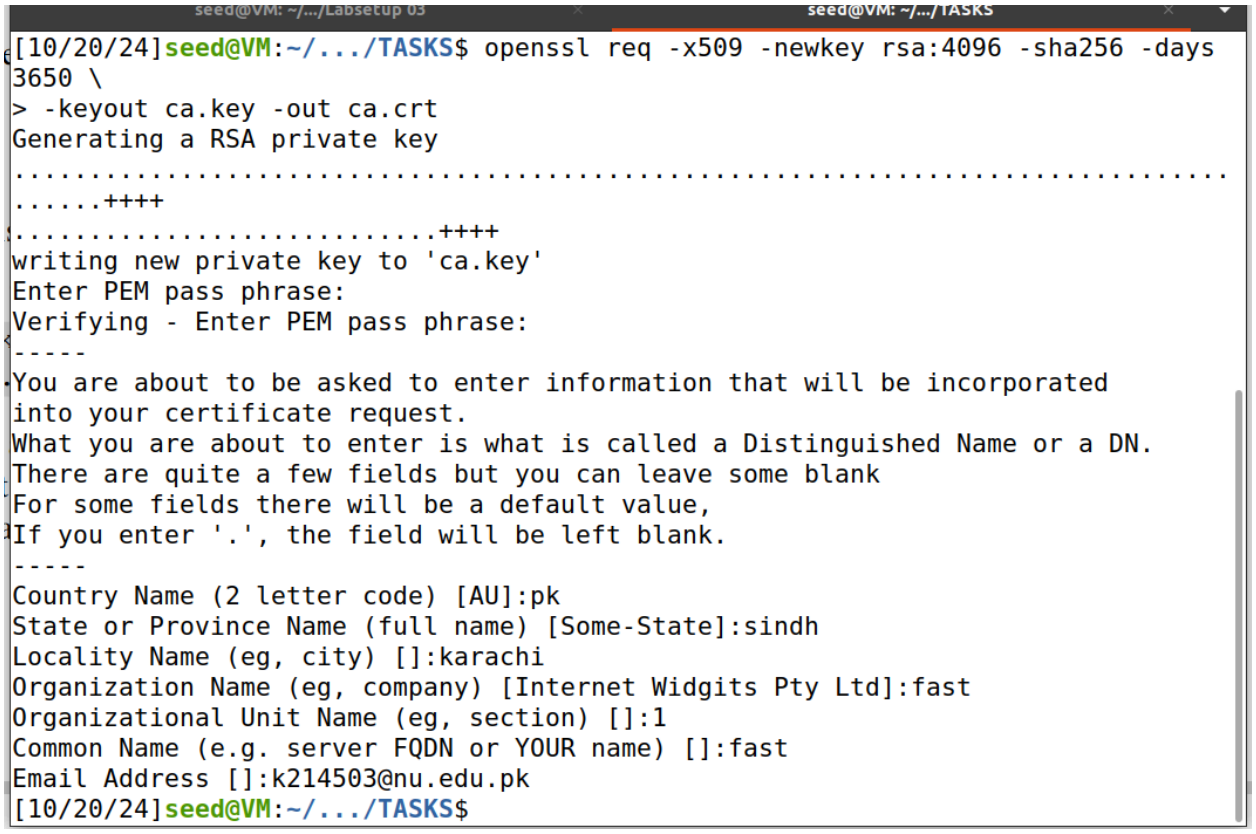
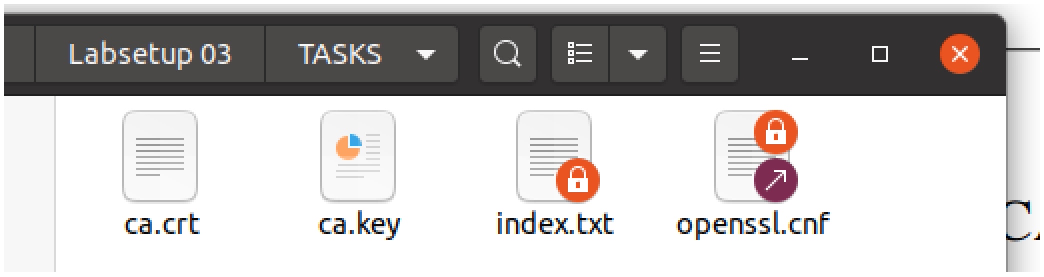
IS Assignment 03

K213279 Insha Javed
K214503 Muhammad Tahir

Task01:

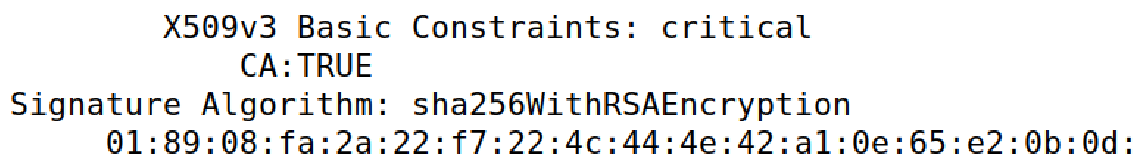


```
[10/20/24]seed@VM:~/.../TASKS$ openssl req -x509 -newkey rsa:4096 -sha256 -days
3650 \
> -keyout ca.key -out ca.crt
Generating a RSA private key
................................................................................
......++++
............................++++
writing new private key to 'ca.key'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
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]:pk
State or Province Name (full name) [Some-State]:sindh
Locality Name (eg, city) []:karachi
Organization Name (eg, company) [Internet Widgits Pty Ltd]:fast
Organizational Unit Name (eg, section) []:1
Common Name (e.g. server FQDN or YOUR name) []:fast
Email Address []:k214503@nu.edu.pk
[10/20/24]seed@VM:~/.../TASKS$
```



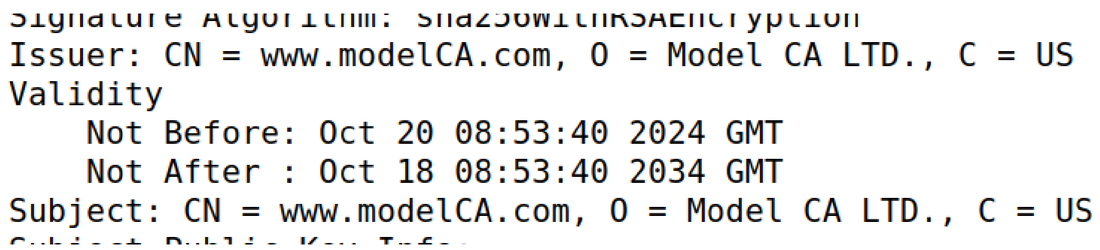
From the output, please identify the following:

• What part of the certificate indicates this is a CA's certificate?



```
            X509v3 Basic Constraints: critical
                CA:TRUE
    Signature Algorithm: sha256WithRSAEncryption
         01:89:08:fa:2a:22:f7:22:4c:44:4e:42:a1:0e:65:e2:0b:0d:
```

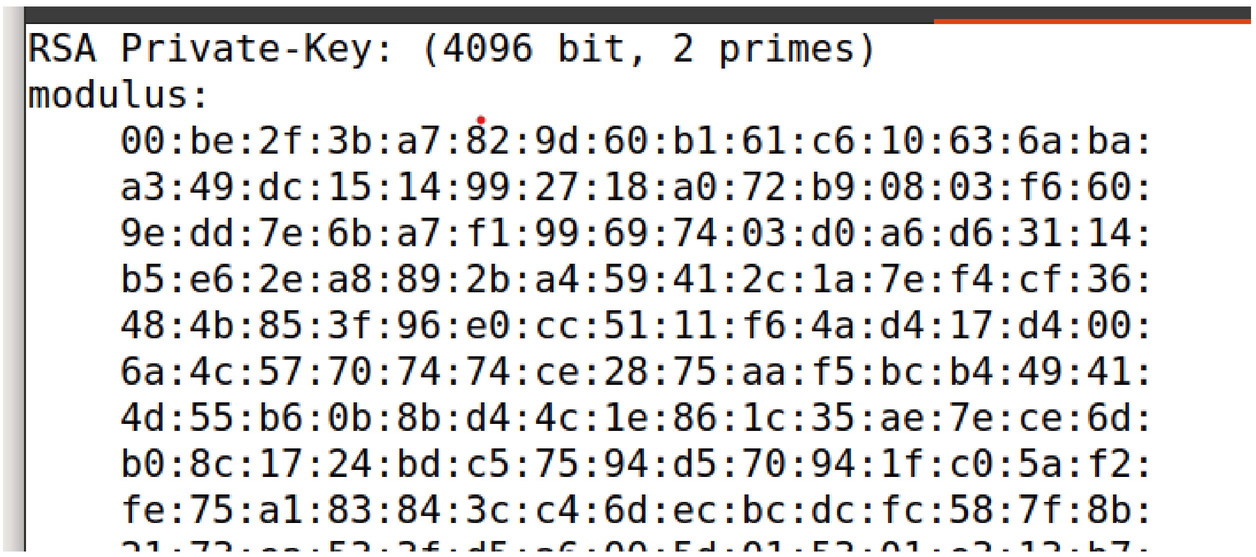
• What part of the certificate indicates this is a self-signed certificate?



```
    Signature Algorithm: sha256WithRSAEncryption
    Issuer: CN = www.modelCA.com, O = Model CA LTD., C = US
    Validity
        Not Before: Oct 20 08:53:40 2024 GMT
        Not After : Oct 18 08:53:40 2034 GMT
    Subject: CN = www.modelCA.com, O = Model CA LTD., C = US
```

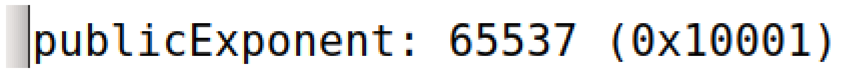
• In the RSA algorithm, we have a public exponent e, a private exponent d, a modulus n, and two secret numbers p and q, such that n = pq. Please identify the values for these elements in your certificate and key files.

Mod n:



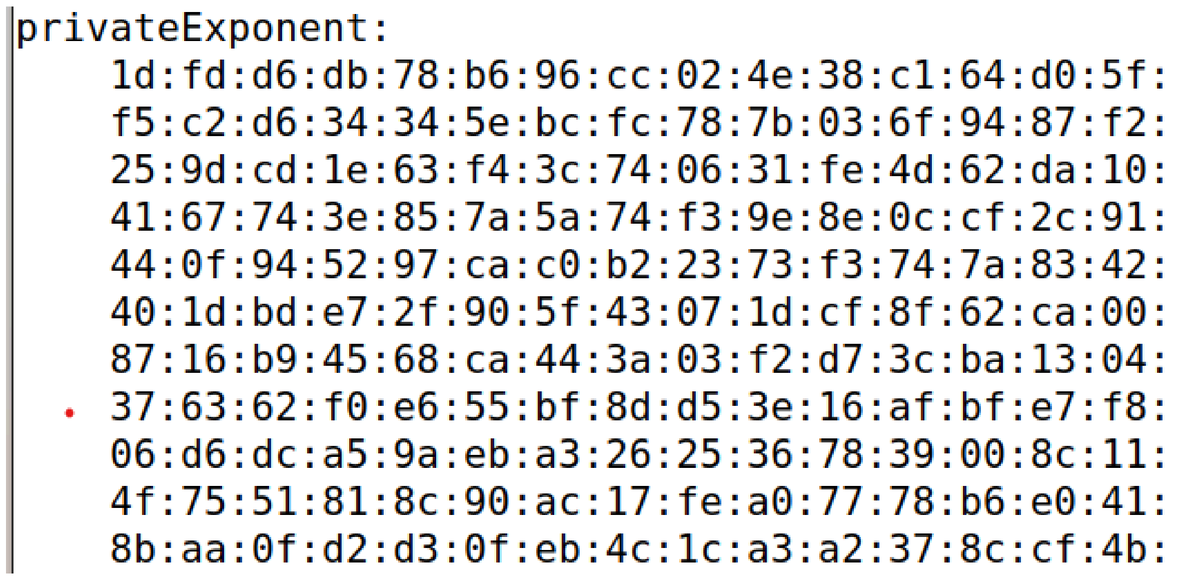
```
RSA Private-Key: (4096 bit, 2 primes)
modulus:
    00:be:2f:3b:a7:82:9d:60:b1:61:c6:10:63:6a:ba:
    a3:49:dc:15:14:99:27:18:a0:72:b9:08:03:f6:60:
    9e:dd:7e:6b:a7:f1:99:69:74:03:d0:a6:d6:31:14:
    b5:e6:2e:a8:89:2b:a4:59:41:2c:1a:7e:f4:cf:36:
    48:4b:85:3f:96:e0:cc:51:11:f6:4a:d4:17:d4:00:
    6a:4c:57:70:74:74:ce:28:75:aa:f5:bc:b4:49:41:
    4d:55:b6:0b:8b:d4:4c:1e:86:1c:35:ae:7e:ce:6d:
    b0:8c:17:24:bd:c5:75:94:d5:70:94:1f:c0:5a:f2:
    fe:75:a1:83:84:3c:c4:6d:ec:bc:dc:fc:58:7f:8b:
```

E;



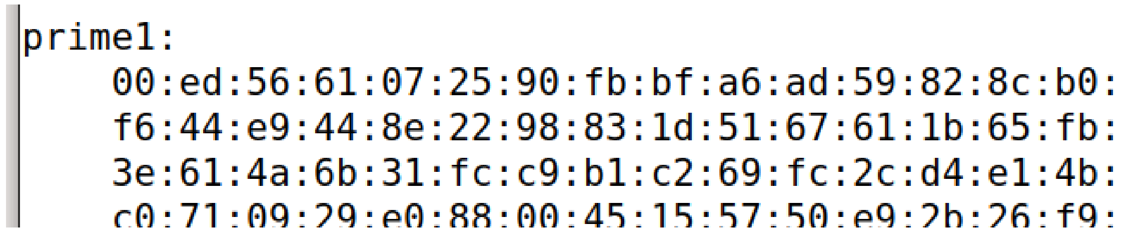
```
publicExponent: 65537 (0x10001)
```

D:



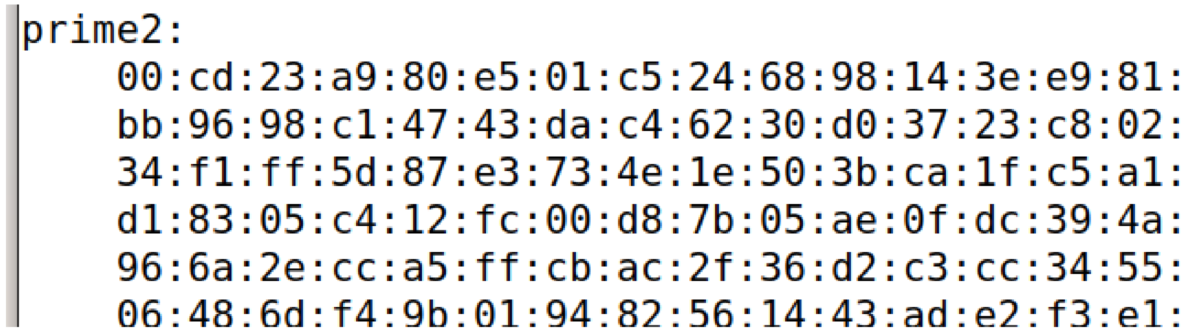
```
privateExponent:
    1d:fd:d6:db:78:b6:96:cc:02:4e:38:c1:64:d0:5f:
    f5:c2:d6:34:34:5e:bc:fc:78:7b:03:6f:94:87:f2:
    25:9d:cd:1e:63:f4:3c:74:06:31:fe:4d:62:da:10:
    41:67:74:3e:85:7a:5a:74:f3:9e:8e:0c:cf:2c:91:
    44:0f:94:52:97:ca:c0:b2:23:73:f3:74:7a:83:42:
    40:1d:bd:e7:2f:90:5f:43:07:1d:cf:8f:62:ca:00:
    87:16:b9:45:68:ca:44:3a:03:f2:d7:3c:ba:13:04:
    37:63:62:f0:e6:55:bf:8d:d5:3e:16:af:bf:e7:f8:
    06:d6:dc:a5:9a:eb:a3:26:25:36:78:39:00:8c:11:
    4f:75:51:81:8c:90:ac:17:fe:a0:77:78:b6:e0:41:
    8b:aa:0f:d2:d3:0f:eb:4c:1c:a3:a2:37:8c:cf:4b:
```

P:



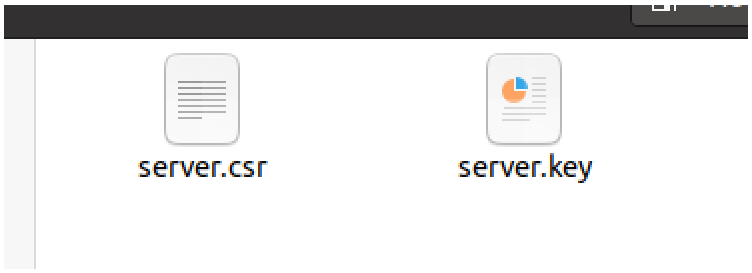
```
prime1:
    00:ed:56:61:07:25:90:fb:bf:a6:ad:59:82:8c:b0:
    f6:44:e9:44:8e:22:98:83:1d:51:67:61:1b:65:fb:
    3e:61:4a:6b:31:fc:c9:b1:c2:69:fc:2c:d4:e1:4b:
    c0:71:09:29:e0:88:00:45:15:57:50:e9:2b:26:f9:
```

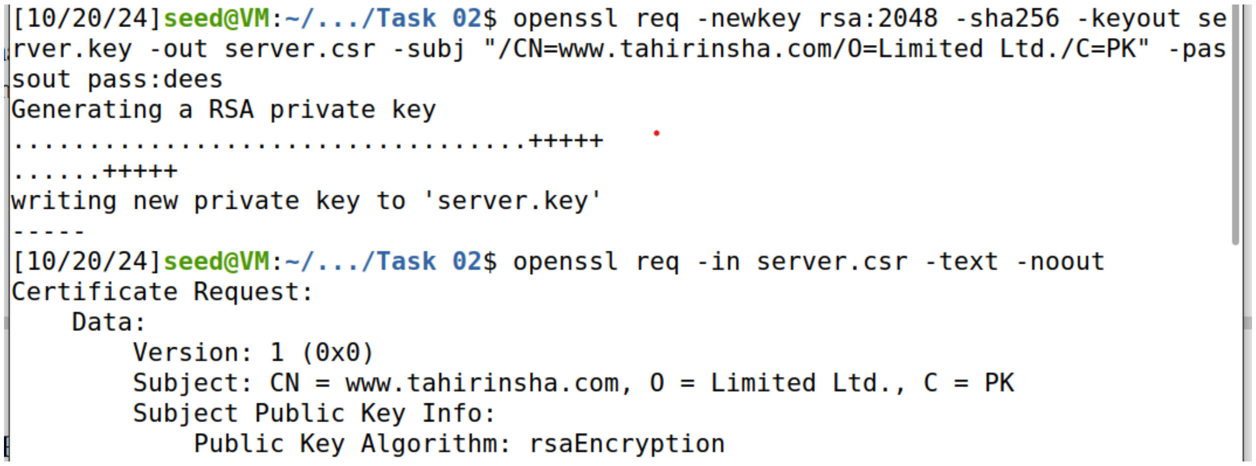
Q:



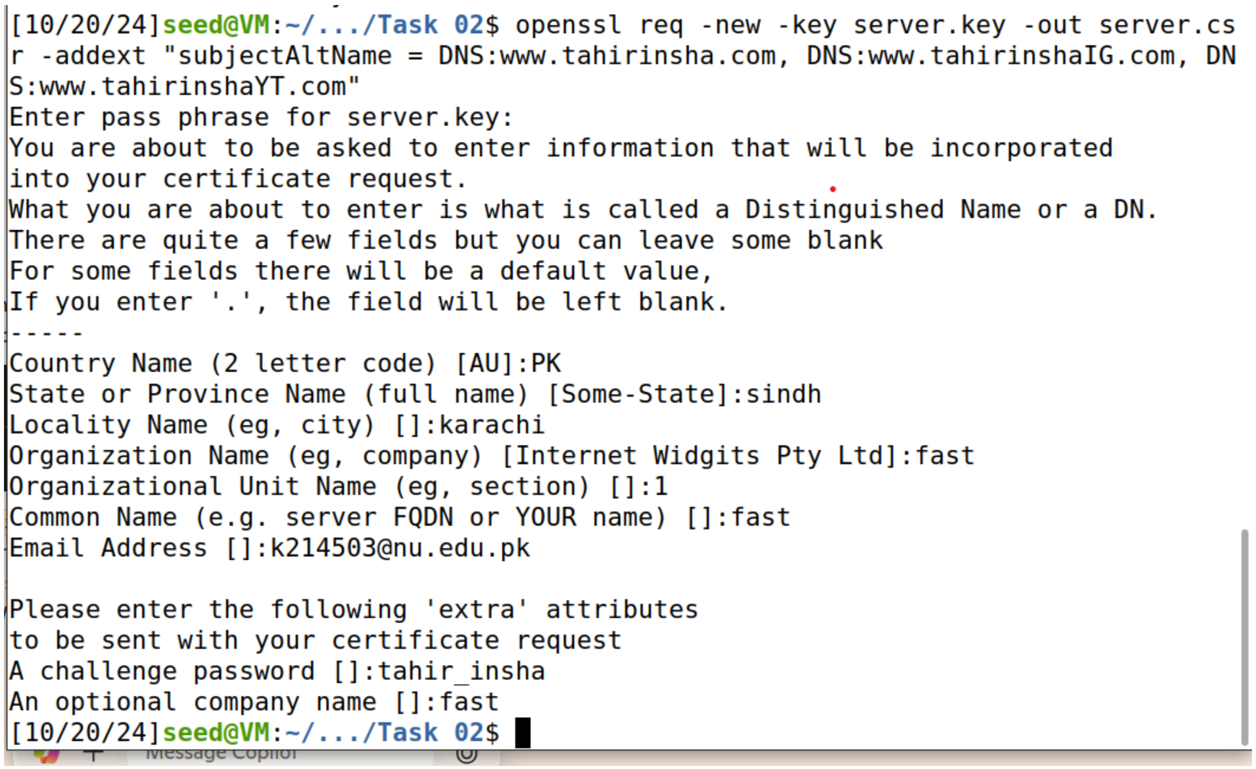
```
prime2:
    00:cd:23:a9:80:e5:01:c5:24:68:98:14:3e:e9:81:
    bb:96:98:c1:47:43:da:c4:62:30:d0:37:23:c8:02:
    34:f1:ff:5d:87:e3:73:4e:1e:50:3b:ca:1f:c5:a1:
    d1:83:05:c4:12:fc:00:d8:7b:05:ae:0f:dc:39:4a:
    96:6a:2e:cc:a5:ff:cb:ac:2f:36:d2:c3:cc:34:55:
    06:48:6d:f4:9b:01:94:82:56:14:43:ad:e2:f3:e1:
```

Task02:





```
[10/20/24]seed@VM:~/.../Task 02$ openssl req -newkey rsa:2048 -sha256 -keyout se
rver.key -out server.csr -subj "/CN=www.tahirinsha.com/O=Limited Ltd./C=PK" -pas
sout pass:dees
Generating a RSA private key
..................................+++++
......+++++
writing new private key to 'server.key'
-----
[10/20/24]seed@VM:~/.../Task 02$ openssl req -in server.csr -text -noout
Certificate Request:
    Data:
        Version: 1 (0x0)
        Subject: CN = www.tahirinsha.com, O = Limited Ltd., C = PK
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
```



```
[10/20/24]seed@VM:~/.../Task 02$ openssl req -new -key server.key -out server.cs
r -addext "subjectAltName = DNS:www.tahirinsha.com, DNS:www.tahirinshaIG.com, DN
S:www.tahirinshaYT.com"
Enter pass phrase for server.key:
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]:PK
State or Province Name (full name) [Some-State]:sindh
Locality Name (eg, city) []:karachi
Organization Name (eg, company) [Internet Widgits Pty Ltd]:fast
Organizational Unit Name (eg, section) []:1
Common Name (e.g. server FQDN or YOUR name) []:fast
Email Address []:k214503@nu.edu.pk

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:tahir_insha
An optional company name []:fast
[10/20/24]seed@VM:~/.../Task 02$
```

Observation:

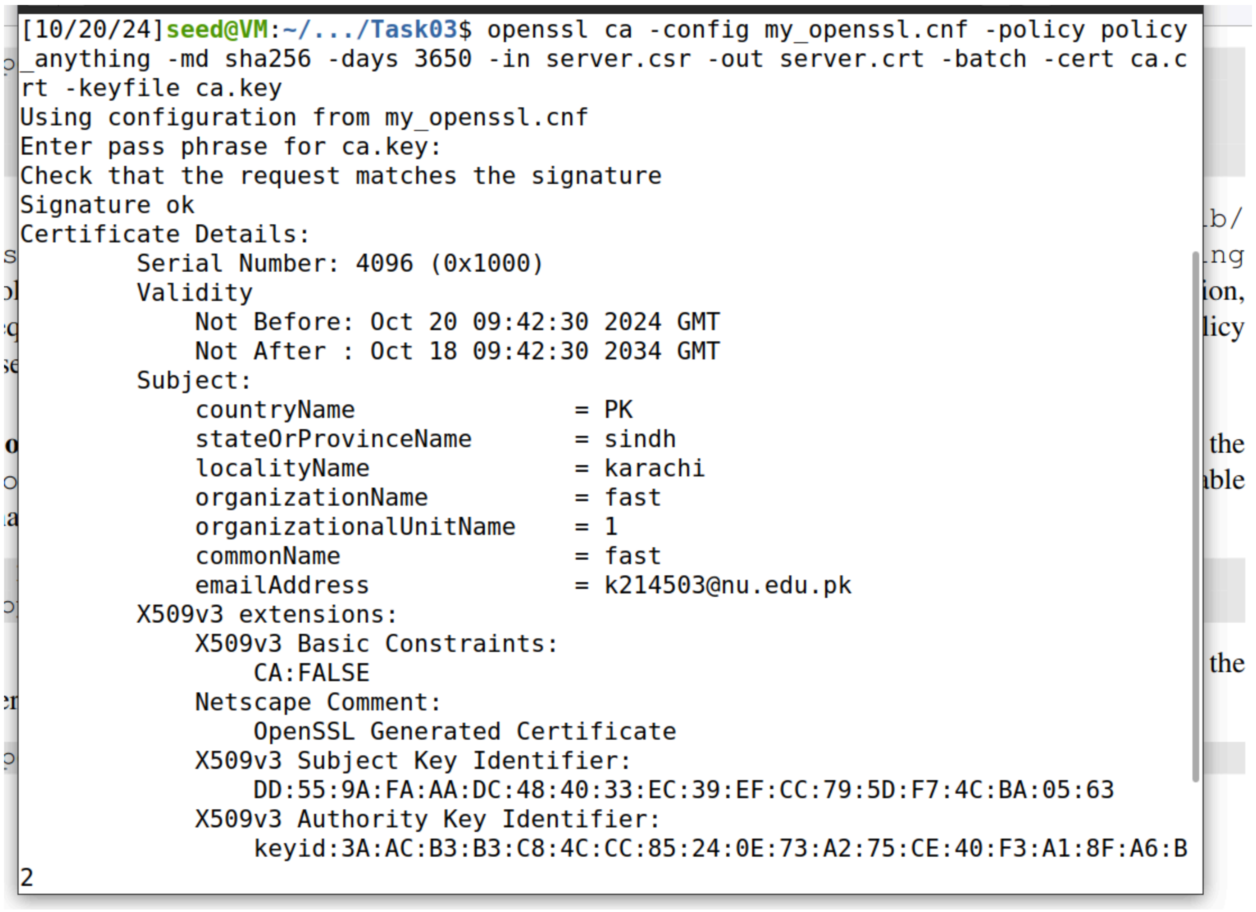
1. Generated a Certificate Signing Request (CSR) for `www.tahirinsha.com` with `openssl req`, producing `server.key` (private key) and `server.csr` (CSR).

2. Verified CSR and private key details using `openssl req` and `openssl rsa`, confirming accurate encoding of public/private keys and identity.

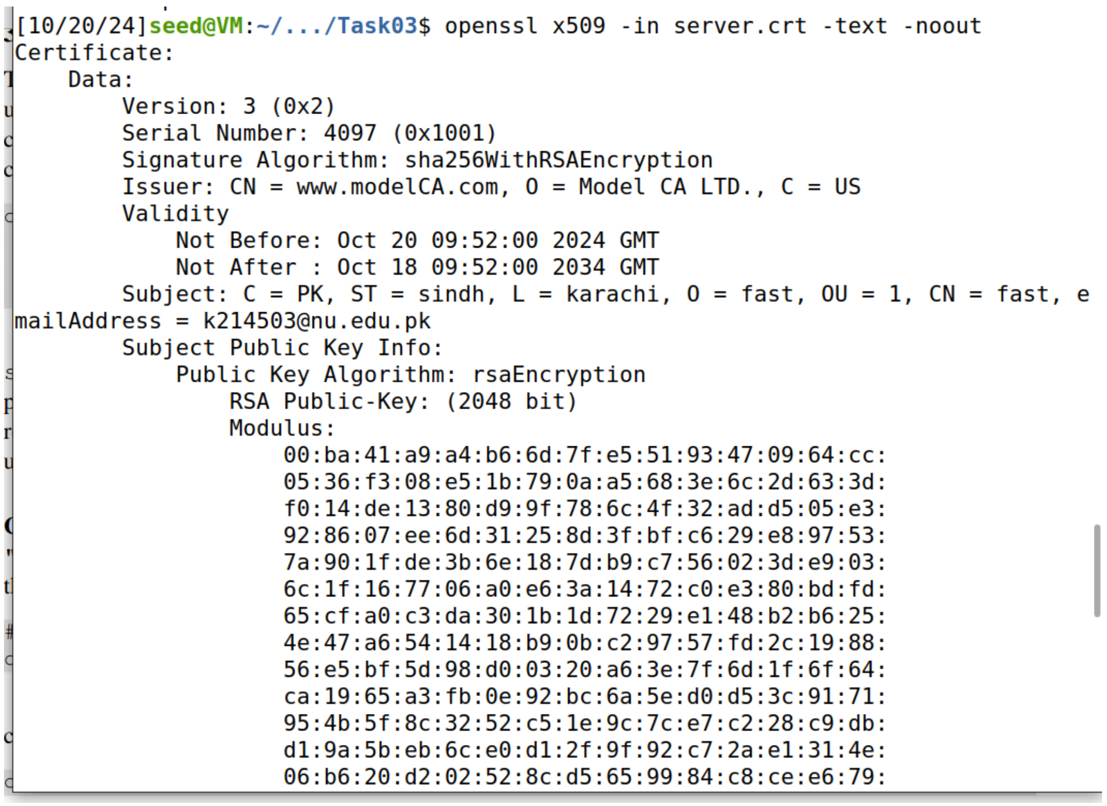
3. Added Subject Alternative Names (SAN) to the CSR with the `subjectAltName` field, ensuring compatibility with multiple server URLs.
    1. www.tahirinsha.com
    2. www.tahirinshaIG.com
    3. www.tahirinshaYT.com

4. Observed successful inclusion of Subject Alternative Names SAN extension, providing flexibility for alternative server names in the certificate.

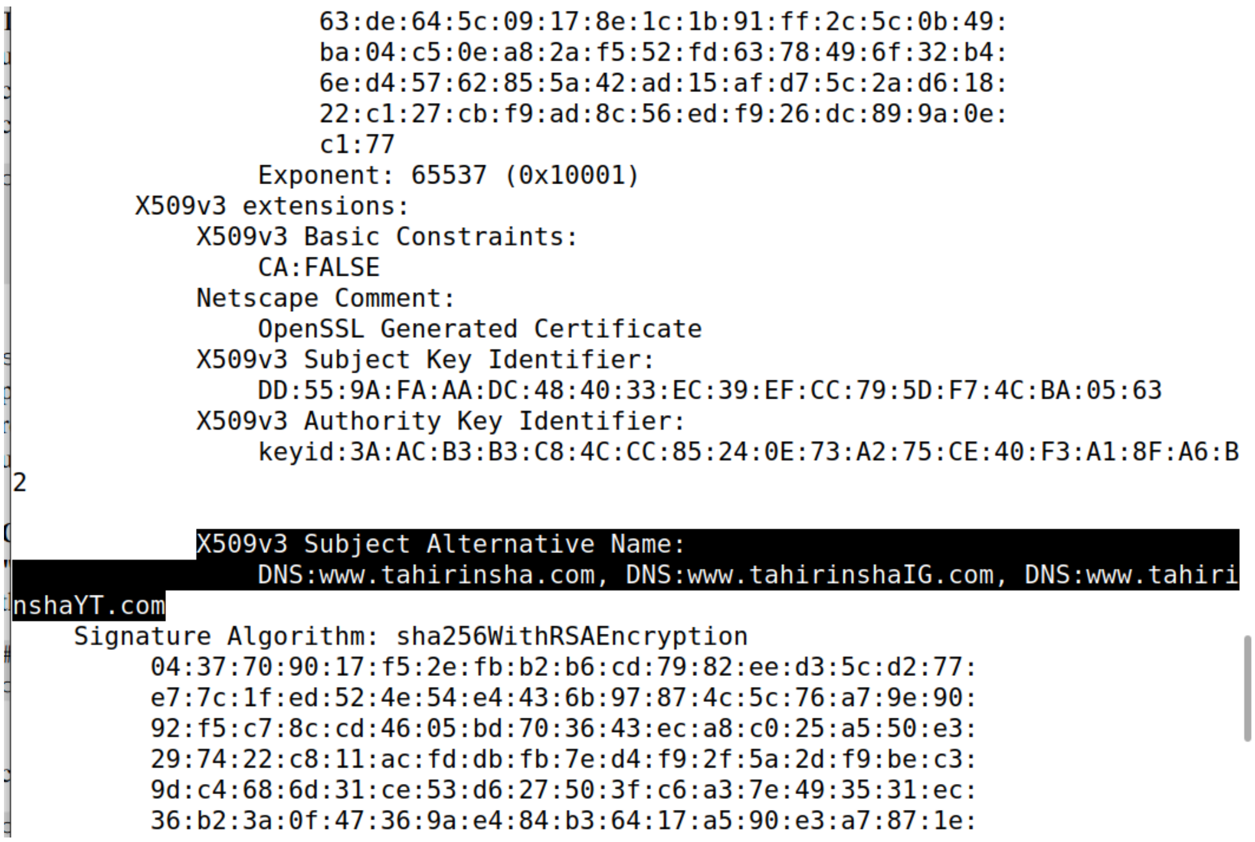
Task03:



```
[10/20/24]seed@VM:~/.../Task03$ openssl ca -config my_openssl.cnf -policy policy
_anything -md sha256 -days 3650 -in server.csr -out server.crt -batch -cert ca.c
rt -keyfile ca.key
Using configuration from my_openssl.cnf
Enter pass phrase for ca.key:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 4096 (0x1000)
        Validity
            Not Before: Oct 20 09:42:30 2024 GMT
            Not After : Oct 18 09:42:30 2034 GMT
        Subject:
            countryName               = PK
            stateOrProvinceName       = sindh
            localityName              = karachi
            organizationName          = fast
            organizationalUnitName    = 1
            commonName                = fast
            emailAddress              = k214503@nu.edu.pk
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                DD:55:9A:FA:AA:DC:48:40:33:EC:39:EF:CC:79:5D:F7:4C:BA:05:63
            X509v3 Authority Key Identifier:
                keyid:3A:AC:B3:B3:C8:4C:CC:85:24:0E:73:A2:75:CE:40:F3:A1:8F:A6:B
2
```



```
[10/20/24]seed@VM:~/.../Task03$ openssl x509 -in server.crt -text -noout
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 4097 (0x1001)
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: CN = www.modelCA.com, O = Model CA LTD., C = US
        Validity
            Not Before: Oct 20 09:52:00 2024 GMT
            Not After : Oct 18 09:52:00 2034 GMT
        Subject: C = PK, ST = sindh, L = karachi, O = fast, OU = 1, CN = fast, e
mailAddress = k214503@nu.edu.pk
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:ba:41:a9:a4:b6:6d:7f:e5:51:93:47:09:64:cc:
                    05:36:f3:08:e5:1b:79:0a:a5:68:3e:6c:2d:63:3d:
                    f0:14:de:13:80:d9:9f:78:6c:4f:32:ad:d5:05:e3:
                    92:86:07:ee:6d:31:25:8d:3f:bf:c6:29:e8:97:53:
                    7a:90:1f:de:3b:6e:18:7d:b9:c7:56:02:3d:e9:03:
                    6c:1f:16:77:06:a0:e6:3a:14:72:c0:e3:80:bd:fd:
                    65:cf:a0:c3:da:30:1b:1d:72:29:e1:48:b2:b6:25:
                    4e:47:a6:54:14:18:b9:0b:c2:97:57:fd:2c:19:88:
                    56:e5:bf:5d:98:d0:03:20:a6:3e:7f:6d:1f:6f:64:
                    ca:19:65:a3:fb:0e:92:bc:6a:5e:d0:d5:3c:91:71:
                    95:4b:5f:8c:32:52:c5:1e:9c:7c:e7:c2:28:c9:db:
                    d1:9a:5b:eb:6c:e0:d1:2f:9f:92:c7:2a:e1:31:4e:
                    06:b6:20:d2:02:52:8c:d5:65:99:84:c8:ce:e6:79:
```



```
                    63:de:64:5c:09:17:8e:1c:1b:91:ff:2c:5c:0b:49:
                    ba:04:c5:0e:a8:2a:f5:52:fd:63:78:49:6f:32:b4:
                    6e:d4:57:62:85:5a:42:ad:15:af:d7:5c:2a:d6:18:
                    22:c1:27:cb:f9:ad:8c:56:ed:f9:26:dc:89:9a:0e:
                    c1:77
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                DD:55:9A:FA:AA:DC:48:40:33:EC:39:EF:CC:79:5D:F7:4C:BA:05:63
            X509v3 Authority Key Identifier:
                keyid:3A:AC:B3:B3:C8:4C:CC:85:24:0E:73:A2:75:CE:40:F3:A1:8F:A6:B
2

            X509v3 Subject Alternative Name:
                DNS:www.tahirinsha.com, DNS:www.tahirinshaIG.com, DNS:www.tahiri
nshaYT.com
    Signature Algorithm: sha256WithRSAEncryption
         04:37:70:90:17:f5:2e:fb:b2:b6:cd:79:82:ee:d3:5c:d2:77:
         e7:7c:1f:ed:52:4e:54:e4:43:6b:97:87:4c:5c:76:a7:9e:90:
         92:f5:c7:8c:cd:46:05:bd:70:36:43:ec:a8:c0:25:a5:50:e3:
         29:74:22:c8:11:ac:fd:db:fb:7e:d4:f9:2f:5a:2d:f9:be:c3:
         9d:c4:68:6d:31:ce:53:d6:27:50:3f:c6:a3:7e:49:35:31:ec:
         36:b2:3a:0f:47:36:9a:e4:84:b3:64:17:a5:90:e3:a7:87:1e:
```

Observation:

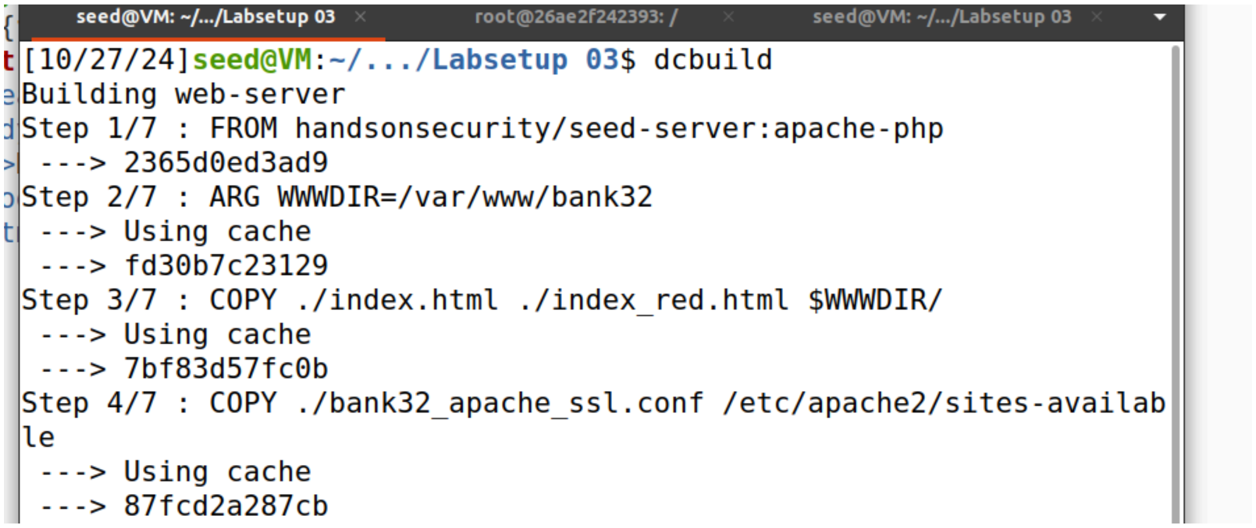
1. Created an X.509 certificate for www.tahirinsha.com by signing server.csr using our CA's ca.crt and ca.key, generating server.crt with the openssl ca command.

2. Used the policy_anything setting in myCA_openssl.cnf to bypass default restrictions, allowing unmatched subject information between the server and CA certificates.

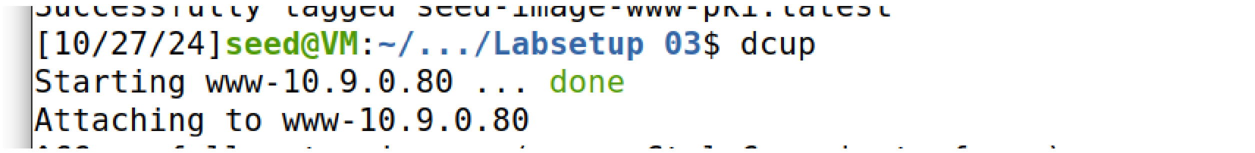
3. Enabled the copying of extension fields by uncommenting copy_extensions = copy in the configuration file, ensuring Subject Alternative Names (SANs) were preserved in the final certificate.

4. Verified server.crt details with openssl x509, confirming that SANs (`www.tahirinsha.com`, `www.tahirinshaA.com`, `www.tahirinshaB.com`) were included successfully.

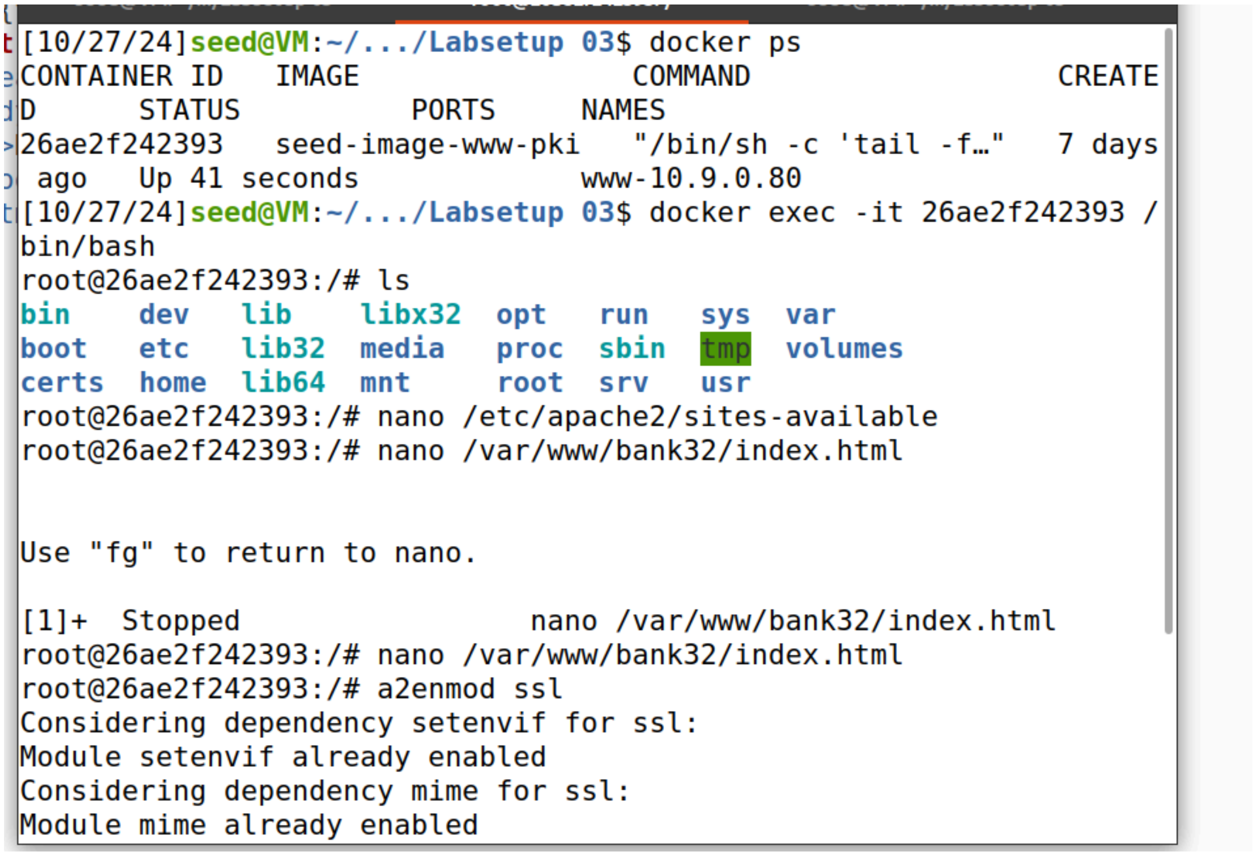
Task04:



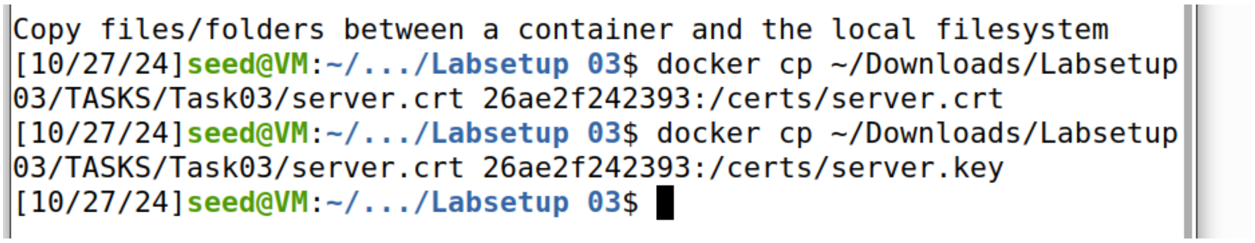
```
[10/27/24]seed@VM:~/.../Labsetup 03$ dcbuild
Building web-server
Step 1/7 : FROM handsonsecurity/seed-server:apache-php
 ---> 2365d0ed3ad9
Step 2/7 : ARG WWWDIR=/var/www/bank32
 ---> Using cache
 ---> fd30b7c23129
Step 3/7 : COPY ./index.html ./index_red.html $WWWDIR/
 ---> Using cache
 ---> 7bf83d57fc0b
Step 4/7 : COPY ./bank32_apache_ssl.conf /etc/apache2/sites-availab
le
 ---> Using cache
 ---> 87fcd2a287cb
```



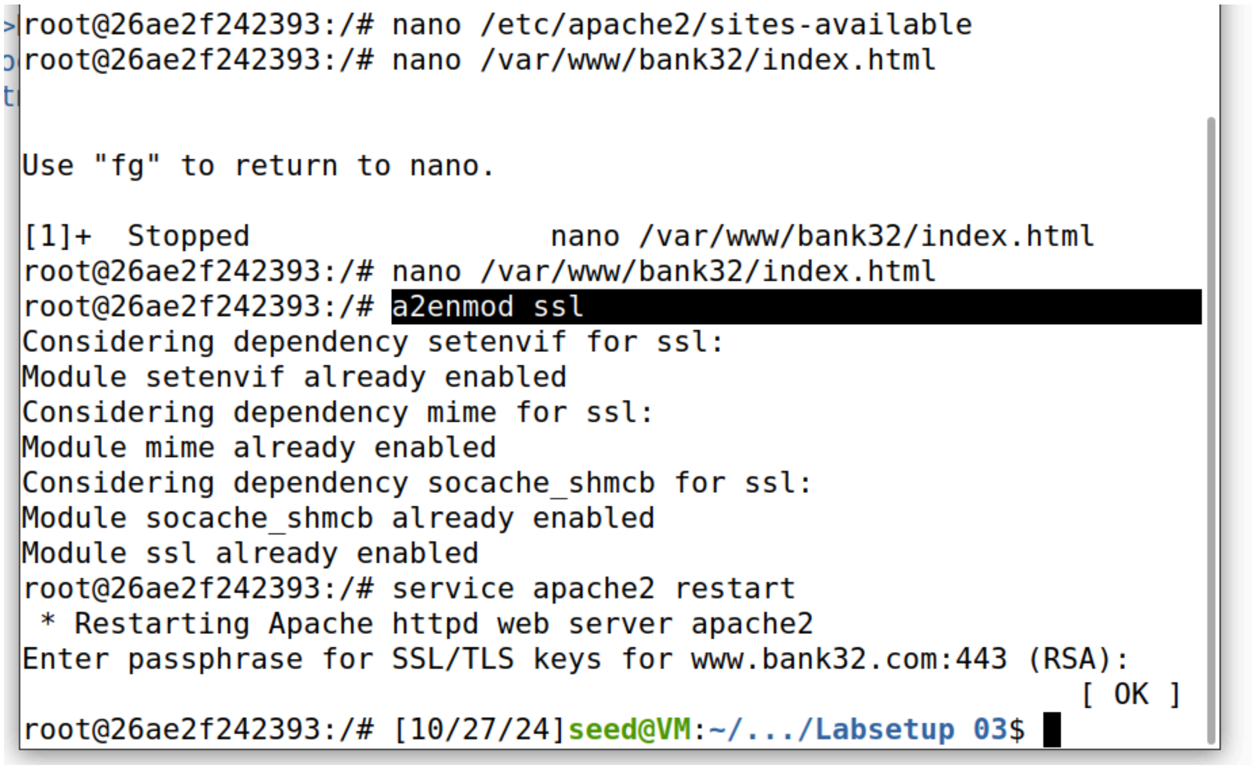
```
[10/27/24]seed@VM:~/.../Labsetup 03$ dcup
Starting www-10.9.0.80 ... done
Attaching to www-10.9.0.80
```



```
[10/27/24]seed@VM:~/.../Labsetup 03$ docker ps
CONTAINER ID   IMAGE                COMMAND                  CREATE
D        STATUS          PORTS     NAMES
26ae2f242393   seed-image-www-pki   "/bin/sh -c 'tail -f…"   7 days
 ago    Up 41 seconds             www-10.9.0.80
[10/27/24]seed@VM:~/.../Labsetup 03$ docker exec -it 26ae2f242393 /
bin/bash
root@26ae2f242393:/# ls
bin    dev   lib    libx32  opt   run   sys  var
boot   etc   lib32  media   proc  sbin  tmp  volumes
certs  home  lib64  mnt     root  srv   usr
root@26ae2f242393:/# nano /etc/apache2/sites-available
root@26ae2f242393:/# nano /var/www/bank32/index.html


Use "fg" to return to nano.

[1]+  Stopped                 nano /var/www/bank32/index.html
root@26ae2f242393:/# nano /var/www/bank32/index.html
root@26ae2f242393:/# a2enmod ssl
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
```

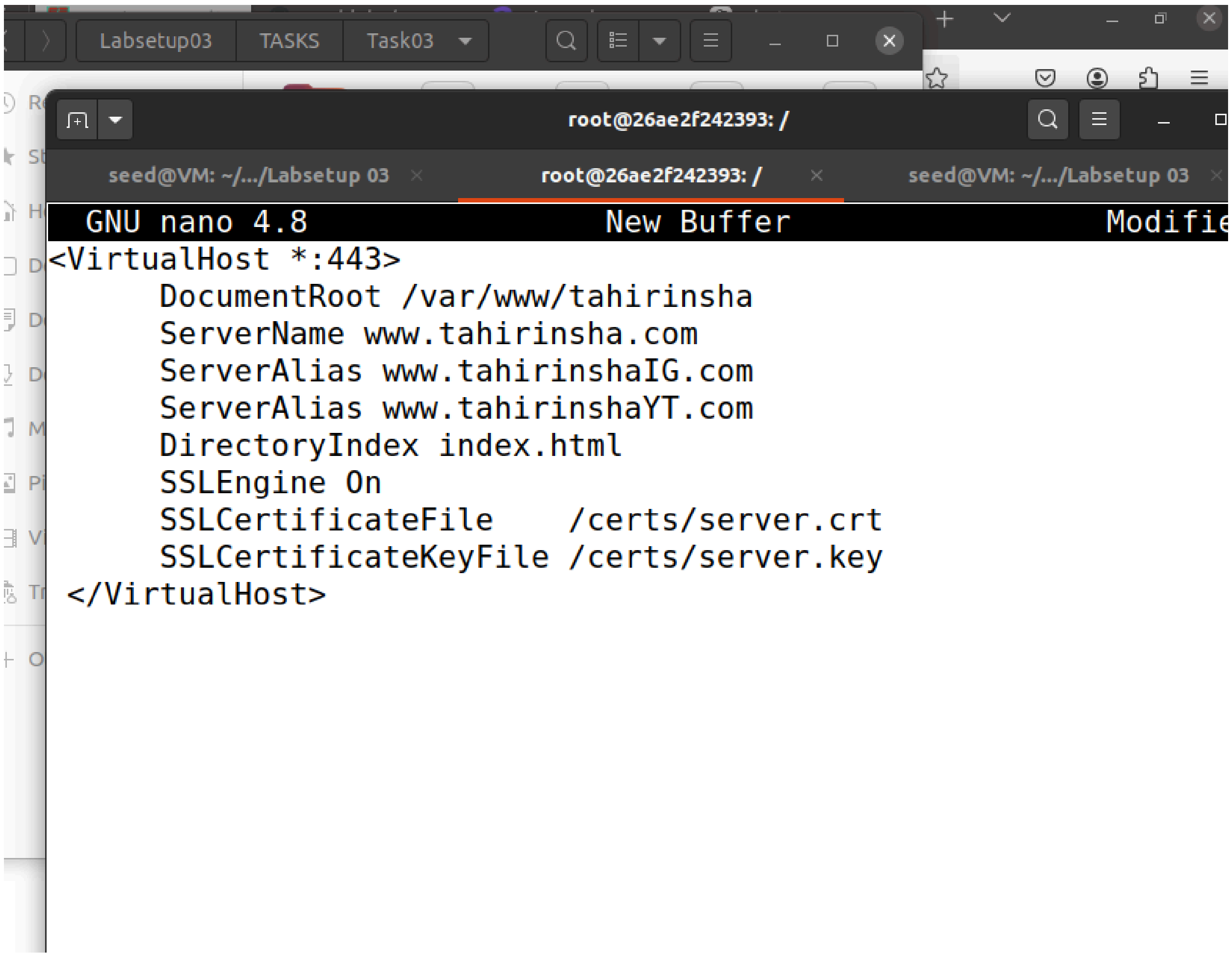


```
Copy files/folders between a container and the local filesystem
[10/27/24]seed@VM:~/.../Labsetup 03$ docker cp ~/Downloads/Labsetup
03/TASKS/Task03/server.crt 26ae2f242393:/certs/server.crt
[10/27/24]seed@VM:~/.../Labsetup 03$ docker cp ~/Downloads/Labsetup
03/TASKS/Task03/server.crt 26ae2f242393:/certs/server.key
[10/27/24]seed@VM:~/.../Labsetup 03$ 
```



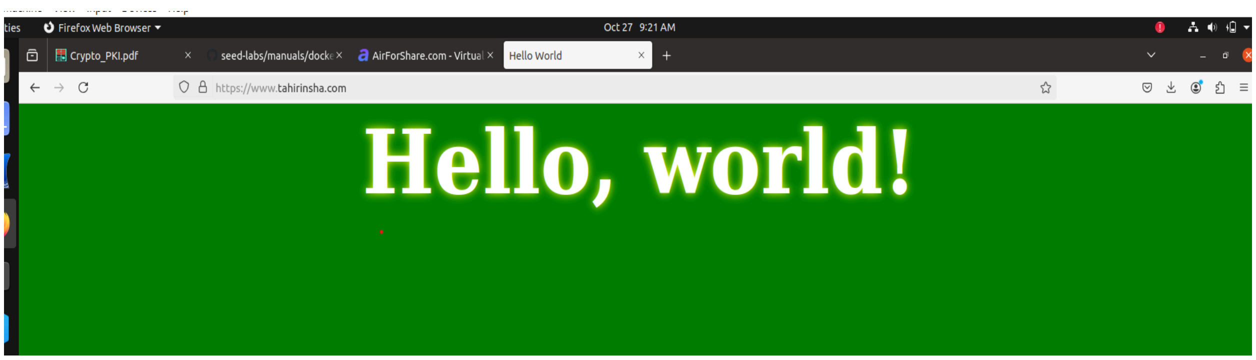
```
root@26ae2f242393:/# nano /etc/apache2/sites-available
root@26ae2f242393:/# nano /var/www/bank32/index.html


Use "fg" to return to nano.

[1]+  Stopped                 nano /var/www/bank32/index.html
root@26ae2f242393:/# nano /var/www/bank32/index.html
root@26ae2f242393:/# a2enmod ssl
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Module socache_shmcb already enabled
Module ssl already enabled
root@26ae2f242393:/# service apache2 restart
 * Restarting Apache httpd web server apache2
Enter passphrase for SSL/TLS keys for www.bank32.com:443 (RSA):
                                                             [ OK ]
root@26ae2f242393:/# [10/27/24]seed@VM:~/.../Labsetup 03$ 
```

tahirinsha_apache_ssl.config



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

Accessing the tahirinsha.com on browser



Observation:

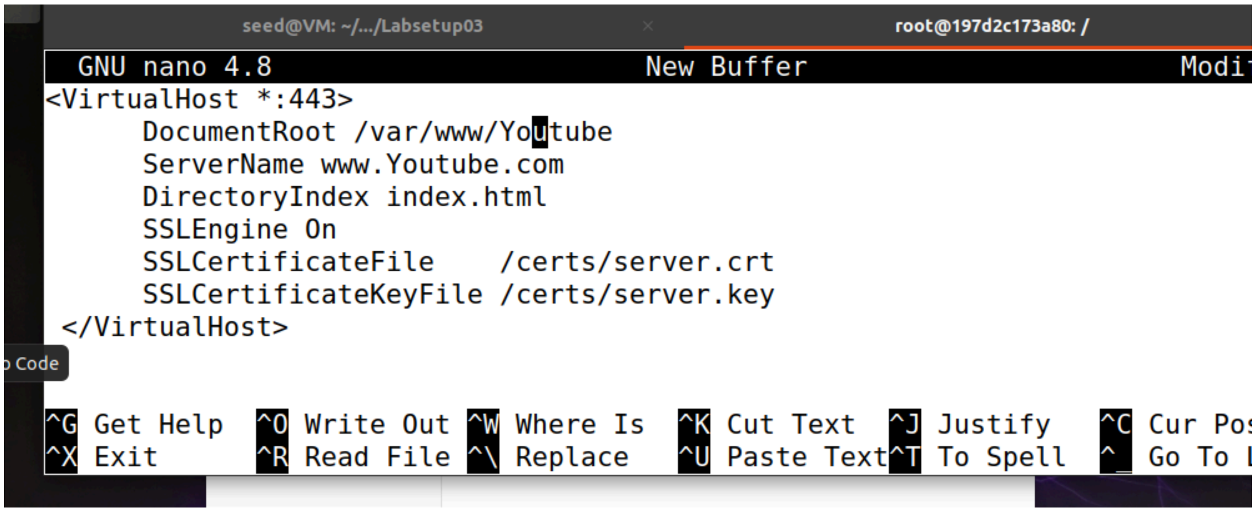
1. Configured an HTTPS website on Apache by creating a `VirtualHost` entry for `www.tahirinsha.com` in `/etc/apache2/sites-available`, setting `ServerName`, `ServerAlias`, and specifying the `DocumentRoot` for website files.

2. Enabled the SSL module with `a2enmod ssl`, added the `tahirinsha_apache_ssl` configuration, and started the Apache server, entering the password ("dees") to unlock the private key.

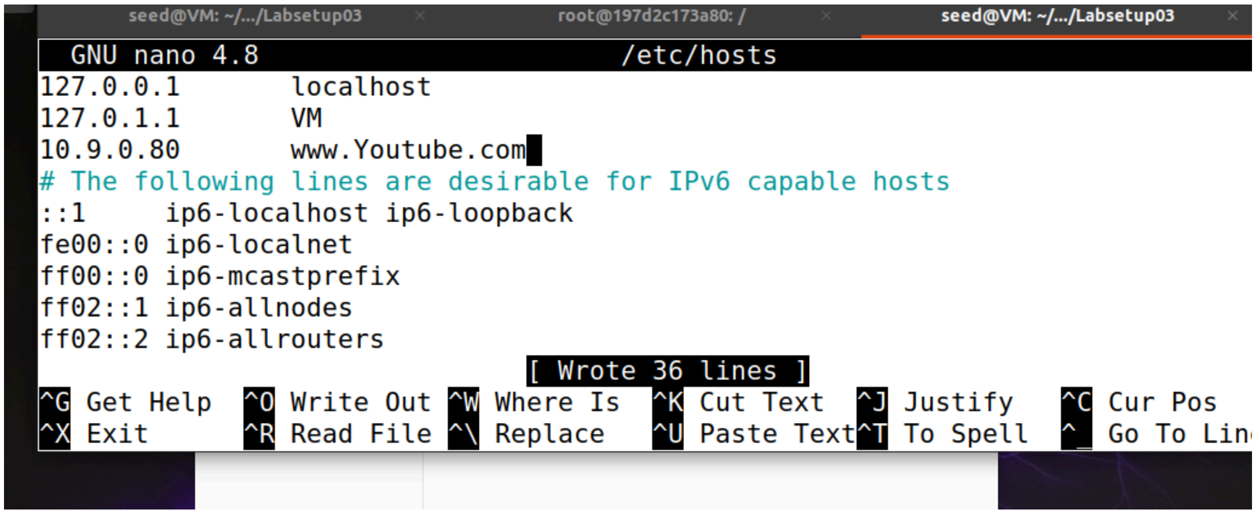
3. Accessed `https://www.tahirinsha.com` in a browser; initially, a certificate error occurred. Imported the custom certificate into Firefox's "Authorities" tab and set it to "Trust this CA to identify websites."

4. After importing, re-accessed the site, which now loaded successfully without certificate errors, verifying encrypted and trusted communication.

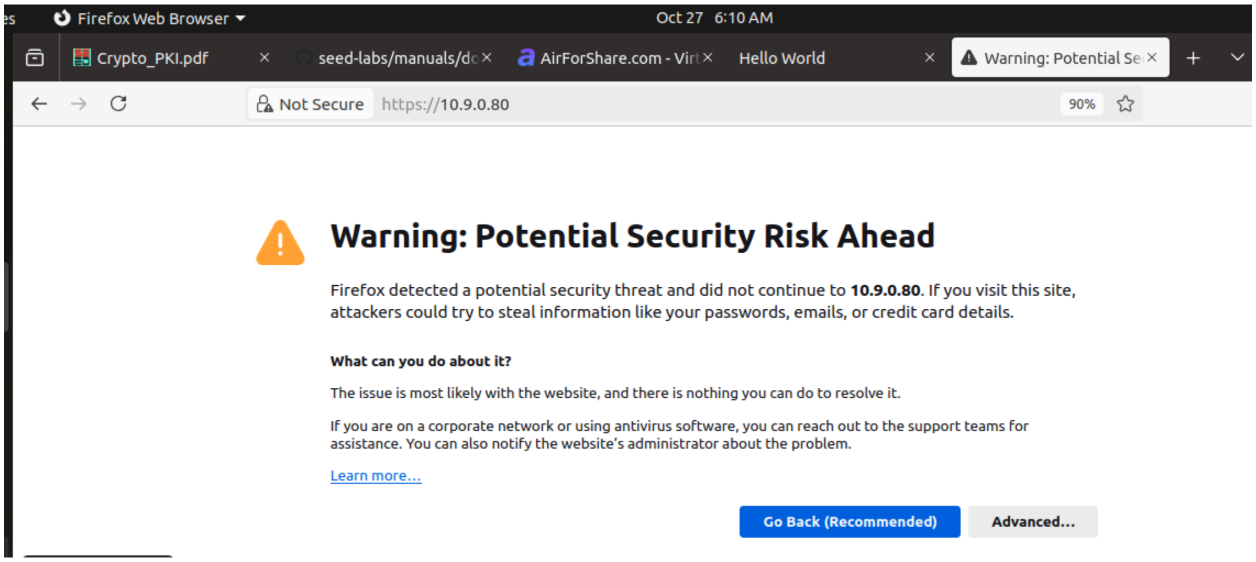
Task05:



```
seed@VM: ~/.../Labsetup03                    root@197d2c173a80: /
 GNU nano 4.8                     New Buffer                    Modi
<VirtualHost *:443>
      DocumentRoot /var/www/Youtube
      ServerName www.Youtube.com
      DirectoryIndex index.html
      SSLEngine On
      SSLCertificateFile    /certs/server.crt
      SSLCertificateKeyFile /certs/server.key
 </VirtualHost>

^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Paste Text^T To Spell  ^_ Go To l
```



```
seed@VM: ~/.../Labsetup03     root@197d2c173a80: /     seed@VM: ~/.../Labsetup03
 GNU nano 4.8                      /etc/hosts
127.0.0.1       localhost
127.0.1.1       VM
10.9.0.80       www.Youtube.com
# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
                               [ Wrote 36 lines ]
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Paste Text^T To Spell  ^_ Go To Lin
```



Warning: Potential Security Risk Ahead

Firefox detected a potential security threat and did not continue to **10.9.0.80**. If you visit this site, attackers could try to steal information like your passwords, emails, or credit card details.

What can you do about it?

The issue is most likely with the website, and there is nothing you can do to resolve it.

If you are on a corporate network or using antivirus software, you can reach out to the support teams for assistance. You can also notify the website's administrator about the problem.

Learn more…

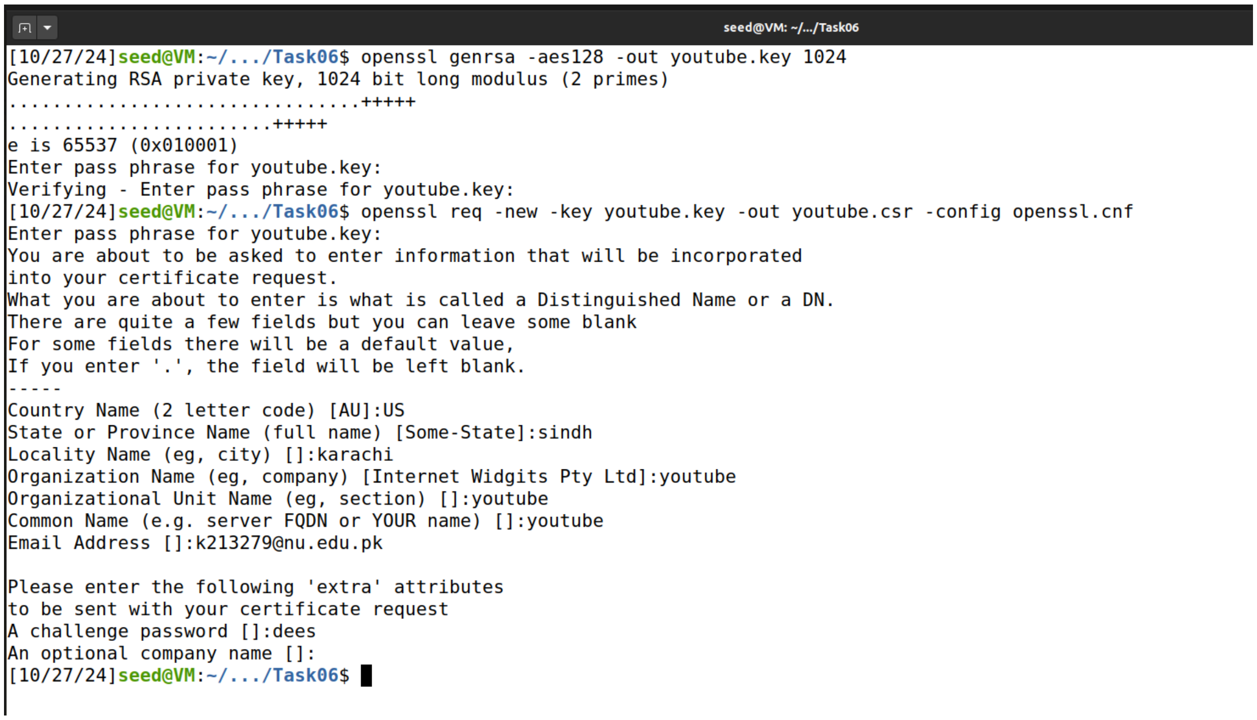
Observation:

We simulated a Man-In-The-Middle (MITM) attack by impersonating www.youtube.com, with a malicious website. After modifying the victim's /etc/hosts file to redirect requests to our server, the user attempting to access www.tahirinsha.com is shown a fake site. If the user cannot distinguish this from the legitimate site, they may unknowingly enter their credentials, exposing them to the attacker.

Additionally, when trying to access www.youtube.com, an error occurs due to an invalid security certificate. This is expected since the presented certificate is for www.tahirinsha.com, not www.youtube.com, resulting in a certificate mismatch that the browser flags as an error. Thus MITM attack failed due to browser efforts.

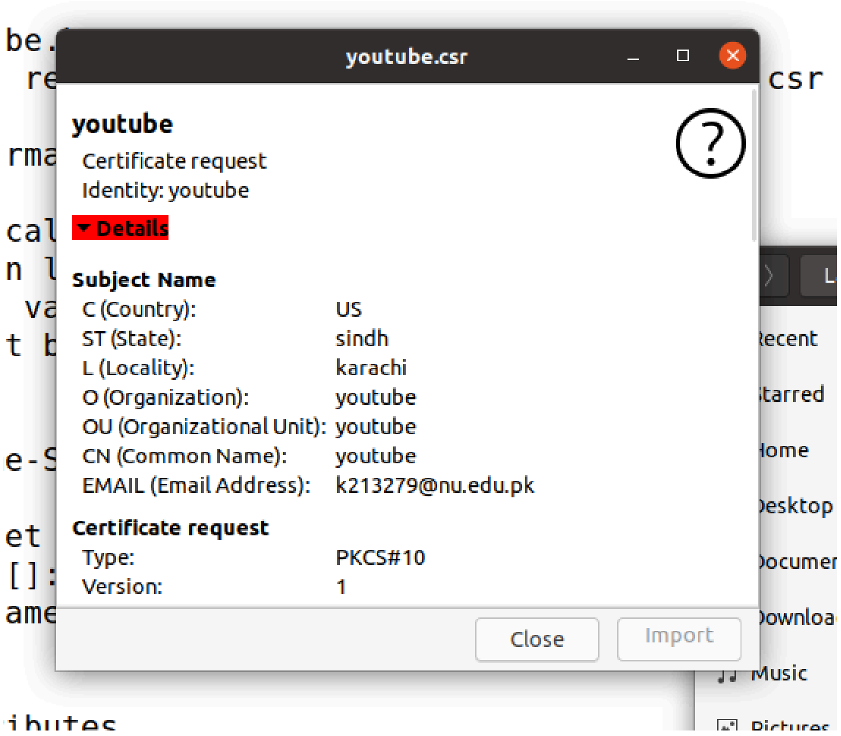
Task06:

Generating the Key for the MITM attacker and then using that key will generate CSR



seed@VM: ~/.../Task06

```
[10/27/24]seed@VM:~/.../Task06$ openssl genrsa -aes128 -out youtube.key 1024
Generating RSA private key, 1024 bit long modulus (2 primes)
................................+++++
........................+++++
e is 65537 (0x010001)
Enter pass phrase for youtube.key:
Verifying - Enter pass phrase for youtube.key:
[10/27/24]seed@VM:~/.../Task06$ openssl req -new -key youtube.key -out youtube.csr -config openssl.cnf
Enter pass phrase for youtube.key:
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]:US
State or Province Name (full name) [Some-State]:sindh
Locality Name (eg, city) []:karachi
Organization Name (eg, company) [Internet Widgits Pty Ltd]:youtube
Organizational Unit Name (eg, section) []:youtube
Common Name (e.g. server FQDN or YOUR name) []:youtube
Email Address []:k213279@nu.edu.pk

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:dees
An optional company name []:
[10/27/24]seed@VM:~/.../Task06$
```



youtube.csr

youtube

Certificate request

Identity: youtube

Details

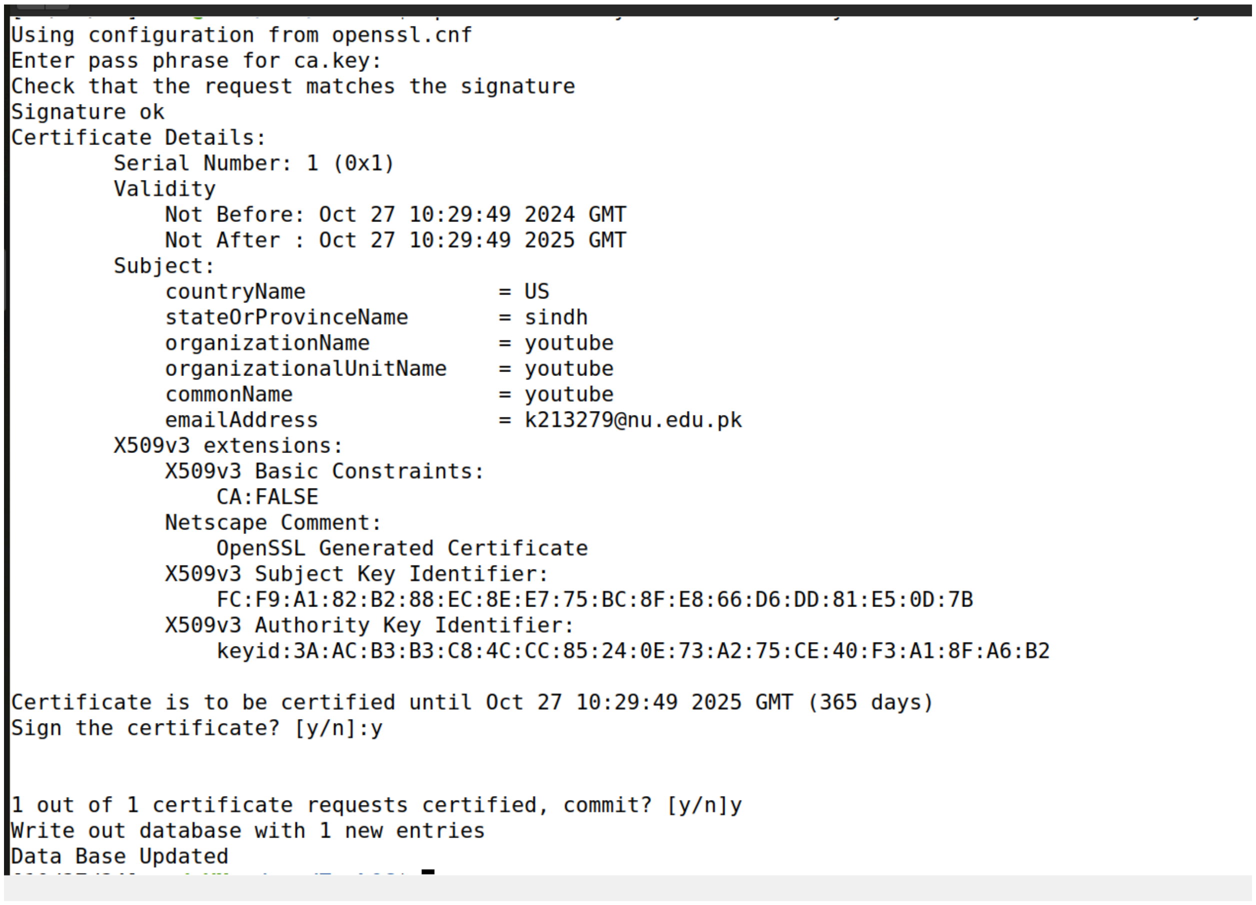
Subject Name

C (Country):	US
ST (State):	sindh
L (Locality):	karachi
O (Organization):	youtube
OU (Organizational Unit):	youtube
CN (Common Name):	youtube
EMAIL (Email Address):	k213279@nu.edu.pk

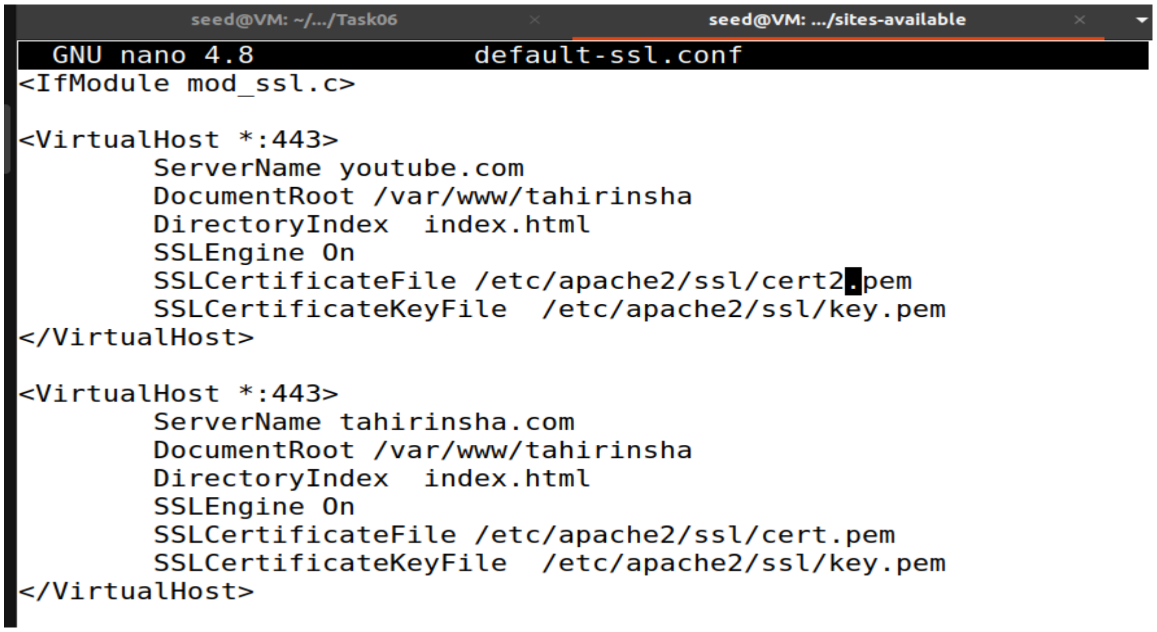
Certificate request

Type:	PKCS#10
Version:	1

By using leaked CA.crt and CA.key, MITM has signed its CSR



```
Using configuration from openssl.cnf
Enter pass phrase for ca.key:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 1 (0x1)
        Validity
            Not Before: Oct 27 10:29:49 2024 GMT
            Not After : Oct 27 10:29:49 2025 GMT
        Subject:
            countryName               = US
            stateOrProvinceName       = sindh
            organizationName          = youtube
            organizationalUnitName    = youtube
            commonName                = youtube
            emailAddress              = k213279@nu.edu.pk
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                FC:F9:A1:82:B2:88:EC:8E:E7:75:BC:8F:E8:66:D6:DD:81:E5:0D:7B
            X509v3 Authority Key Identifier:
                keyid:3A:AC:B3:B3:C8:4C:CC:85:24:0E:73:A2:75:CE:40:F3:A1:8F:A6:B2

Certificate is to be certified until Oct 27 10:29:49 2025 GMT (365 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```



seed@VM: ~/.../Task06 | seed@VM: .../sites-available

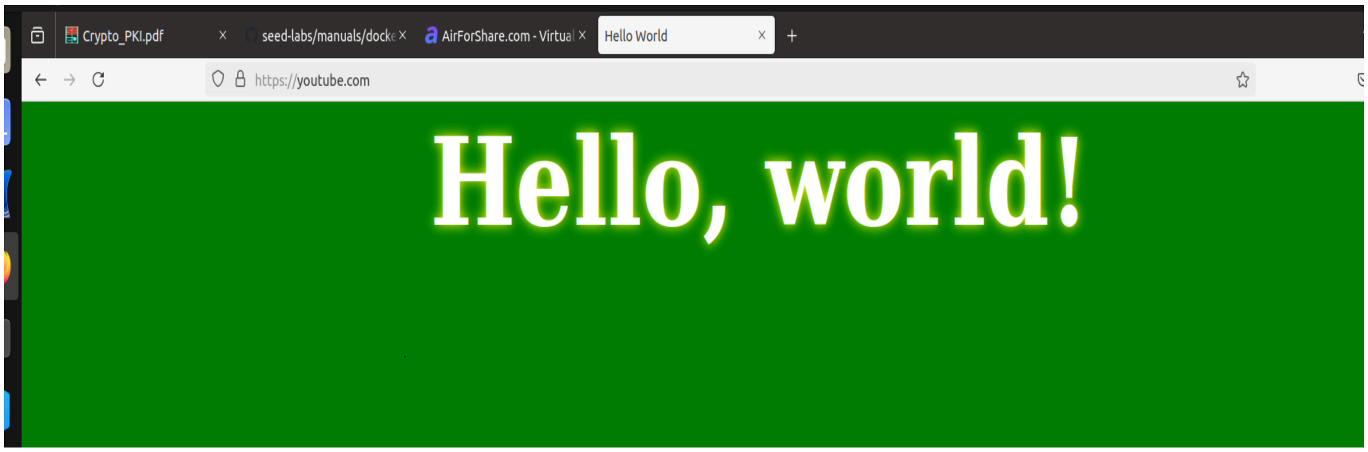
```
  GNU nano 4.8              default-ssl.conf
<IfModule mod_ssl.c>

<VirtualHost *:443>
        ServerName youtube.com
        DocumentRoot /var/www/tahirinsha
        DirectoryIndex  index.html
        SSLEngine On
        SSLCertificateFile /etc/apache2/ssl/cert2.pem
        SSLCertificateKeyFile  /etc/apache2/ssl/key.pem
</VirtualHost>

<VirtualHost *:443>
        ServerName tahirinsha.com
        DocumentRoot /var/www/tahirinsha
        DirectoryIndex  index.html
        SSLEngine On
        SSLCertificateFile /etc/apache2/ssl/cert.pem
        SSLCertificateKeyFile  /etc/apache2/ssl/key.pem
</VirtualHost>
```

Observation:

When accessing `youtube.com` in the browser, there are no certificate errors displayed.

In this scenario, a man-in-the-middle (MitM) attacker has successfully obtained a valid certificate for youtube.com from a compromised Certificate Authority (CA). By exploiting the CA's leaked private key, the attacker can create a certificate that appears authentic. Because the certificate is technically valid and signed by what the browser recognizes as a trusted CA, it bypasses the usual security checks without raising any warnings or alerts to the user. Consequently, the browser treats the connection as secure, establishing it without flagging any issues, and allowing the attacker to intercept and potentially manipulate data unnoticed.

This highlights the risk that arises when a CA's private key is compromised, as it enables attackers to masquerade as legitimate sites to deceive users.


Crypto_PKI.pdf
seed-labs/manuals/docke
AirForShare.com - Virtual
Hello World
https://youtube.com
Hello, world!