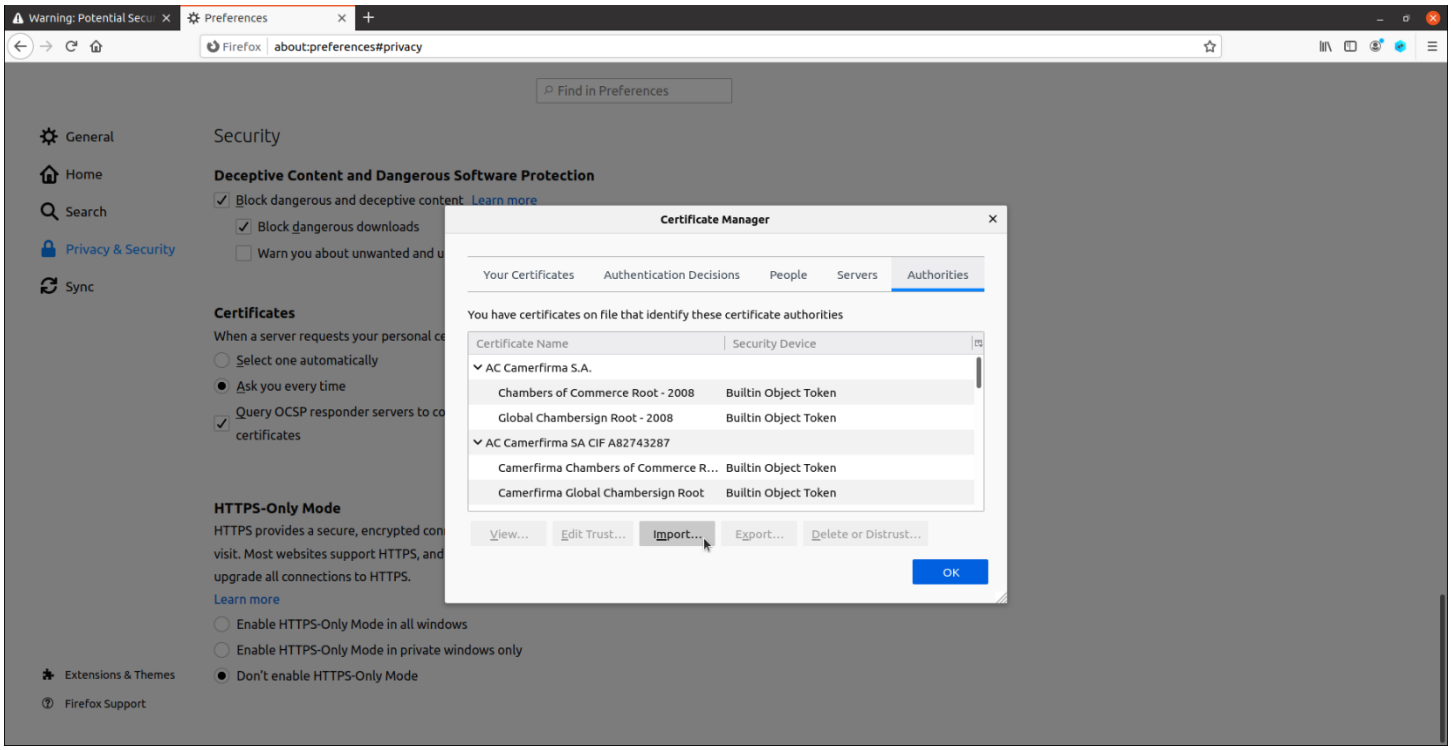


Ta nhận được warning này vì lúc này browser chưa tin tưởng website của chúng ta vì không có CA nào mà firefox tin tưởng có thể chứng minh website của ta không phải là website xấu. Để làm browser tin tưởng, ta phải thêm certificate của CA – file `ca.crt` được tạo ra từ Task 1 – vào trong list các CA được tin tưởng bởi firefox. Ta thêm bằng cách như sau:

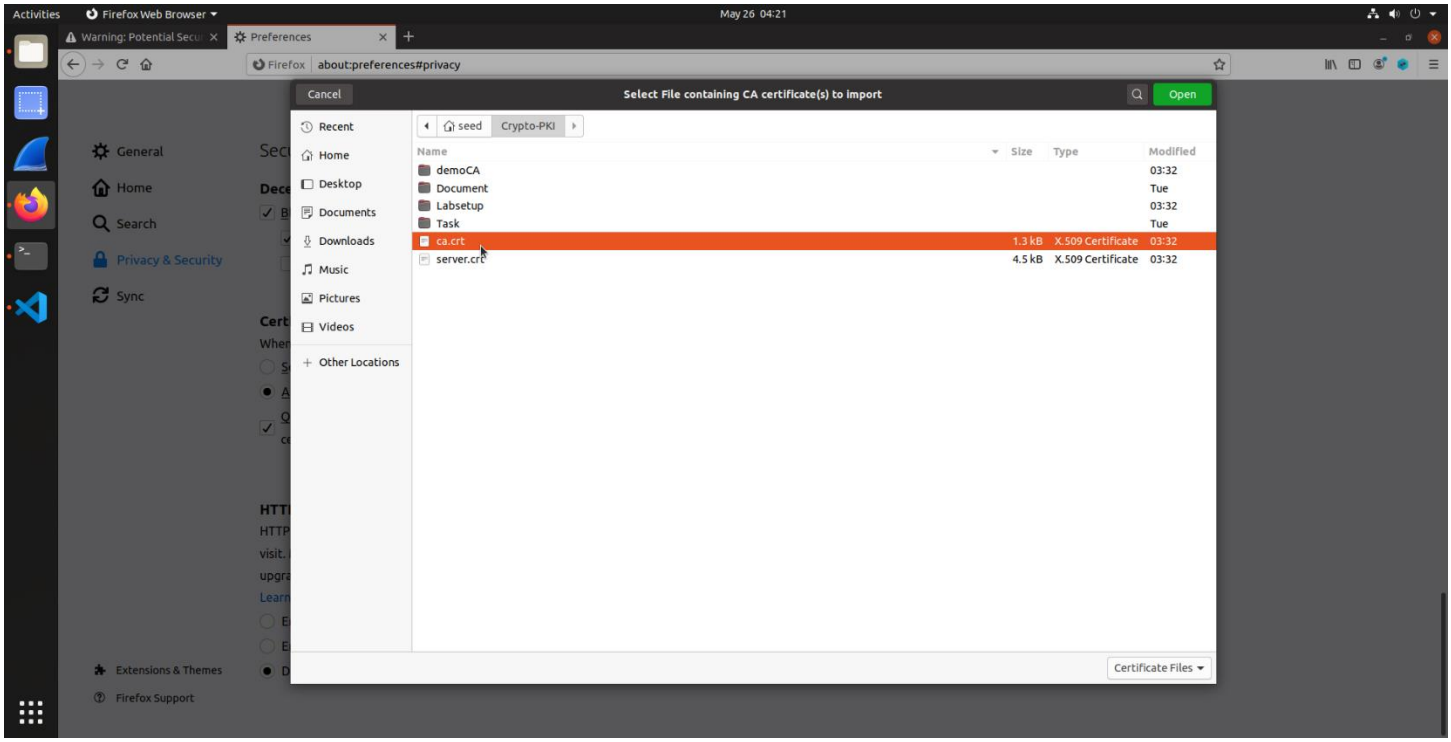
Truy cập about:preferences#privacy và kéo xuống dưới phần Certificates:



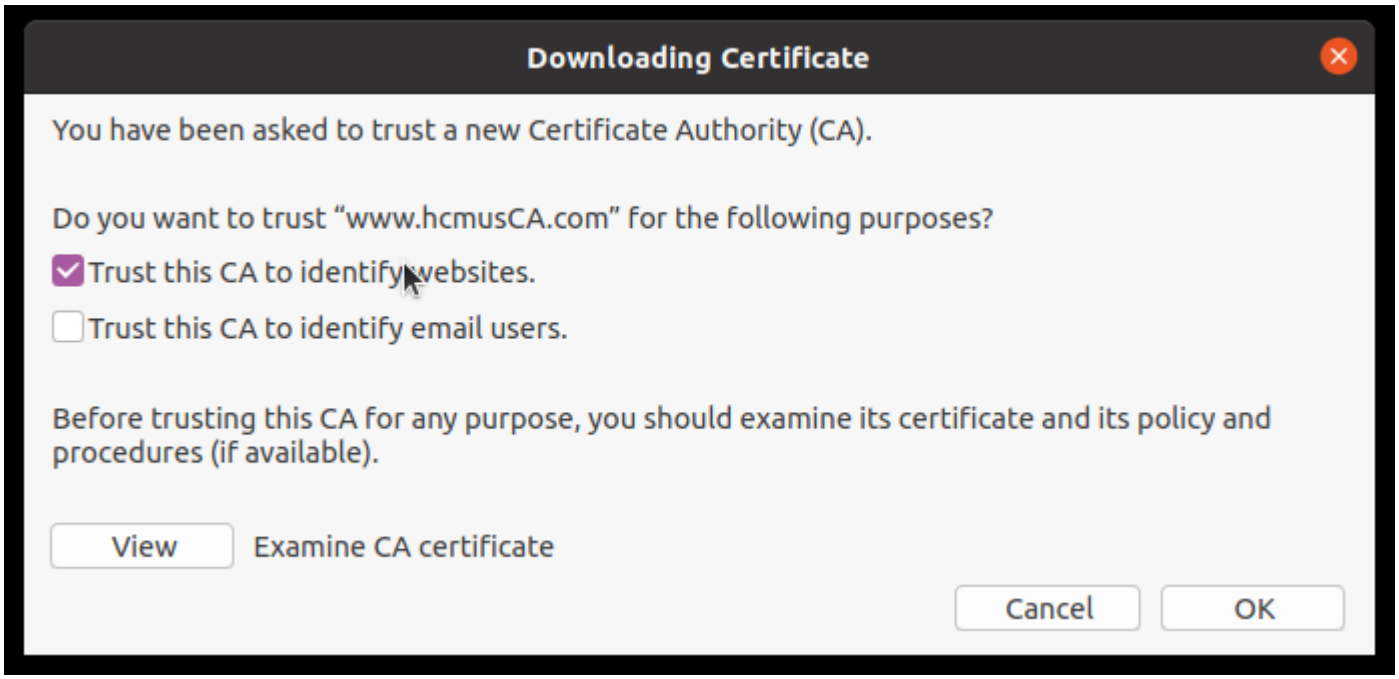
Bấm vào nút View Certificates, ta sẽ có danh sách các certificate của các CA mà firefox tin tưởng:



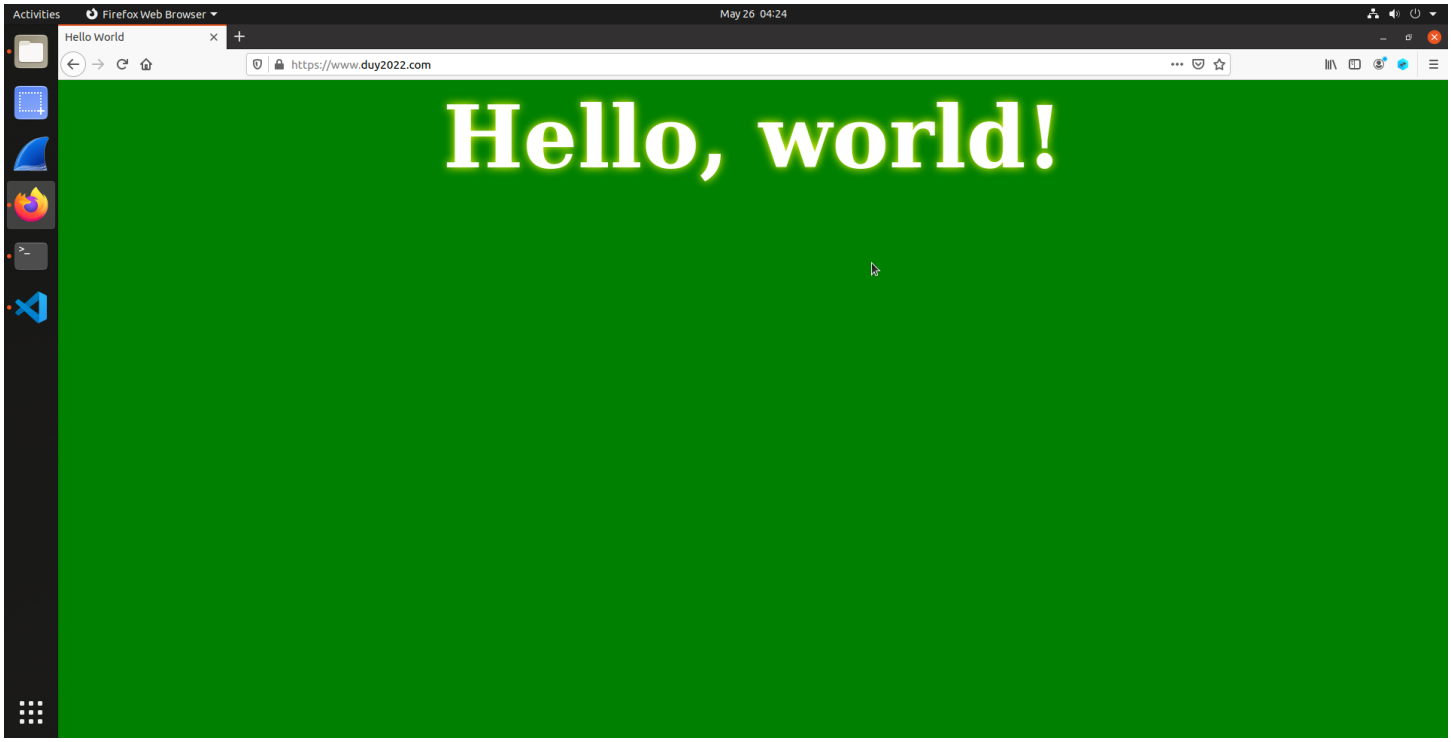
Bấm vào Import, và chọn file ca.crt từ Task 1:



Tick “Trust this CA to identify websites” và bấm OK -> OK:



Sau đó ta sẽ tắt trình duyệt mở lại để chắc chắn, và truy cập lại https://www.duy2022.com, lúc này trang web đã hoạt đông bình thường:



Các bước ở trên em dùng các lệnh mà ta có thể dùng lúc bình thường để thực hiện, nhưng trong source code mà chúng em nộp cho thầy, vì muốn có tính chất tự động, em có các thay đổi sau đây:

- Khi chạy container, vì nó sẽ chạy trên terminal mà nó được chạy cho đến khi nó dừng, nên em phải đưa nó ra một terminal mới để cho các lệnh sau đó có thể được thực hiện. Em làm như sau:

```
gnome-terminal -- sh -c "docker-compose up"
```

  Sau câu lệnh này em có để lệnh sleep 2 để đợi cho tới khi nào container bắt đầu khởi chạy, nhưng đây chỉ là đối với máy em, với từng máy thì tốc độ sẽ khác nhau nên cần chỉnh lại số giây đợi để phù hợp.

- Việc lấy ID của container, thay vì dùng lệnh `docker ps`, thì em dùng `docker ps --format "{{.ID}}"` để chỉ lấy ID của container thôi, và để output của nó vào trong 1 biến trong terminal, cụ thể đầy đủ câu lệnh như sau:

```
CONTAINER_ID=$(docker ps --format "{{.ID}}")
```

- Và việc chạy lệnh trong container, thì em viết sẵn các lệnh mà em muốn chạy trong container vào trong 1 file bash riêng (`container_commands.sh`). Sau đó em sẽ dùng lệnh `cat` để in hết nội dung file đó ra, và để vào trong 1 biến khác, và kết hợp biến `CONTAINER_ID` để có thể thành công chạy được những lệnh em mong muốn trong container. Cụ thể lệnh như sau:

```
CONTAINER_COMMANDS=$(cat ../Task/Task4/container_commands.sh)
docker exec -it $CONTAINER_ID /bin/bash -c "$CONTAINER_COMMANDS"
```

  Em làm tương tự với source code của Task 5 và Task 6.

5. Task 5: Launching a Man-In-The-Middle Attack

Với Task này và Task 6, em sẽ chọn www.google.com để làm website mục tiêu. Việc setup web www.google.com giả em cũng sẽ làm các bước y hệt như Task 4, trừ bước setup certificate, private key và bước nạp certificate của CA cho browser. Vì ta không phải là CA của www.google.com, nên ta sẽ không thể generate certificate và private key cho www.google.com giả, cho nên bước setup certificate ta phải xài tạm certificate cũ từ Task 3, cũng vì lí do đó, ta cũng không thể thực hiện bước nạp certificate của CA cho browser, vì ta không phải CA.

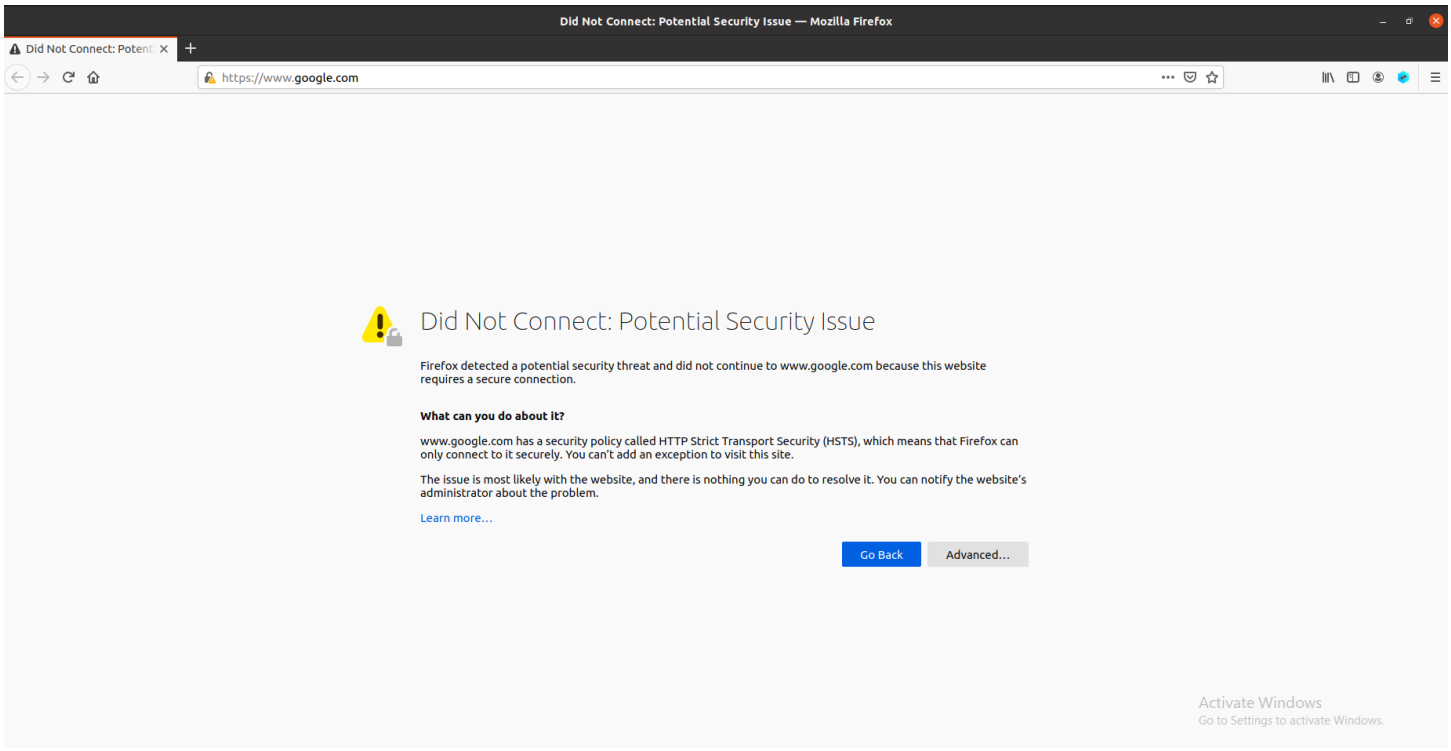
Bước config website thì nội dung file config em để như sau:

```
<VirtualHost *:443>
    DocumentRoot /var/www/google
    ServerName www.google.com
    DirectoryIndex index.html
    SSLEngine On
    SSLCertificateFile /certs/server.crt
    SSLCertificateKeyFile /certs/server.key
</VirtualHost>

<VirtualHost *:80>
    DocumentRoot /var/www/google
    ServerName www.google.com
    DirectoryIndex index_red.html
</VirtualHost>

# Set the following gloal entry to suppress an annoying warning message
ServerName localhost
```

Như trên đã nói, `SSLCertificateFile` và `SSLCertificateKeyFile` em dùng lại 2 file từ Task 2, 3.

Với các bước setup tài nguyên, enable website, setup DNS thì em làm tương tự Task 4. Tuy nhiên lúc này bước setup DNS dù ta đang làm y hệt Task 4, nhưng ý nghĩa của việc làm đó lại khác, lúc này ta đang giả sử thực hiện một cuộc tấn công DNS cache poisoning làm cho DNS bị thay đổi.

Sau khi làm xong hết các bước, thì đây là những gì browser hiện:



Ta thấy rằng browser không tin tưởng website giả của chúng ta, đơn giản vì certificate của website có subject là www.duy2022.com, không phải www.google.com, dù cho nó có được một CA cung cấp certificate. Điều đó chứng minh được PKI có thể chống được một cuộc tấn công Man-in-the-middle như vậy.

6. Task 6: Launching a Man-In-The-Middle Attack with a Compromised CA

Task này ta giả sử ta có được private key của một CA, cụ thể là CA ta tạo ở Task 1, lúc này ta có thể dụng private key đó để tạo một certificate mới của CA, và từ đó ta có thể tạo được certificate

cho bất kỳ web nào, và dùng nó để thực hiện thành công tấn công Man-in-the-middle với bất kỳ web nào.

Dù đề không đề cập, nhưng em tự hiểu ngoài private key của CA, thì ta cũng có luôn passphrase của private key đó, bước đầu tiên là em sẽ tạo một certificate của CA đó bằng lệnh như sau:

```
openssl req -new -x509 -key ca.key -out ca.crt -config openssl.cnf -subj
"/C=VN/ST=HCM/L=HCM/O=HCMUS/OU=HCMUS/CN=www.hcmusCA.com" -passin pass:123456
```

Lệnh này khác so với lệnh ở Task 1 chỉ ở chỗ tham số -key thay vì là -keyout, vì ta đã có private key của CA rồi, ta chỉ đang sử dụng private key đó để tạo ra certificate mới, và -passin thay vì là -passout vì ta đang sử dụng một private key đã có passphrase.

Sau đó ta tạo certificate request cho www.google.com (giống Task 2):

```
openssl req -newkey rsa:2048 -sha256 -keyout google.key -out google.csr -subj
"/CN=www.google.com/O=Google Inc./C=US" -passout pass:123456
```

Và từ đó tạo ra certificate cho www.google.com bằng certificate của CA và certificate request ở trên (giống Task 3):

```
openssl ca -config openssl.cnf -policy policy_anything -md sha256 -days 3650 -in google.csr -
out google.crt -batch -cert ca.crt -keyfile ca.key -passin pass:123456
```

Sau khi làm những bước trên, ta đã có file `google.crt, google.key`, ta sẽ đưa chúng vào trong thư mục `/certs` của container và chỉnh lại config của www.google.com như sau:

```
<VirtualHost *:443>
    DocumentRoot /var/www/google
    ServerName www.google.com
    DirectoryIndex index.html
    SSLEngine On
    SSLCertificateFile /certs/google.crt
    SSLCertificateKeyFile /certs/google.key
</VirtualHost>

<VirtualHost *:80>
    DocumentRoot /var/www/google
    ServerName www.google.com
    DirectoryIndex index_red.html
</VirtualHost>

# Set the following gloal entry to suppress an annoying warning message
ServerName localhost
```

Ta sử dụng 2 file google.crt, google.key vừa được tạo thay vì 2 file cũ từ Task 3.

Sau khi làm hết những điều trên, khi ta truy cập lại https://www.google.com:

Như vậy ta đã thực hiện thành công cuộc tấn công Man-in-the-middle với website https://www.google.com chỉ với private key của một CA.

VIDEO CHẠY CHƯƠNG TRÌNH

Chúng em có làm thêm một video quay lại quá trình chạy hết 6 task để thầy xem qua: https://studenthcmusedu-my.sharepoint.com/:v:/g/personal/19120219_student_hcmus_edu_vn/Ee0r3NAyiftMmunzl4rzgDkBMZBAV2Bdo9voNcuapOrv0g?e=vR7OaD