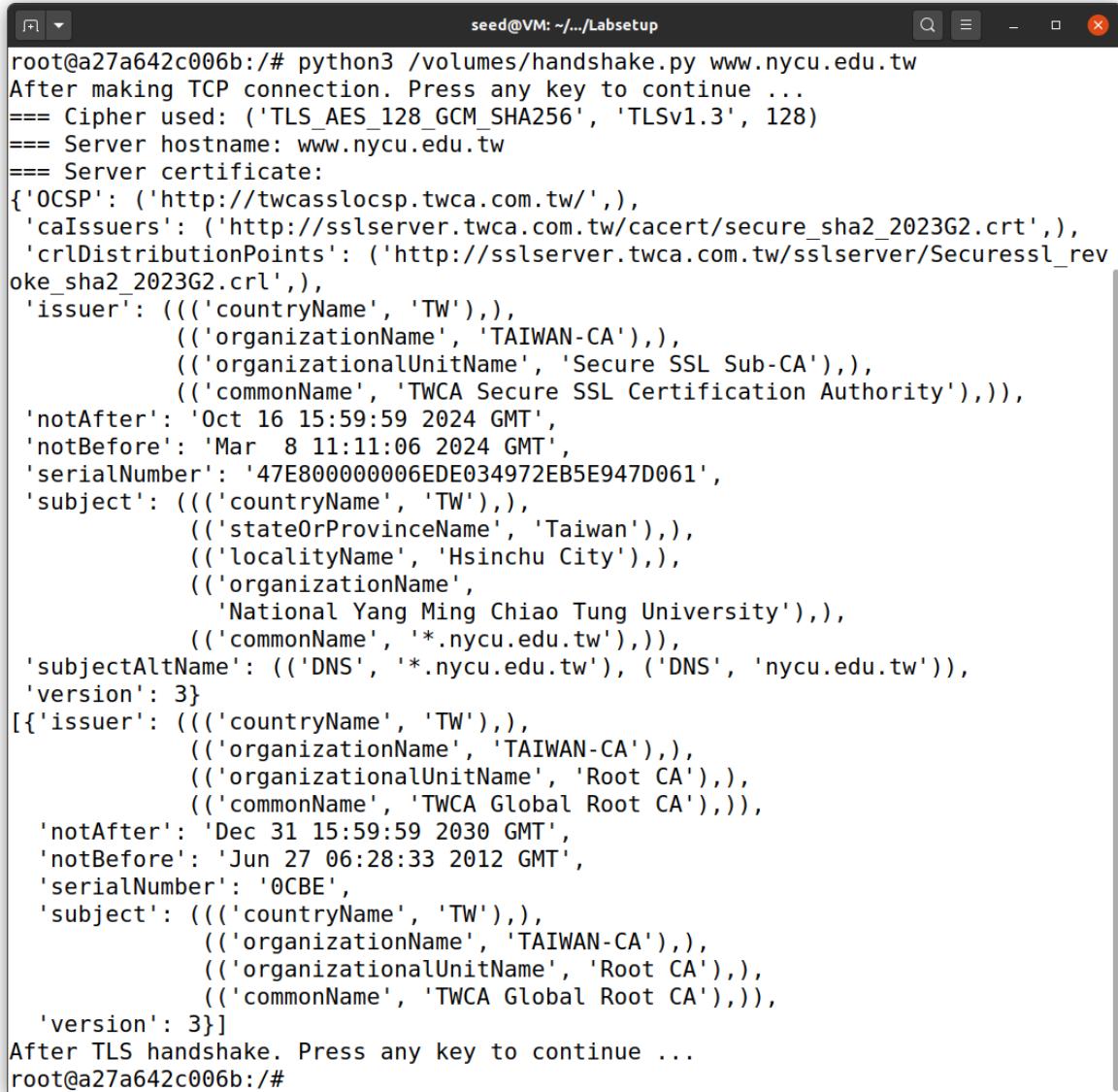


姓名：蔡佩蓉 學號：109511286

## Transport Layer Security (TLS) Lab (Lab6)

### Task 1: TLS Client

#### Task 1.1: TLS handshake



```
seed@VM: ~/.../Labsetup
root@a27a642c006b:/# python3 /volumes/handshake.py www.nycu.edu.tw
After making TCP connection. Press any key to continue ...
==> Cipher used: ('TLS_AES_128_GCM_SHA256', 'TLSv1.3', 128)
==> Server hostname: www.nycu.edu.tw
==> Server certificate:
{'OCSP': ('http://twcasslocsp.twca.com.tw/'),
 'caIssuers': ('http://sslserver.twca.com.tw/cacert/secure_sha2_2023G2.crt'),
 'crlDistributionPoints': ('http://sslserver.twca.com.tw/sslserver/Securessl_revoke_sha2_2023G2.crl'),
 'issuer': (((('countryName', 'TW'),),
              (('organizationName', 'TAIWAN-CA'),),
              (('organizationalUnitName', 'Secure SSL Sub-CA'),),
              (('commonName', 'TWCA Secure SSL Certification Authority'))),
             'notAfter': 'Oct 16 15:59:59 2024 GMT',
             'notBefore': 'Mar 8 11:11:06 2024 GMT',
             'serialNumber': '47E800000006EDE034972EB5E947D061',
             'subject': (((('countryName', 'TW'),),
                           (('stateOrProvinceName', 'Taiwan'),),
                           (('localityName', 'Hsinchu City'),),
                           (('organizationName',
                             'National Yang Ming Chiao Tung University'),),
                           (('commonName', '*.nycu.edu.tw'))),
            'subjectAltName': (('DNS', '*.nycu.edu.tw'), ('DNS', 'nycu.edu.tw')),
            'version': 3}
[{'issuer': (((('countryName', 'TW'),),
              (('organizationName', 'TAIWAN-CA'),),
              (('organizationalUnitName', 'Root CA'),),
              (('commonName', 'TWCA Global Root CA'))),
             'notAfter': 'Dec 31 15:59:59 2030 GMT',
             'notBefore': 'Jun 27 06:28:33 2012 GMT',
             'serialNumber': '0CBE',
             'subject': (((('countryName', 'TW'),),
                           (('organizationName', 'TAIWAN-CA'),),
                           (('organizationalUnitName', 'Root CA'),),
                           (('commonName', 'TWCA Global Root CA'))),
            'version': 3}]
After TLS handshake. Press any key to continue ...
root@a27a642c006b:/#
```

#### 1. What is the cipher used between the client and the server?

Cipher used: ('TLS\_AES\_128\_GCM\_SHA256', 'TLSv1.3', 128)

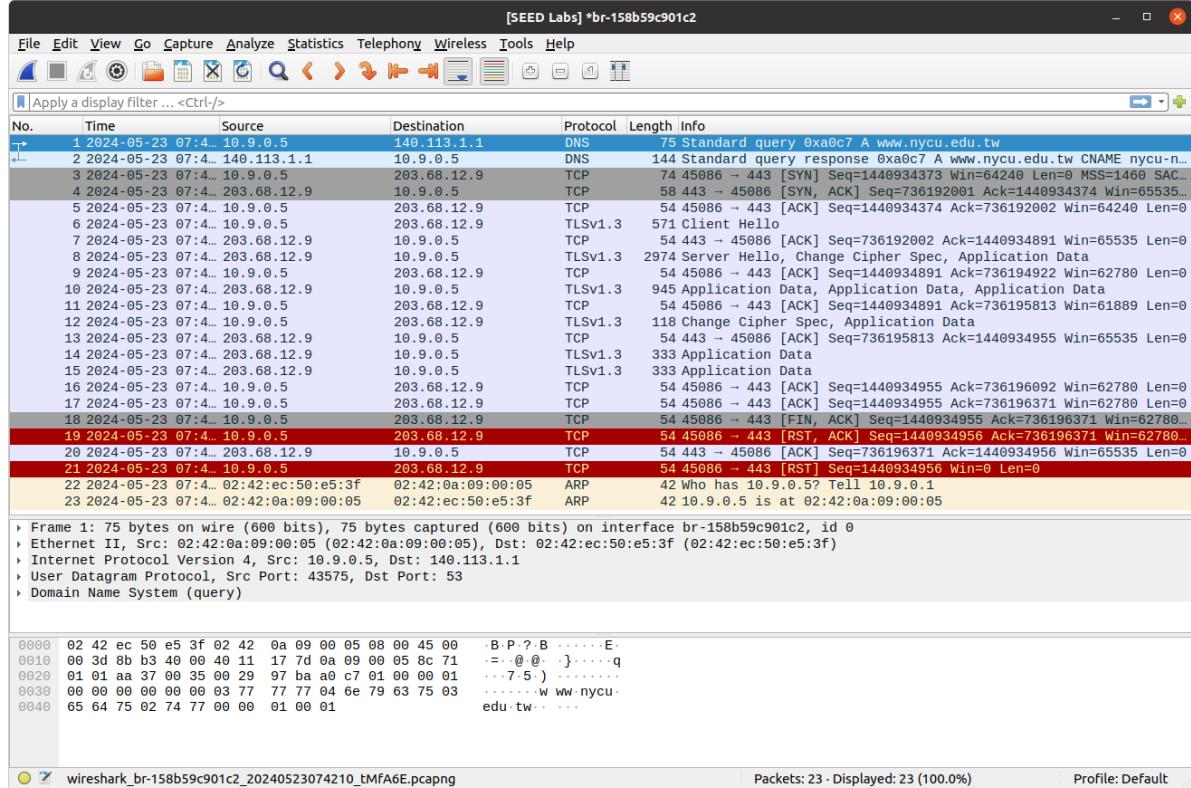
#### 2. Please print out the server certificate in the program.

Shown above.

#### 3. Explain the purpose of /etc/ssl/certs.

The directory /etc/ssl/certs contains a collection of root (trusted) CA (Certificate Authority) certificates. These certificates are used by the client to verify the authenticity of the server's certificate. By specifying this directory, the script ensures that it only trusts certificates issued by recognized CAs stored in this location.

**4. Use Wireshark to capture the network traffics during the execution of the program, and explain your observation. In particular, explain which step triggers the TCP handshake, and which step triggers the TLS handshake. Explain the relationship between the TLS handshake and the TCP handshake.**

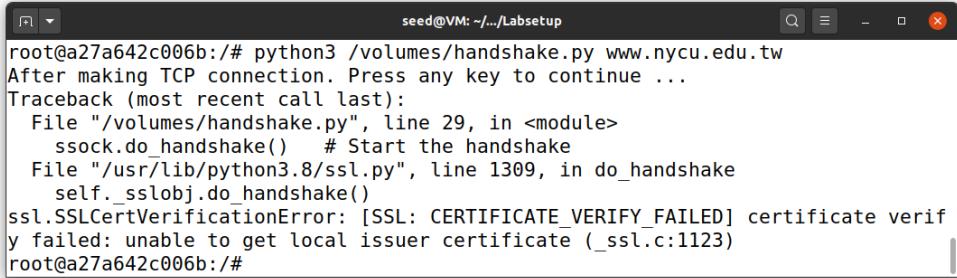


The TCP handshake consists of three packets: SYN, SYN-ACK, and ACK [index 3 to 5], while the TLS handshake involves multiple packets. [index 6 to 13]

The client first sends out Client Hello related data, including all cipher suites supported by the client, client random numbers (used as Nonce), and etc. The server then replies to the client Server Hello related data and the certificate. Subsequently, the client verifies the validity of the certificate. If it is valid, the client sends a Key Exchange Client Key Exchange and a Change Cipher Spec (change cipher specification message) to the server, and the client handshake ends. The server replies to Change Cipher Spec (change cipher specification message) and a New Session value, and the handshake ends.

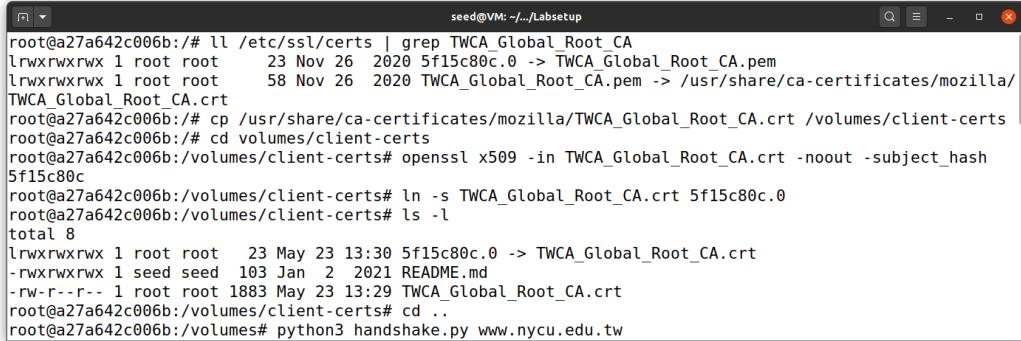
The TLS handshake occurs after the TCP handshake. The TCP handshake establishes a basic, reliable connection, and the TLS handshake builds on this connection to negotiate security parameters and establish an encrypted session. Essentially, the TLS handshake requires the underlying TCP connection to be established first, ensuring that the communication is both reliable and secure.

## Task 1.2: CA's Certificate



```
seed@VM: ~/.../Labsetup
root@a27a642c006b:/# python3 /volumes/handshake.py www.nycu.edu.tw
After making TCP connection. Press any key to continue ...
Traceback (most recent call last):
  File "/volumes/handshake.py", line 29, in <module>
    ssock.do_handshake()  # Start the handshake
  File "/usr/lib/python3.8/ssl.py", line 1309, in do_handshake
    self._sslobj.do_handshake()
ssl.SSLCertVerificationError: [SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed: unable to get local issuer certificate (_ssl.c:1123)
root@a27a642c006b:/#
```

As observed, TLS failed. This is because the corresponding CA certificate was not found in ./client-certs.



```
seed@VM: ~/.../Labsetup
root@a27a642c006b:/# ll /etc/ssl/certs | grep TWCA_Global_Root_CA
lrwxrwxrwx 1 root root 23 Nov 26 2020 5f15c80c.0 -> TWCA_Global_Root_CA.pem
lrwxrwxrwx 1 root root 58 Nov 26 2020 TWCA_Global_Root_CA.pem -> /usr/share/ca-certificates.mozilla/TWCA_Global_Root_CA.crt
root@a27a642c006b:/# cp /usr/share/ca-certificates.mozilla/TWCA_Global_Root_CA.crt /volumes/client-certs
root@a27a642c006b:/# cd /volumes/client-certs
root@a27a642c006b:/volumes/client-certs# openssl x509 -in TWCA_Global_Root_CA.crt -noout -subject_hash 5f15c80c
root@a27a642c006b:/volumes/client-certs# ln -s TWCA_Global_Root_CA.crt 5f15c80c.0
root@a27a642c006b:/volumes/client-certs# ls -l
total 8
lrwxrwxrwx 1 root root 23 May 23 13:30 5f15c80c.0 -> TWCA_Global_Root_CA.crt
-rw-rw-r-- 1 seed seed 103 Jan 2 2021 README.md
-rw-r--r-- 1 root root 1883 May 23 13:29 TWCA_Global_Root_CA.crt
root@a27a642c006b:/volumes/client-certs# cd ..
root@a27a642c006b:/volumes# python3 handshake.py www.nycu.edu.tw
```

Identify the required CA certificate to verify the server (www.nycu.edu.tw) certificate, and then copy the certificate from the /etc/ssl/certs into ./client-certs. Simply copying the CA certificate into the client-certs directory does not create the necessary hashed symbolic links that the TLS client expects. Without these links, the client cannot locate the CA certificate needed to verify the server's certificate.



```
seed@VM: ~/.../Labsetup
root@a27a642c006b:/volumes# python3 handshake.py www.nycu.edu.tw
After making TCP connection. Press any key to continue ...
==> Cipher used: ('TLS_AES_128_GCM_SHA256', 'TLSv1.3', 128)
==> Server hostname: www.nycu.edu.tw
==> Server certificate:
{'OCSP': ('http://twcasslocsp.twca.com.tw/'),
 'caIssuers': ('http://sslserver.twca.com.tw/cacert/secure_sha2_2023G2.crt',),
 'crlDistributionPoints': ('http://sslserver.twca.com.tw/sslserver/SecureSSL_revocate_sha2_2023G2.crl',),
 'issuer': ((('countryName', 'TW'),),
            (('organizationName', 'TAIWAN-CA'),),
            (('organizationalUnitName', 'Secure SSL Sub-CA'),),
            (('commonName', 'TWCA Secure SSL Certification Authority'))),
 'notAfter': 'Oct 16 15:59:59 2024 GMT',
 'notBefore': 'Mar 8 11:11:06 2024 GMT',
 'serialNumber': '47E800000006EDE034972EB5E947D061',
 'subject': ((('countryName', 'TW'),),
             (('stateOrProvinceName', 'Taiwan'),),
             (('localityName', 'Hsinchu City'),),
             (('organizationName', 'National Yang Ming Chiao Tung University'),),
             (('commonName', '*.nycu.edu.tw'))),
 'subjectAltName': (('DNS', '*.nycu.edu.tw'), ('DNS', 'nycu.edu.tw')),
 'version': 3}
[{'issuer': ((('countryName', 'TW'),),
            (('organizationName', 'TAIWAN-CA'),),
            (('organizationalUnitName', 'Root CA'),),
            (('commonName', 'TWCA Global Root CA'))),
 'notAfter': 'Dec 31 15:59:59 2030 GMT',
 'notBefore': 'Jun 27 06:28:33 2012 GMT',
 'serialNumber': '0CBE',
 'subject': ((('countryName', 'TW'),),
             (('organizationName', 'TAIWAN-CA'),),
             (('organizationalUnitName', 'Root CA'),),
             (('commonName', 'TWCA Global Root CA'))),
 'version': 3}]
After TLS handshake. Press any key to continue ...
root@a27a642c006b:/volumes#
```

## Additional requirement:

```
seed@VM: ~/.../Labsetup
root@a27a642c006b:/volumes# python3 handshake.py www.google.com
After making TCP connection. Press any key to continue ...
==> Cipher used: ('TLS_AES_256_GCM_SHA384', 'TLSv1.3', 256)
==> Server hostname: www.google.com
==> Server certificate:
{'OCSP': ('http://ocsp.pki.goog/gts1c3',),
 'caIssuers': ('http://pki.goog/repo/certs/gts1c3.der',),
 'crlDistributionPoints': ('http://crls.pki.goog/gts1c3/Q0vJ0N1sT2A.crl',),
 'issuer': ((('countryName', 'US'),),
             (('organizationName', 'Google Trust Services LLC'),),
             (('commonName', 'GTS CA 1C3'))),
 'notAfter': 'Jul 29 14:44:49 2024 GMT',
 'notBefore': 'May 6 14:44:50 2024 GMT',
 'serialNumber': '815C75BD499012C10A63E95EDC4CD136',
 'subject': ((('commonName', 'www.google.com'),),
              ('subjectAltName': ((('DNS', 'www.google.com'),),
              'version': 3)
[{'issuer': ((('countryName', 'US'),),
             (('organizationName', 'Google Trust Services LLC'),),
             (('commonName', 'GTS Root R1'))),
 'notAfter': 'Jun 22 00:00:00 2036 GMT',
 'notBefore': 'Jun 22 00:00:00 2016 GMT',
 'serialNumber': '6E47A9C54B470C0DEC33D089B91CF4E1',
 'subject': ((('countryName', 'US'),),
             (('organizationName', 'Google Trust Services LLC'),),
             (('commonName', 'GTS Root R1'))),
 'version': 3}]
After TLS handshake. Press any key to continue ...
```

```
seed@VM: ~/.../Labsetup
root@a27a642c006b:# ll /etc/ssl/certs | grep GTS_Root_R1
lrwxrwxrwx 1 root root 15 Nov 26 2020 1001acf7.0 -> GTS_Root_R1.pem
lrwxrwxrwx 1 root root 50 Nov 26 2020 GTS_Root_R1.pem -> /usr/share/ca-certificates/mozilla/GTS_Root_R1.crt
root@a27a642c006b:# cp /usr/share/ca-certificates/mozilla/GTS_Root_R1.crt /volumes/client-certs
root@a27a642c006b:# cd /volumes/client-certs
root@a27a642c006b:/volumes/client-certs# openssl x509 -in GTS_Root_R1.crt -noout -subject_hash 1001acf7
root@a27a642c006b:/volumes/client-certs# ln -s GTS_Root_R1.crt 1001acf7.0
root@a27a642c006b:/volumes/client-certs# ls -l
total 12
lrwxrwxrwx 1 root root 15 May 23 13:39 1001acf7.0 -> GTS_Root_R1.crt
lrwxrwxrwx 1 root root 23 May 23 13:30 5f15c80c.0 -> TWCA_Global_Root_CA.crt
-rw-r--r-- 1 root root 1915 May 23 13:38 GTS_Root_R1.crt
-rwxrwxrwx 1 seed seed 103 Jan 2 2021 README.md
-rw-r--r-- 1 root root 1883 May 23 13:29 TWCA_Global_Root_CA.crt
root@a27a642c006b:/volumes/client-certs# cd ..
root@a27a642c006b:/volumes# python3 handshake.py www.google.com
```

```
seed@VM: ~.../Labsetup
root@a27a642c006b:/volumes# python3 handshake.py www.google.com
After making TCP connection. Press any key to continue ...
==> Cipher used: ('TLS_AES_256_GCM_SHA384', 'TLSv1.3', 256)
==> Server hostname: www.google.com
==> Server certificate:
{'OCSP': ('http://ocsp.pki.goog/gts1c3',),
 'caIssuers': ('http://pki.goog/repo/certs/gts1c3.der',),
 'crlDistributionPoints': ('http://crls.pki.goog/gts1c3/Q0vJ0N1sT2A.crl',),
 'issuer': (((('countryName', 'US'),),
              (('organizationName', 'Google Trust Services LLC'),),
              (('commonName', 'GTS CA 1C3'))),
             {'notAfter': 'Jul 29 14:44:49 2024 GMT',
              'notBefore': 'May 6 14:44:50 2024 GMT',
              'serialNumber': '815C75BD499012C10A63E95EDC4CD136',
              'subject': (((('commonName', 'www.google.com'),),
                           {'version': 3}),
                         {'issuer': (((('countryName', 'US'),),
                                      (('organizationName', 'Google Trust Services LLC'),),
                                      (('commonName', 'GTS Root R1'))),
                           {'notAfter': 'Jun 22 00:00:00 2036 GMT',
                            'notBefore': 'Jun 22 00:00:00 2016 GMT',
                            'serialNumber': '6E47A9C54B470C0DEC33D089B91CF4E1',
                            'subject': (((('countryName', 'US'),),
                                         {'version': 3})]}),
             After TLS handshake. Press any key to continue ...
```

### Task 1.3: Experiment with the hostname check

```
seed@VM: ~/Desktop
[05/23/24] seed@VM:~/Desktop$ dig www.google.com

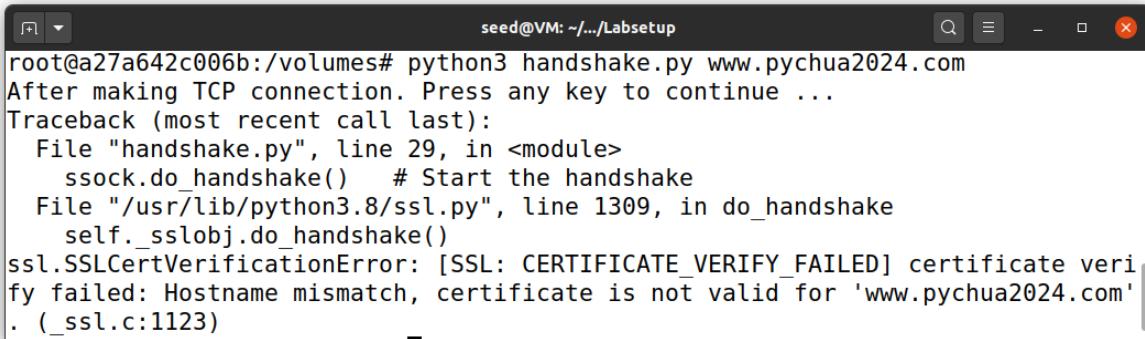
; <>> DiG 9.16.1-Ubuntu <>> www.google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 48786
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
www.google.com.           IN      A

;; ANSWER SECTION:
www.google.com.      2879      IN      A      172.217.160.68

;; Query time: 0 msec
;; SERVER: 127.0.0.53#53(127.0.0.53)
;; WHEN: Thu May 23 10:12:12 EDT 2024
;; MSG SIZE  rcvd: 59
```

### When context.check\_hostname = True:

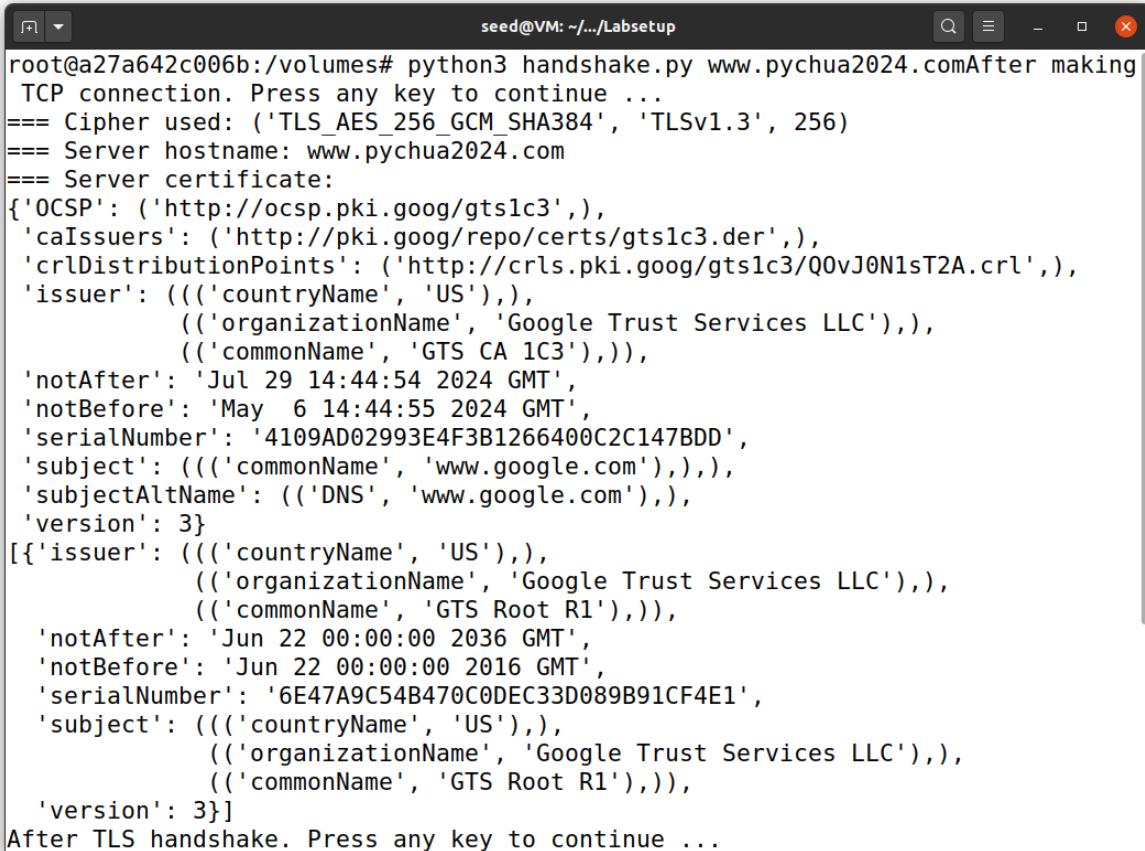


```
root@a27a642c006b:/volumes# python3 handshake.py www.pychua2024.com
After making TCP connection. Press any key to continue ...
Traceback (most recent call last):
  File "handshake.py", line 29, in <module>
    ssock.do_handshake()    # Start the handshake
  File "/usr/lib/python3.8/ssl.py", line 1309, in do_handshake
    self._sslobj.do_handshake()
ssl.SSLCertVerificationError: [SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed: Hostname mismatch, certificate is not valid for 'www.pychua2024.com' . (_ssl.c:1123)
```

Observation: The client program will fail to connect to www.pychua2024.com, raising an ssl.SSLCertVerificationError indicating a hostname mismatch.

Explanation: Enabling hostname verification ensures that the server certificate matches the hostname the client is trying to connect to (www.pychua2024.com). Since the certificate is for www.google.com, the verification fails, and the client does not establish a connection.

### When context.check\_hostname = False:



```
root@a27a642c006b:/volumes# python3 handshake.py www.pychua2024.com
After making TCP connection. Press any key to continue ...
==> Cipher used: ('TLS_AES_256_GCM_SHA384', 'TLSv1.3', 256)
==> Server hostname: www.pychua2024.com
==> Server certificate:
{'OCSP': ('http://ocsp.pki.goog/gts1c3',),
 'caIssuers': ('http://pki.goog/repo/certs/gts1c3.der',),
 'crlDistributionPoints': ('http://crls.pki.goog/gts1c3/Q0vJ0N1sT2A.crl',),
 'issuer': ((('countryName', 'US'),),
             (('organizationName', 'Google Trust Services LLC'),),
             (('commonName', 'GTS CA 1C3'))),
 'notAfter': 'Jul 29 14:44:54 2024 GMT',
 'notBefore': 'May  6 14:44:55 2024 GMT',
 'serialNumber': '4109AD02993E4F3B1266400C2C147BDD',
 'subject': ((('commonName', 'www.google.com'),),
             ('subjectAltName': (('DNS', 'www.google.com'),),
              'version': 3}
[{'issuer': ((('countryName', 'US'),),
             (('organizationName', 'Google Trust Services LLC'),),
             (('commonName', 'GTS Root R1'))),
 'notAfter': 'Jun 22 00:00:00 2036 GMT',
 'notBefore': 'Jun 22 00:00:00 2016 GMT',
 'serialNumber': '6E47A9C54B470C0DEC33D089B91CF4E1',
 'subject': ((('countryName', 'US'),),
             (('organizationName', 'Google Trust Services LLC'),),
             (('commonName', 'GTS Root R1'))),
 'version': 3}]
After TLS handshake. Press any key to continue ...
```

Observation: The client program connects to www.pychua2024.com without verifying if the hostname in the server certificate matches www.pychua2024.com. This connection will succeed as long as the server certificate is valid (even if the hostname doesn't match).

Explanation: Disabling hostname verification means the client doesn't check if the certificate's Common Name (CN) or Subject Alternative Name (SAN) fields match the hostname it connects to. The connection succeeds if the certificate is otherwise valid.

Hostname verification is a critical security measure in TLS/SSL for the following reasons:

- Preventing Man-in-the-Middle (MitM) Attacks: Without hostname verification, an attacker could intercept traffic by presenting a valid certificate for a different domain. This would allow them to decrypt, read, and modify the data being transmitted.
  - Ensuring Authenticity: Hostname checks ensure that the client is actually connecting to the intended server and not an imposter. This prevents spoofing attacks where a malicious server could pretend to be a legitimate server.
  - Trustworthiness of Communication: Verifying the hostname helps to ensure the integrity and trustworthiness of the communication channel, making sure that the data is transmitted to and received from the correct source.

If the client program does not perform hostname verification, it is vulnerable to various security threats:

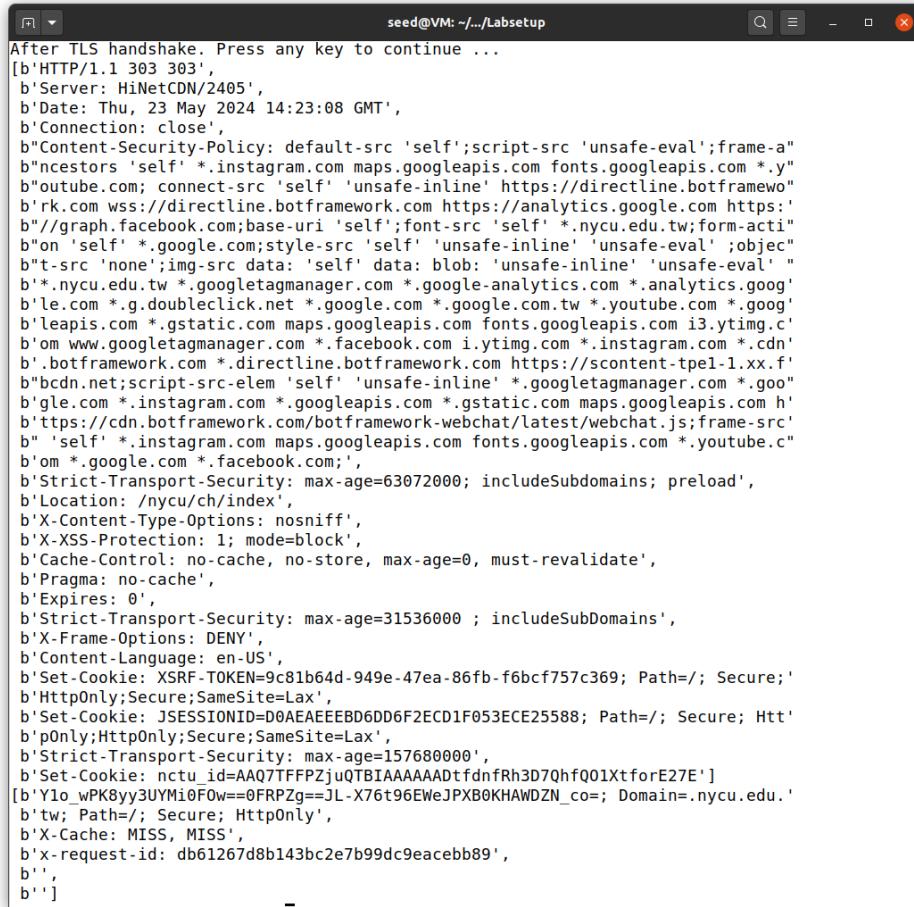
- Impersonation: Attackers could present valid certificates from other domains to impersonate the intended server, leading to unauthorized data access.
  - Data Interception: Attackers could perform MitM attacks, intercepting and potentially altering the communication between the client and the legitimate server.
  - Loss of Data Confidentiality and Integrity: Without hostname verification, the security guarantees of TLS/SSL are significantly weakened, leading to potential breaches of data confidentiality and integrity.

## **Task 1.4: Sending and getting Data**

1. Please add the data sending/receiving code to your client program, and report your observation.

```
Open Save ⌘ x
handshake.py
-/Desktop/LabSetup/volumes
24 input("After making TCP connection. Press any key to continue ...")
25
26 # Add the TLS
27 ssock = context.wrap_socket(sock, server_hostname=hostname,
28                             do_handshake_on_connect=False)
29 ssock.do_handshake()    # Start the handshake
30 print("==> Cipher used: {}".format(ssock.cipher()))
31 print("==> Server hostname: {}".format(ssock.server_hostname))
32 print("==> Server certificate:")
33 pprint.pprint(ssock.getpeercert())
34 pprint.pprint(context.get_ca_certs())
35 input("After TLS handshake. Press any key to continue ...")
36
37 # Send HTTP Request to Server
38 request = b"GET / HTTP/1.0\r\nHost: " + \
39             hostname.encode('utf-8') + b"\r\n\r\n"
40 ssock.sendall(request)
41
42 # Read HTTP Response from Server
43 response = ssock.recv(2048)
44 while response:
45     pprint.pprint(response.split(b"\r\n"))
46     response = ssock.recv(2048)
47
48 # Close the TLS Connection
49 ssock.shutdown(socket.SHUT_RDWR)
50 ssock.close()
```

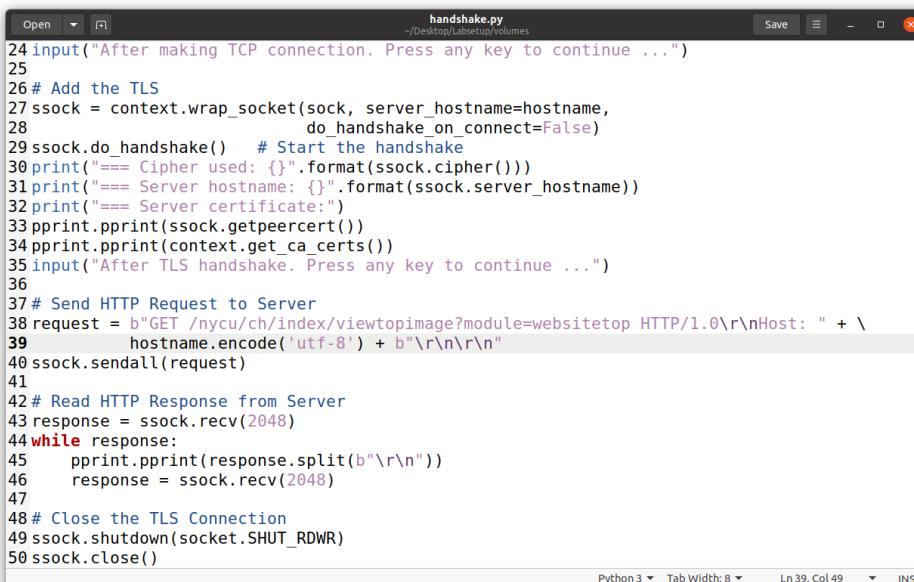
```
python3 handshake.py www.nycu.edu.tw
```



```
After TLS handshake. Press any key to continue ...
[b'HTTP/1.1 303 303',
 b'Server: HiNetCDN/2405',
 b'Date: Thu, 23 May 2024 14:23:08 GMT',
 b'Connection: close',
 b"Content-Security-Policy: default-src 'self';script-src 'unsafe-eval';frame-a"
 b"ncestors 'self' *.instagram.com maps.googleapis.com fonts.googleapis.com *.y"
 b"outube.com; connect-src 'self' 'unsafe-inline' https://directline.botframewo"
 b"rk.com wss://directline.botframework.com https://analytics.google.com https:"
 b"//graph.facebook.com;base-uri 'self';font-src 'self' *.nycu.edu.tw;form-acti"
 b"on 'self' *.google.com;style-src 'self' 'unsafe-inline' 'unsafe-eval' ;objec"
 b"t-src 'none';img-src data: 'self' data: blob: 'unsafe-inline' 'unsafe-eval' "
 b"*.nycu.edu.tw *.googletagmanager.com *.google-analytics.com *.analytics.goog"
 b"le.com *.g.doubleclick.net *.google.com *.google.com.tw *.youtube.com *.goog"
 b"leapis.com *.gstatic.com maps.googleapis.com fonts.googleapis.com i3.ytimg.c"
 b"om www.googletagmanager.com *.facebook.com i.ytimg.com *.instagram.com *.cdn"
 b".botframework.com *.directline.botframework.com https://scontent-tpe1-1.xx.f"
 b"cdn.net;script-src-elem 'self' 'unsafe-inline' *.googletagmanager.com *.goo"
 b"gle.com *.instagram.com *.googleapis.com *.gstatic.com maps.googleapis.com h"
 b"ttps://cdn.botframework.com/botframework-webchat/latest/webchat.js;frame-src"
 b" 'self' *.instagram.com maps.googleapis.com fonts.googleapis.com *.youtube.c"
 b"om *.google.com *.facebook.com",
 b'Strict-Transport-Security: max-age=63072000; includeSubdomains; preload',
 b'Location: /nycu/ch/index',
 b'X-Content-Type-Options: nosniff',
 b'X-XSS-Protection: 1; mode=block',
 b'Cache-Control: no-cache, no-store, max-age=0, must-revalidate',
 b'Pragma: no-cache',
 b'Expires: 0',
 b'Strict-Transport-Security: max-age=31536000 ; includeSubDomains',
 b'X-Frame-Options: DENY',
 b'Content-Language: en-US',
 b'Set-Cookie: XSRF-TOKEN=9c81b64d-949e-47ea-86bf-f6bcf757c369; Path=/; Secure;',
 b'HttpOnly;Secure;SameSite=Lax',
 b'Set-Cookie: JSESSIONID=D0AEAEEDB06DD6F2ECD1F053ECE25588; Path=/; Secure; Htt"
 b'pOnly;HttpOnly;Secure;SameSite=Lax',
 b'Strict-Transport-Security: max-age=157680000',
 b'Set-Cookie: nctu_id=AA07TFFPZjuQTBIAAAAADtfdnfRh3D7QhfQ01XtforE27E'
 [b'Ylo_wPK8yy3UYMi0FOw==0FRPZg==JL-X76t96EWeJPXB0KHAWDZN_co; Domain=.nycu.edu.'
 b'tw; Path=/; Secure; HttpOnly',
 b'X-Cache: MISS, MISS',
 b'x-request-id: db61267d8b143bc2e7b99dc9eacebb89',
 b '',
 b'']
```

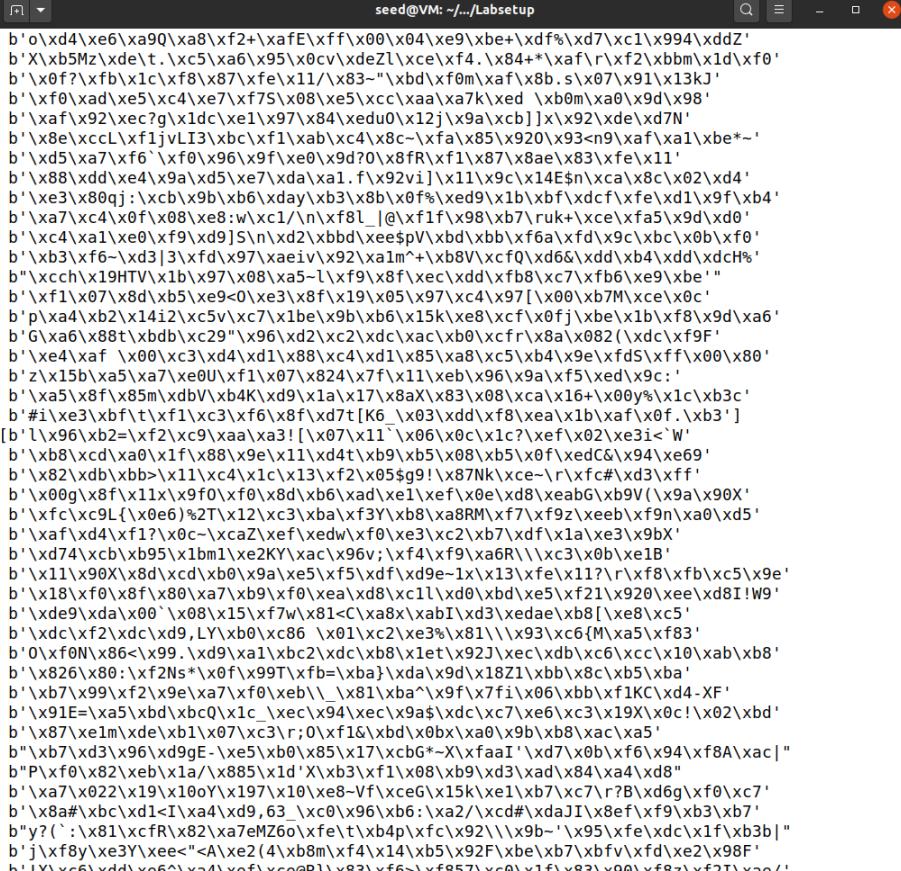
Obtained the source code (html) of www.nycu.edu.tw.

## 2. Please modify the HTTP request, so you can fetch an image file of your choice from an HTTPS server (there is no need to display the image).



```
Open Save
handshake.py
24 input("After making TCP connection. Press any key to continue ...")
25
26 # Add the TLS
27 ssock = context.wrap_socket(sock, server_hostname=hostname,
28                      do_handshake_on_connect=False)
29 ssock.do_handshake() # Start the handshake
30 print("==> Cipher used: {}".format(ssock.cipher()))
31 print("==> Server hostname: {}".format(ssock.server_hostname))
32 print("==> Server certificate:")
33 pprint.pprint(ssock.getpeercert())
34 pprint.pprint(context.get_ca_certs())
35 input("After TLS handshake. Press any key to continue ...")
36
37 # Send HTTP Request to Server
38 request = b"GET /nycu/ch/index/viewtopimage?module=websitetop HTTP/1.0\r\nHost: " + \
39         hostname.encode('utf-8') + b"\r\n\r\n"
40 ssock.sendall(request)
41
42 # Read HTTP Response from Server
43 response = ssock.recv(2048)
44 while response:
45     pprint.pprint(response.split(b"\r\n"))
46     response = ssock.recv(2048)
47
48 # Close the TLS Connection
49 ssock.shutdown(socket.SHUT_RDWR)
50 ssock.close()
```

**python3 handshake.py www.nycu.edu.tw (too long, thus only screenshot part of it)**

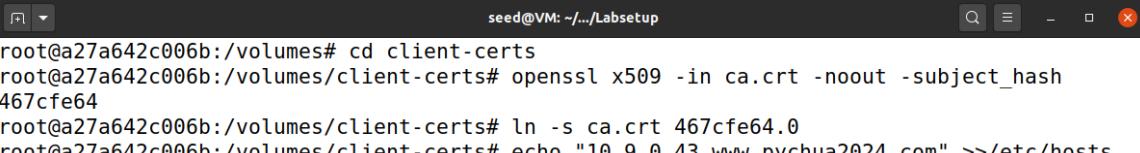


```
seed@VM: ~.../Labsetup
b'0\xd4\xe6\x90\x8\xf2+\xafE\xff\x00\x04\xe9\xbe+\xdf%\xd7\xc1\x994\xddZ'
b'X\xb5Mz\xde\t.\xc5\x95\x0cv\xdeZl\xce\xf4.\x84+*\xaf\r\xf2\xbbm\x1d\xf0'
b'\x0f\xfb\x1c\xf8\x87\xfe\x11\x83-\xbd\xf0\xaf\xb8.s\x07\x91\x13kJ'
b'\xf0\xad\xe5\xc4\xe7\xf7\x08\xe5\xcc\xaa\x7\xed\xb0\xa0\x9d\x98'
b'\xa\x92\xec?g\x1dc\xe1\x97\x84\xedu0\x12j\x9a\xcb]]\x92\xde\xd7N'
b'\x8e\xccL\xf1\xb1\xab\xc4\x8c-\xfa\x85\x920\x93<\n\xaf\x1\xbe~'
b'\xd5\x7\xf6\xf0\x96\xf9\xe0\xd7\xf8R\xf1\x87\x8ae\x83\xfe\x11'
b'\x88\xdd\xe4\x9a\xd5\xe7\xda\xa1.f\x92vi]\x11\x9c\x14E$\n\xca\x8c\x02\xd4'
b'\xe3\x80q:\xcb\x9b\xb6\xday\xb3\x8b\x0f%\xed9\x1b\xbf\xdc\xfe\xd1\x9f\xb4'
b'\xa7\xc4\x0f\x08\xe0:w\xc1\xf8l_@\xf1\x98\xb7\xruk+\xce\xfa5\x9d\xd0'
b'\xc4\x1\xe0\xf9\xd9S\x2\xbbd\xees\xbd\xbb\xf6\xd1\x9c\xbc\x0b\xf0'
b'\xb3\xf6-\xd3\xfd\x97\xaeiv\x92\xal^+\x8b\xcf0\xd6&\xdd\xb4\xdd\xdcH'
b"\xch\x19HTV\x1b\x97\x08\x5~\lf\xf9\x8f\xec\xdd\xfb8\xc7\xfb6\xe9\xbe"
b'\xf1\x07\x8d\xb5\xe9<0\xe3\x8f\x19\x05\x97\xc4\x97[\x00\xb7\xce\x0c'
b'\p\x4\xb2\x14i2\xc5\xc7\x1be\x9b\xb6\x15k\xe8\xcf\x0fj\xbe\x1b\xf8\x9d\xa6'
b'\x6\x88\x29"\x96\xd2\xcd\xac\xb0\xcf\x8a\x082\xdc\xf9F'
b'\xe4\xaf\x00\xc3\xd4\xd1\x88\xc4\xd1\x85\xa8\xc5\xb4\x9e\xfd\xff\x00\x80'
b'\z\x15b\xa5\x7\xe0U\xf1\x07\x824\x7f\x11\xeb\x96\x9a\xf5\xed\x9c'
b'\x5\x8f\x85m\xdb\xb4K\xd9\x1a\x17\x8aX\x83\x08\xca\x16+\x00%\x1c\xb3c'
b'\#i\xe3\xbf\x1\xf1\xc3\xf6\x8f\xd7[K6_\x03\xdd\xfb8\xea\x1b\xaf\x0f.\xb3'
[b'\x96\xb2=\xf2\xc9\xaa\x3!\x07\x11\x06\x0c\x1?\xef\x02\xe3<W'
b'\xb\xcd\xa0\x1f\x88\x9e\x11\xd4t\xb9\xb5\x0f\xedC\x94\xe69'
b'\x82\xdb\xbb>\x11\xc4\x1c\x13\xf2\x05g9!\x87Nk\xce-\r\xfc#\xd3\xff'
b'\x0g\x8f\x11\x9f0\xf0\x8d\xb6\xad\xe1\xef\xe0\xd8\xeabG\xb9V(\x9a\x90X'
b'\xfc\xc9\x06\x0e6)%T\x12\xc3\xba\xf3Y\xb8\x8RM\xf7\xf9z\xeeb\xf9n\xa0\xd5'
b'\xf4\xd4\xf1?\x0c-\xcaZ\xef\xedw\xf0\xe3\xc2\xb7\xdf\x1a\xe3\x9bX'
b'\xd7\xcb\xb9\x1b\x1\xe2K\xac\x96v;\xf4\xf9\x6R\\\'\x3\x0b\xe1B'
b'\x11\x90X\x8d\xcd\xb0\x9a\x5\xf5\xdf\xd9-1\x13\xfe\x11?\r\xf8\xfb\xc5\x9e'
b'\x18\xf0\x8f\x80\x7\xb9\xf0\xea\xd8\xc1\xd0\xbd\xe5\xf21\x920\xee\xd8I!W9'
b'\xde9\xda\x00\x08\x15\xf7\x81<\x8c\xabI\xd3\xedae\xb8[\x8e\xc5'
b'\xd\xf2\xdc\xd9,LY\xb0\xc86 \x01\xc2\xe3\x81\\\'\x93\xc6{M\xa5\xf83'
b'\x0\xf0\x86<\x99.\xd9\xa1\xbc2\xdc\xb8\x1et\x92J\xec\xdb\xc6\xcc\x10\xab\xb8'
b'\x826\x80:\xf2Ns\x0f\x99T\xfb=\xba\xda\x9d\x18Z1\xb1\x8c\xb5\xba'
b'\xb7\x99\xf2\x9e\x7\xf0\xeb\\\'\x81\xba^\x9f\x7f\x06\xbb\xf1K\xd4-XF'
b'\x91E=\xa5\xbd\xbc0\x1c_\xec\x94\xec\x9a\xdc\xc7\xe6\xc3\x19X\x0c!\x02\xbd'
b'\x87\xe1m\xde\xb1\x07\xc3;r\x1f\xbd\x0b\x0\x9b\xb8\xac\xa5'
b'\xb7\xd3\x96\xd9gE-\x5\xb0\x85\x17\xcb6~\xfaaI'\xd7\x0b\xf6\x94\xf8A\xac|"
b'\x7\x02\xeb\x1a\x885\x1d\xb3\xf1\x08\xb9\xd3\xad\x84\x4\xd8'
b'\xa\x02\x19\x10oY\x197\x10\xe8-V\xceG\x15K\xe1\xb7\xc7\r\xB\xd6g\xf0\xc7'
b'\x8a#\xbc\xd1-I\x4\xd9,63_\xc0\x96\xb6:\x2\xcd#\xdaJ\x8ef\xf9\xb3\xb7'
b'?':\x81\xcfR\x82\x7\xeM\x0\xfe\xt\xb4p\xfc\x92\\\'\x95\xfe\xdc\x1f\xb3b|"
b'j\xf8\xe3Y\xee<<\x2e2(4\xb8m\xf4\x14\xb5\x92F\xbe\xb7\xbf\xfd\xe2\x98F'
b'!X\xc6\xdd\xe6^\\\'\x4\xef\xce@R}\x83\xf6>\xf857\xc0\x1f\x83\x90\xf8z\xf2I\xae/'
```

The server will respond with the image data, which will be printed as part of the HTTP response. However, since we're not displaying the image in the client program, the output will consist of the HTTP response headers followed by the binary image data. Observed that the binary image data will be printed as a list of byte strings.

## Task 2: TLS Server

### Task 2.1. Implement a simple TLS server

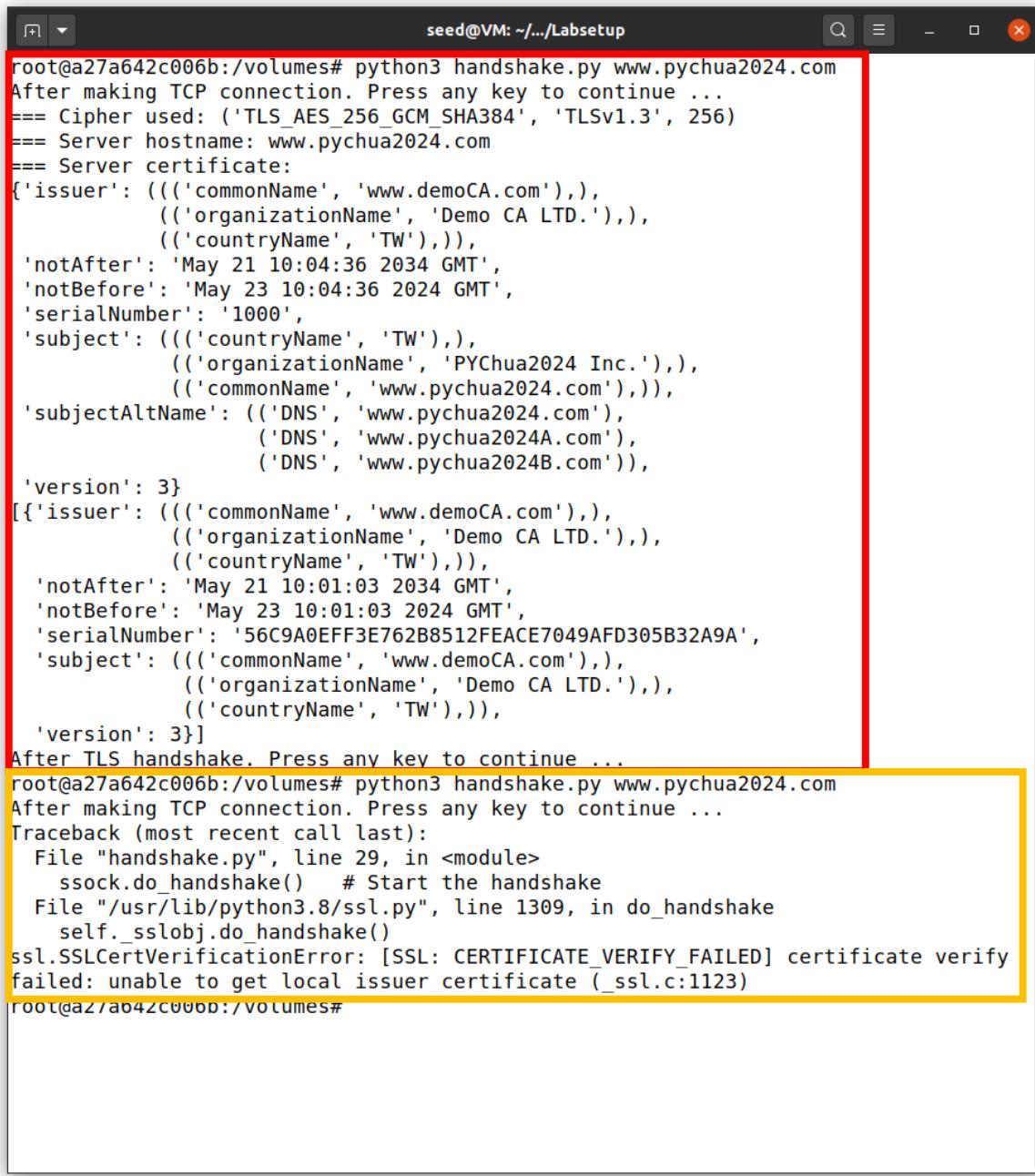


```
root@a27a642c006b:/volumes# cd client-certs
root@a27a642c006b:/volumes/client-certs# openssl x509 -in ca.crt -noout -subject_hash
467cf64
root@a27a642c006b:/volumes/client-certs# ln -s ca.crt 467cf64.0
root@a27a642c006b:/volumes/client-certs# echo "10.9.0.43 www.pychua2024.com" >>/etc/hosts
```

**Server:**

```
seed@VM: ~/.../volumes
[05/23/24] seed@VM:~/.../volumes$ dockps
0c201ef79cff  server-10.9.0.43
496dce725291  mitm-proxy-10.9.0.143
a27a642c006b  client-10.9.0.5
7d644cbf68f2  www-10.9.0.80
[05/23/24] seed@VM:~/.../volumes$ docksh 0c
root@0c201ef79cff:/# cd volumes
root@0c201ef79cff:/volumes# python3 server.py
Enter PEM pass phrase:
TLS connection established
"Request: b''"
TLS connection fails
TLS connection fails
```

## Client:

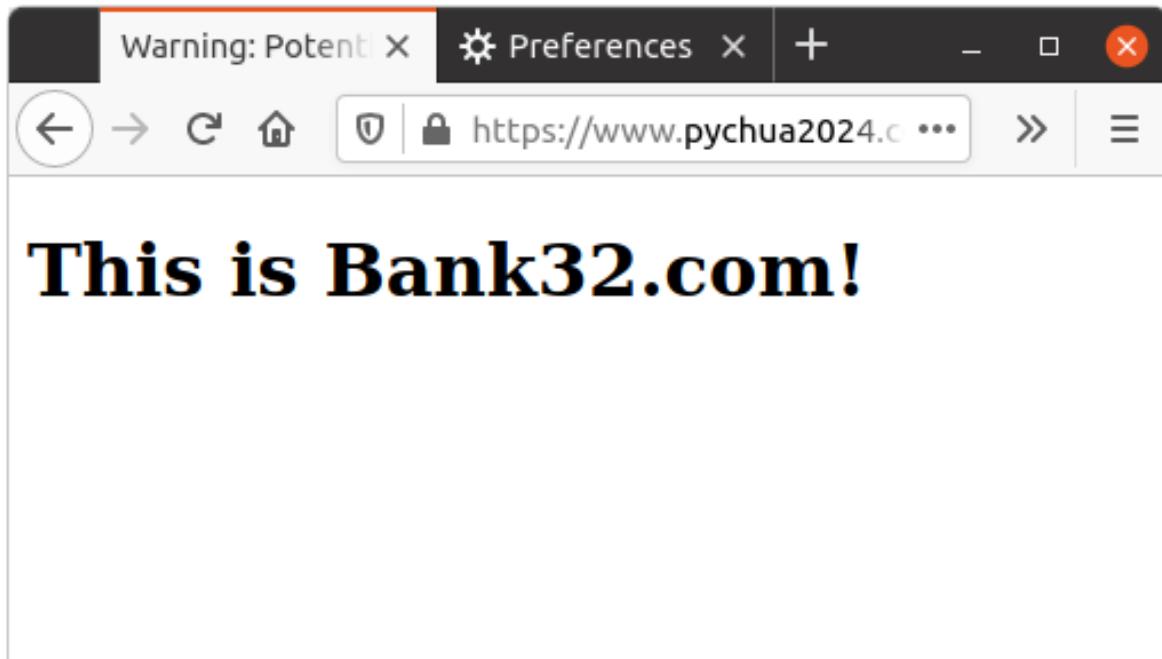


```
seed@VM: ~/.../Labsetup
root@a27a642c006b:/volumes# python3 handshake.py www.pychua2024.com
After making TCP connection. Press any key to continue ...
==> Cipher used: ('TLS_AES_256_GCM_SHA384', 'TLSv1.3', 256)
==> Server hostname: www.pychua2024.com
==> Server certificate:
{'issuer': (((('commonName', 'www.demoCA.com'),),
              (('organizationName', 'Demo CA LTD.'),),
              (('countryName', 'TW'))),
             'notAfter': 'May 21 10:04:36 2034 GMT',
             'notBefore': 'May 23 10:04:36 2024 GMT',
             'serialNumber': '1000',
             'subject': (((('countryName', 'TW'),),
                           (('organizationName', 'PYChua2024 Inc.'),),
                           (('commonName', 'www.pychua2024.com'))),
             'subjectAltName': (('DNS', 'www.pychua2024.com'),
                               ('DNS', 'www.pychua2024A.com'),
                               ('DNS', 'www.pychua2024B.com')),
             'version': 3}
[{'issuer': (((('commonName', 'www.demoCA.com'),),
              (('organizationName', 'Demo CA LTD.'),),
              (('countryName', 'TW'))),
             'notAfter': 'May 21 10:01:03 2034 GMT',
             'notBefore': 'May 23 10:01:03 2024 GMT',
             'serialNumber': '56C9A0EFF3E762B8512FEACE7049AFD305B32A9A',
             'subject': (((('commonName', 'www.demoCA.com'),),
                           (('organizationName', 'Demo CA LTD.'),),
                           (('countryName', 'TW'))),
             'version': 3}]
After TLS handshake. Press any key to continue ...
root@a27a642c006b:/volumes# python3 handshake.py www.pychua2024.com
After making TCP connection. Press any key to continue ...
Traceback (most recent call last):
  File "handshake.py", line 29, in <module>
    ssock.do_handshake()  # Start the handshake
  File "/usr/lib/python3.8/ssl.py", line 1309, in do_handshake
    self._sslobj.do_handshake()
ssl.SSLCertVerificationError: [SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed: unable to get local issuer certificate (_ssl.c:1123)
root@a27a642c006b:/volumes#
```

Red box indicates when `cadir = './client-certs'`; orange box indicates when `cadir = '/etc/ssl/certs'`.

Cannot find a certificate for `www.pychua2024.com` in `cadir = '/etc/ssl/certs'` thus no TLS connection.

## Task 2.2. Testing the server program using browsers



```
seed@VM: ~/.../volumes
root@0c201ef79cff:/volumes# python3 server.py
Enter PEM pass phrase:
TLS connection established
("Request: b'GET / HTTP/1.1\r\nHost: www.pychua2024.com\r\nUser-Agent: "
'Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:83.0) Gecko/20100101 '
'Firefox/83.0\r\nAccept: '
'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8\r\nAccept-Language: '
'en-US,en;q=0.5\r\nAccept-Encoding: gzip, deflate, br\r\nConnection: '
'keep-alive\r\nUpgrade-Insecure-Requests: 1\r\nCache-Control: '
"max-age=0\r\n\r\n")
```

## Task 2.3. Certificate with multiple names

Server:

```
seed@VM: ~
root@0c201ef79cff:/volumes# python3 server.py
Enter PEM pass phrase:
TLS connection established
"Request: b'GET / HTTP/1.0\r\nHost: www.pychua2024.com\r\n\r\n'"
TLS connection established
"Request: b'GET / HTTP/1.0\r\nHost: www.pychua2024A.com\r\n\r\n'"
TLS connection established
"Request: b'GET / HTTP/1.0\r\nHost: www.pychua2024B.com\r\n\r\n'"
```

**Client:**

```
seed@VM: ~/.../Labsetup
root@a27a642c006b:/volumes# python3 handshake.py www.pychua2024.com
After making TCP connection. Press any key to continue ...
==> Cipher used: ('TLS_AES_256_GCM_SHA384', 'TLSv1.3', 256)
==> Server hostname: www.pychua2024.com
==> Server certificate:
{'issuer': (((('commonName', 'www.demoCA.com'),),
              (('organizationName', 'Demo CA LTD.'),),
              (('countryName', 'TW'))),
             'notAfter': 'May 21 15:51:57 2034 GMT',
             'notBefore': 'May 23 15:51:57 2024 GMT',
             'serialNumber': '1003',
             'subject': (((('countryName', 'TW'),),
                           (('organizationName', 'Demo CA LTD.'),),
                           (('commonName', 'www.pychua2024.com'))),
                        'subjectAltName': ((('DNS', 'www.pychua2024.com'),
                                         ('DNS', 'www.pychua2024A.com'),
                                         ('DNS', 'www.pychua2024B.com')),
                                         'version': 3}
[{'issuer': (((('commonName', 'www.demoCA.com'),),
              (('organizationName', 'Demo CA LTD.'),),
              (('countryName', 'TW'))),
             'notAfter': 'May 21 10:01:03 2034 GMT',
             'notBefore': 'May 23 10:01:03 2024 GMT',
             'serialNumber': '56C9A0EFF3E762B8512FEACE7049AFD305B32A9A',
             'subject': (((('commonName', 'www.demoCA.com'),),
                           (('organizationName', 'Demo CA LTD.'),),
                           (('countryName', 'TW'))),
                         'version': 3}]
After TLS handshake. Press any key to continue ...
[b'\nHTTP/1.1 200 OK',
 b'Content-Type: text/html',
 b'',  
 b'\n<!DOCTYPE html><html><body><h1>This is Bank32.com!</h1></body></html>\n']
```

```
seed@VM: ~/.../Labsetup
root@a27a642c006b:/volumes# python3 handshake.py www.pychua2024A.com
After making TCP connection. Press any key to continue ...
==> Cipher used: ('TLS_AES_256_GCM_SHA384', 'TLSv1.3', 256)
==> Server hostname: www.pychua2024A.com
==> Server certificate:
{'issuer': (((('commonName', 'www.demoCA.com'),),
              (('organizationName', 'Demo CA LTD.'),),
              (('countryName', 'TW'))),
             'notAfter': 'May 21 15:51:57 2034 GMT',
             'notBefore': 'May 23 15:51:57 2024 GMT',
             'serialNumber': '1003',
             'subject': (((('countryName', 'TW'),),
                           (('organizationName', 'Demo CA LTD.'),),
                           (('commonName', 'www.pychua2024.com'))),
             'subjectAltName': (('DNS', 'www.pychua2024.com'),
                               ('DNS', 'www.pychua2024A.com'),
                               ('DNS', 'www.pychua2024B.com')),
             'version': 3}
[{'issuer': (((('commonName', 'www.demoCA.com'),),
              (('organizationName', 'Demo CA LTD.'),),
              (('countryName', 'TW'))),
             'notAfter': 'May 21 10:01:03 2034 GMT',
             'notBefore': 'May 23 10:01:03 2024 GMT',
             'serialNumber': '56C9A0EFF3E762B8512FEACE7049AFD305B32A9A',
             'subject': (((('commonName', 'www.demoCA.com'),),
                           (('organizationName', 'Demo CA LTD.'),),
                           (('countryName', 'TW'))),
             'version': 3}]
After TLS handshake. Press any key to continue ...
[b'\nHTTP/1.1 200 OK',
 b'Content-Type: text/html',
 b '',
 b'\n<!DOCTYPE html><html><body><h1>This is Bank32.com!</h1></body></html>\n']
```

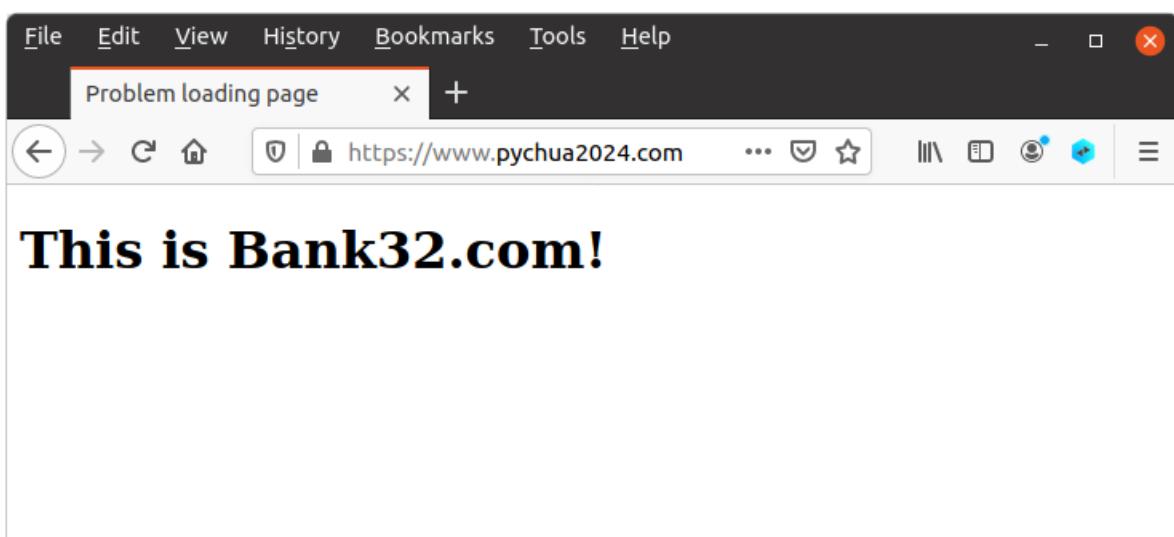
```
seed@VM: ~/.../Labsetup
root@a27a642c006b:/volumes# python3 handshake.py www.pychua2024B.com
After making TCP connection. Press any key to continue ...
==> Cipher used: ('TLS_AES_256_GCM_SHA384', 'TLSv1.3', 256)
==> Server hostname: www.pychua2024B.com
==> Server certificate:
{'issuer': (((('commonName', 'www.demoCA.com'),),
              (('organizationName', 'Demo CA LTD.'),),
              (('countryName', 'TW'))),
             'notAfter': 'May 21 15:51:57 2034 GMT',
             'notBefore': 'May 23 15:51:57 2024 GMT',
             'serialNumber': '1003',
             'subject': (((('countryName', 'TW'),),
                           (('organizationName', 'Demo CA LTD.'),),
                           (('commonName', 'www.pychua2024.com'))),
             'subjectAltName': (('DNS', 'www.pychua2024.com'),
                               ('DNS', 'www.pychua2024A.com'),
                               ('DNS', 'www.pychua2024B.com')),
             'version': 3}
[{'issuer': (((('commonName', 'www.demoCA.com'),),
              (('organizationName', 'Demo CA LTD.'),),
              (('countryName', 'TW'))),
             'notAfter': 'May 21 10:01:03 2034 GMT',
             'notBefore': 'May 23 10:01:03 2024 GMT',
             'serialNumber': '56C9A0EFF3E762B8512FEACE7049AFD305B32A9A',
             'subject': (((('commonName', 'www.demoCA.com'),),
                           (('organizationName', 'Demo CA LTD.'),),
                           (('countryName', 'TW'))),
             'version': 3}]
After TLS handshake. Press any key to continue ...
[b'\nHTTP/1.1 200 OK',
 b'Content-Type: text/html',
 b '',
 b'\n<!DOCTYPE html><html><body><h1>This is Bank32.com!</h1></body></html>\n']
```

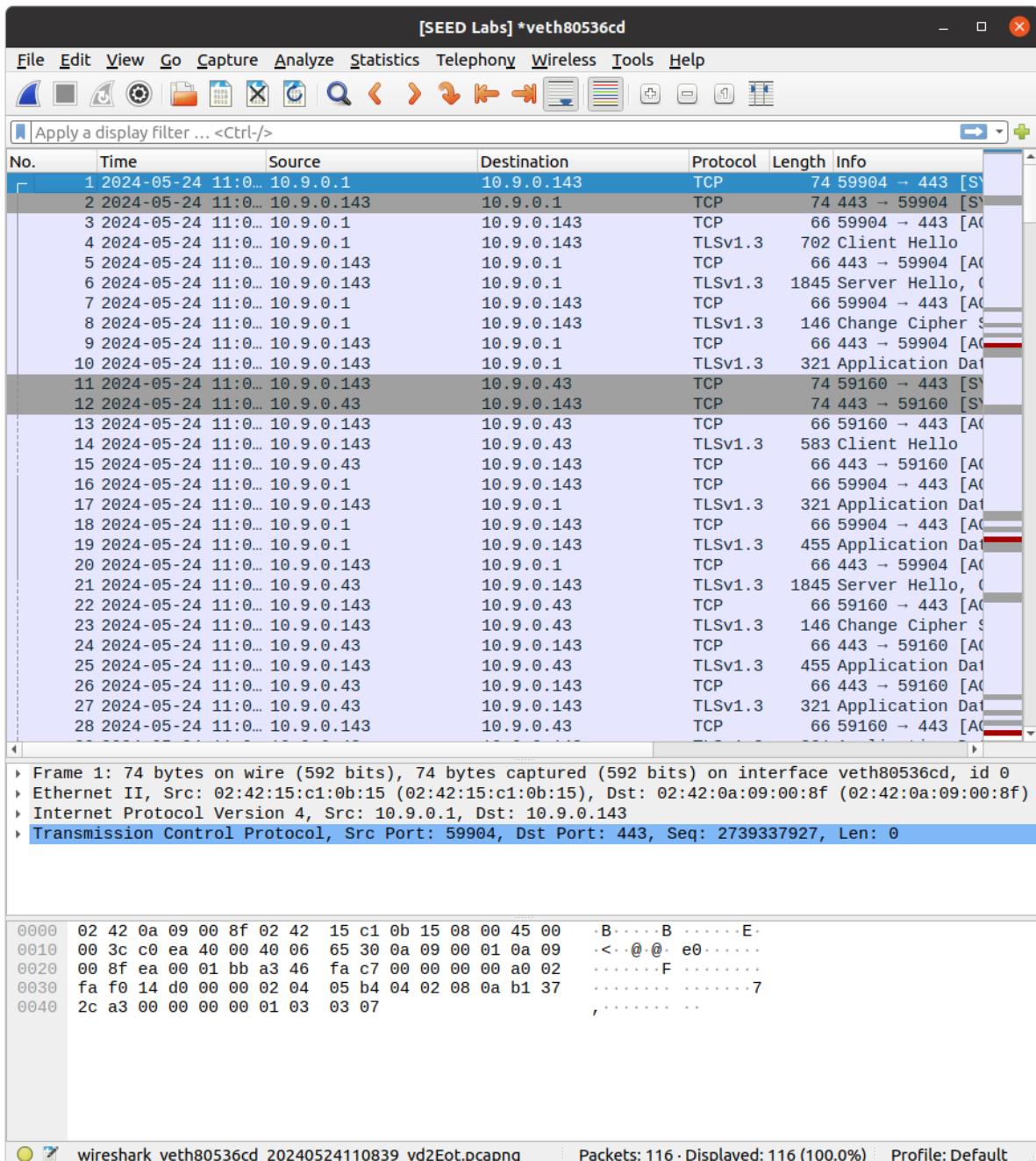
### Task 3: A Simple HTTPS Proxy

MITM attack against own server:

```
root@0c201ef79cff:/volumes# python3 server.py
Enter PEM pass phrase:
TLS connection established
("Request: b'GET / HTTP/1.1\r\nHost: www.pychua2024.com\r\nUser-Agent: "
'Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:83.0) Gecko/20100101 '
'Firefox/83.0\r\nAccept: '
'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8\r\nAccept-Language: '
'en-US,en;q=0.5\r\nAccept-Encoding: gzip, deflate, br\r\nConnection: '
'keep-alive\r\nUpgrade-Insecure-Requests: 1\r\nCache-Control: '
"max-age=0\r\n\r\n\r\n")
TLS connection established
("Request: b'GET / HTTP/1.1\r\nHost: www.pychua2024.com\r\nUser-Agent: "
'Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:83.0) Gecko/20100101 '
'Firefox/83.0\r\nAccept: '
'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8\r\nAccept-Language: '
'en-US,en;q=0.5\r\nAccept-Encoding: gzip, deflate, br\r\nConnection: '
'keep-alive\r\nUpgrade-Insecure-Requests: 1\r\nCache-Control: '
"max-age=0\r\n\r\n\r\n")
TLS connection established
("Request: b'GET / HTTP/1.1\r\nHost: www.pychua2024.com\r\nUser-Agent: "
'Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:83.0) Gecko/20100101 '
```

```
seed@VM: ~.../volumes
root@496dce725291:/volumes# python3 proxy.py
Enter PEM pass phrase:
listening
Connection accepted from ('10.9.0.1', 59904)
sock_for_server
Connection accepted from ('10.9.0.1', 59910)
sock_for_server
Connection accepted from ('10.9.0.1', 59914)
sock_for_server
```





## MITM attack on a real HTTPS website:

Originally tried with www.yahoo.com, however unable to login because of security concern

```
seed@VM: ~/.../volumes$ openssl req -newkey rsa:2048 -subj "/CN=www.yahoo.com/0" -PYChua2024 Inc./C=TW" -sha256 -keyout yahoo.key -out yahoo.csr -passout pass:dees
Generating a RSA private key
.....+++++
.....+++++
writing new private key to 'yahoo.key'
-----
[05/24/24]seed@VM:~/.../volumes$ openssl ca -md sha256 -days 3650 -config ./openssl.cnf -policy policy_anything -batch -in yahoo.csr -out yahoo.crt -cert ca.crt -keyfile ca.key
Using configuration from ./openssl.cnf
Enter pass phrase for ca.key:
Check that the request matches the signature
Signature ok
Certificate Details:
    Serial Number: 4111 (0x100f)
    Validity
        Not Before: May 24 08:42:36 2024 GMT
        Not After : May 22 08:42:36 2034 GMT
    Subject:
        countryName          = TW
        organizationName     = PYChua2024 Inc.
        commonName           = www.yahoo.com
X509v3 extensions:
    X509v3 Basic Constraints:
```

```
seed@VM: ~/.../volumes
root@496dce725291:/volumes# python3 proxy.py
Enter PEM pass phrase:
listening
Connection accepted from ('10.9.0.1', 58234)
sock_for_server
Connection accepted from ('10.9.0.1', 58294)
sock_for_server
```



Hence, tried with codeforces.com

```
[05/24/24]seed@VM:~/.../volumes$ openssl req -newkey rsa:2048 -subj "/CN=codeforces.com/0=PYChua2024 Inc./C=TW" -sha256 -keyout cf.key -out cf.csr -passout pass:dees
Generating a RSA private key
+++++
writing new private key to 'cf.key'
-----
[05/24/24]seed@VM:~/.../volumes$ openssl ca -md sha256 -days 3650 -config ./openssl.cnf -policy policy_anything -batch -in cf.csr -out cf.crt -cert ca.crt -keyfile ca.key
Using configuration from ./openssl.cnf
Enter pass phrase for ca.key:
Check that the request matches the signature
Signature ok
Certificate Details:
    Serial Number: 4114 (0x1012)
    Validity
        Not Before: May 24 09:10:10 2024 GMT
        Not After : May 22 09:10:10 2034 GMT
    Subject:
        countryName          = TW
        organizationName     = PYChua2024 Inc.
        commonName           = codeforces.com
X509v3 extensions:
X509v3 Basic Constraints:
```

```
[+] seed@VM: ~/.../volumes
root@496dce725291:/volumes# python3 proxy.py
Enter PEM pass phrase:
listening
Connection accepted from ('10.9.0.1', 59418)
sock_for_server
Connection accepted from ('10.9.0.1', 59432)
Connection accepted from ('10.9.0.1', 59434)
Connection accepted from ('10.9.0.1', 59436)
Connection accepted from ('10.9.0.1', 59438)
```

The Wireshark interface displays a list of network frames. The 16th frame is highlighted in blue. The details pane shows the frame structure:

```

> Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface veth80536cd, id 0
> Ethernet II, Src: 02:42:15:c1:0b:15 (02:42:15:c1:0b:15), Dst: 02:42:0a:09:00:8f (02:42:0a:09:00:8f)
> Internet Protocol Version 4, Src: 10.9.0.1, Dst: 10.9.0.143
> Transmission Control Protocol, Src Port: 58234, Dst Port: 443, Seq: 310058897, Len: 0

```

The bytes pane shows the raw hex and ASCII data for the selected frame.

Codeforces

https://codeforces.com

# CODEFORCES

Sponsored by TON

Enter | Register

HOME TOP CATALOG CONTESTS GYM PROBLEMSET GROUPS RATING EDU API CALENDAR HELP

## ICPC Alumni Event in Silicon Valley 2024

By Temirulan, [history](#), 47 hours ago,

We invite you to first ICPC alumni event in Silicon Valley on June 4 at 6pm.

This is a unique event that will bring together top ICPC alumni and friends. The event will be attended by ICPC President Bill Poucher, Freedom Holding CEO Timur Turlov and Bagdat Mussin, President of the Kazakh Federation of Competitive Programming.

Register here: [https://lu.ma/ICPC\\_Meetup](https://lu.ma/ICPC_Meetup)

P.S: wdty about some small entertainy contest?

[Full text and comments »](#)

icpc

+152

Temirulan 47 hours ago 22

### Pay attention

Before contest  
[Codeforces Round \(Div. 1 + Div. 2\)](#)  
27:08:45  
[Register now »](#)  
\*has extra registration

### Streams

[Round 1975 Div 1 + Div 2 Solution Discussion \(with Jan\)](#)  
By Shayan  
Before stream 30:13:45  
[View all →](#)

### Top rated

#	User	Rating
1	tourist	3690
2	jiangly	3647
3	Benq	3581
4	orzdevinwang	3570
5	Geothermal	3569
5	cnnfis_csy	3569
7	Radewoosh	3509
8	ecnerwala	3486
9	jqdai0815	3474
10	gyh20	3447

[Countries](#) | [Cities](#) | [Organizations](#) [View all →](#)

### Top contributors

#	User	Contrib.
1	maomao90	171
2	adaman	164
3	awoo	163

## Codeforces Round #946 (Div. 3)

By Vladoeiva, [history](#), 5 days ago, translation

File Edit View History Bookmarks Tools Help

Login - Codeforces +

https://codeforces.com/enter?back=%2Fenter%2F

CODEFORCES Sponsored by TON

Enter | Register

HOME TOP CATALOG CONTESTS GYM PROBLEMSET GROUPS RATING EDU API CALENDAR HELP

Fill in the form to login into Codeforces.  
You can use [Gmail](#) as an alternative way to enter.

**Login into Codeforces**

Handle/Email

Password

Remember me for a month

**Login**

[Forgot your password?](#)

[Use Gmail](#)

[Codeforces](#) (c) Copyright 2010-2024 Mike Mirzayanov  
The only programming contests Web 2.0 platform  
Server time: May/24/2024 07:27:31<sub>UTC-4</sub> (k2).  
Desktop version, switch to [mobile version](#).  
[Privacy Policy](#)

Supported by

 ITMO UNIVERSITY