Cryptograhphy Hands-On submission 5 | PKI

Details :

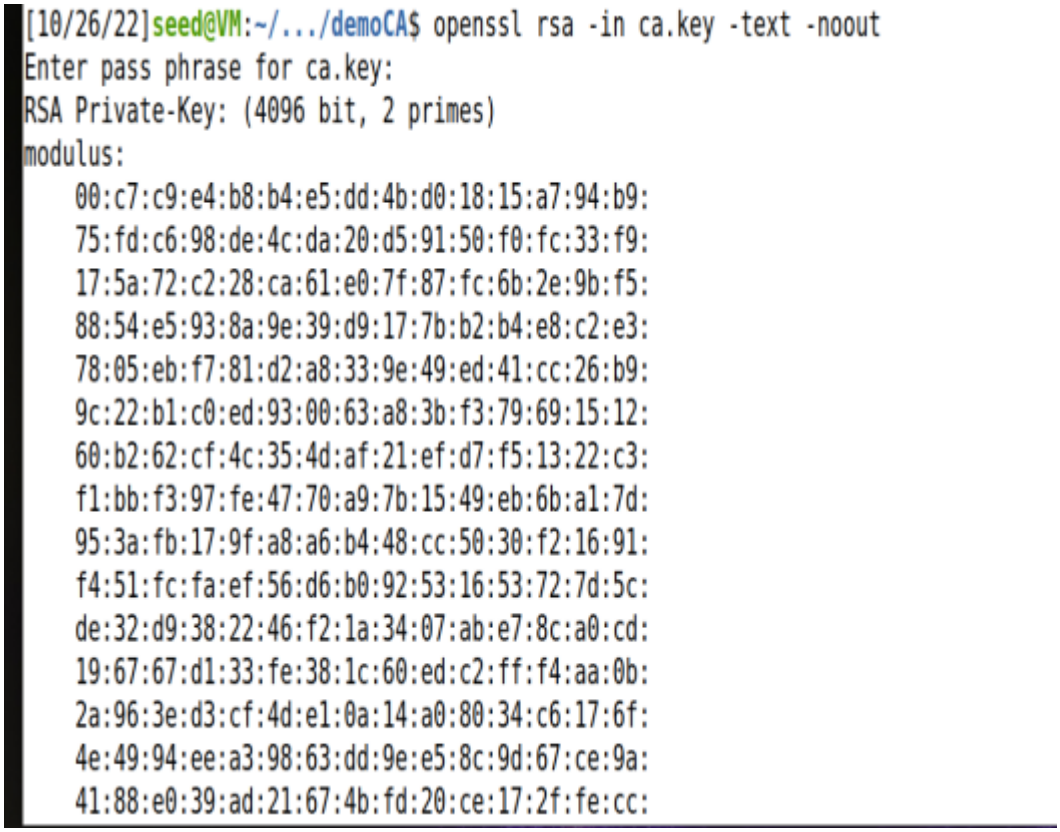
- SRN : PES2UG20CS237
- Name : P K Navin Shrinivas
- Section : D

TASK 1 : Becoming a CA

Screenshots :

```
[10/26/22]seed@VM:~/PKILAB$ cp /usr/lib/ssl/openssl.cnf .
[10/26/22]seed@VM:~/PKILAB$ mkdir demoCA
[10/26/22]seed@VM:~/PKILAB$ cd demoCA/
[10/26/22]seed@VM:~/.../demoCA$ mkdir certs
[10/26/22]seed@VM:~/.../demoCA$ mkdir crl
[10/26/22]seed@VM:~/.../demoCA$ mkdir newcerts
[10/26/22]seed@VM:~/.../demoCA$ touch index.txt
[10/26/22]seed@VM:~/.../demoCA$ touch Serial
[10/26/22]seed@VM:~/.../demoCA$ vim Serial
[10/26/22]seed@VM:~/.../demoCA$ openssl req -x509 -newkey rsa:4096 -sha256 -days 3650 \
> -keyout ca.key -out ca.crt \
> -subj "/CN=www.modelCA.com/O=Model CA LTD./C=US" \
> -passout pass:dees
Generating a RSA private key
.....................................................................++++
.....................++++
writing new private key to 'ca.key'
-----
[10/26/22]seed@VM:~/.../demoCA$ ls
ca.crt  ca.key  certs  crl  index.txt  newcerts  Serial
```

```
[10/26/22]seed@VM:~/.../demoCA$ openssl x509 -in ca.crt -text -noout
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            43:f7:b7:b9:ae:00:1f:45:8c:45:81:38:9d:4d:7b:07:0e:85:52:70
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: CN = www.modelCA.com, O = Model CA LTD., C = US
        Validity
            Not Before: Oct 26 06:06:50 2022 GMT
            Not After : Oct 23 06:06:50 2032 GMT
        Subject: CN = www.modelCA.com, O = Model CA LTD., C = US
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (4096 bit)
                Modulus:
                    00:c7:c9:e4:b8:b4:e5:dd:4b:d0:18:15:a7:94:b9:
                    75:fd:c6:98:de:4c:da:20:d5:91:50:f0:fc:33:f9:
                    17:5a:72:c2:28:ca:61:e0:7f:87:fc:6b:2e:9b:f5:
```



Observation :

- Above using openssl, we generate a certificate and a key. On seeing certificate details, we see its a SHA256RSA encrypted certificate.
- This certificate also hold the modulus and public key parts of this system.
- To see details of private key, one needs the password with which this system was created with. This hold all the private and public parts of the key.

TASK 2 : Creating priv/pub for requesting certificate.

Screenshots :

```
[10/26/22]seed@VM:~/.../PKILAB$ cd demoCA/
[10/26/22]seed@VM:~/.../demoCA$ openssl req -newkey rsa:2048 -sha256 \
> -keyout server.key -out server.csr \
> -subj "/CN=www.bank32.com/O=Bank32 Inc./C=US" \
> -passout pass:dees \
> -addext "subjectAltName = DNS:www.bank32.com, \
> DNS:www.bank32A.com, \
> DNS:www.bank32B.com"
Generating a RSA private key
.+++++
.....................................................+++++
writing new private key to 'server.key'
-----
[10/26/22]seed@VM:~/.../demoCA$ ls
ca.crt  ca.key  certs  crl  index.txt  newcerts  Serial  server.csr  server.key
[10/26/22]seed@VM:~/.../demoCA$ openssl req -in server.csr-text -noout
Can't open server.csr-text for reading, No such file or directory
140564274062656:error:02001002:system library:fopen:No such file or directory:crypto/bio/bss_file.c:69:fopen('server.csr-text','
r')
140564274062656:error:2006D080:BIO routines:BIO_new_file:no such file:crypto/bio/bss_file.c:76:
[10/26/22]seed@VM:~/.../demoCA$ openssl req -in server.csr -text -noout
Certificate Request:
    Data:
        Version: 1 (0x0)
        Subject: CN = www.bank32.com, O = Bank32 Inc., C = US
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
```

```
[10/26/22]seed@VM:~/.../demoCA$ openssl rsa -in server.key -text -noout
Enter pass phrase for server.key:
RSA Private-Key: (2048 bit, 2 primes)
modulus:
    00:a8:77:9a:7f:8c:26:eb:35:5a:a9:42:62:2e:f4:
    4a:97:05:56:90:ec:ba:2a:f9:92:40:94:26:9a:0e:
    1c:ad:77:23:7c:f3:fa:fa:50:a9:ec:d7:d7:01:d6:
    5c:27:fe:08:e2:76:39:e2:a3:85:57:18:5c:eb:41:
    ed:75:42:8a:ff:e0:ce:3a:92:7a:5f:e1:bf:80:71:
    b3:93:a0:62:41:7d:f6:fb:06:f4:56:a9:29:99:6e:
    ac:98:02:c4:e0:85:2f:9c:72:9d:41:37:f1:52:56:
    7b:1f:33:38:76:f7:d7:14:2b:9b:56:53:79:80:6c:
    e8:bc:15:b9:34:4d:b8:e8:b4:45:78:54:6a:a9:eb:
    4e:cf:17:be:a8:94:36:31:b3:7e:11:4f:5c:04:b8:
    ec:16:0c:11:8b:58:22:8a:33:6b:79:33:3b:ae:49:
    fb:e2:96:57:cf:09:2b:12:a6:70:c7:85:2f:00:d1:
    af:d3:4a:30:ad:52:7d:1a:94:58:de:16:61:2e:fa:
    6a:c0:fa:cc:e2:7c:82:ff:be:48:35:0b:ee:62:af:
    c1:f2:09:ef:3a:59:cf:5c:dd:85:10:3d:6e:4c:3e:
    67:1a:04:2b:bc:24:42:5f:72:7d:b5:e6:be:42:3f:
    8b:96:e8:d9:37:c4:1a:ba:ca:3a:5e:05:92:ed:20:
```

Observation :

- Above we create a csr, that is a certificate request with things that we want the server to enc their private material.
- We also have the priv/pub stuff stored in the .key file.

TASK 3 : Creating cerificate for our server [From which we will be requesting using previous csr]

Screenshots :

```
[10/26/22]seed@VM:~/PKILAB$ openssl ca -config openssl.cnf -policy policy_anything -md sha256 -days 3650 -in server.csr -out ser
ver.crt -batch -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: 4369 (0x1111)
        Validity
            Not Before: Oct 26 07:27:29 2022 GMT
            Not After : Oct 23 07:27:29 2032 GMT
        Subject:
            countryName               = US
            organizationName          = Bank32 Inc.
            commonName                = www.bank32.com
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                03:29:BB:05:C7:28:1D:4D:A4:1C:A4:B8:5E:2B:05:F9:7E:4B:30:EA
            X509v3 Authority Key Identifier:
```

Observation :

- Above we are creating a certificate acting as www.Bank32.com using the csr that we generated in the previous step, along with serial.

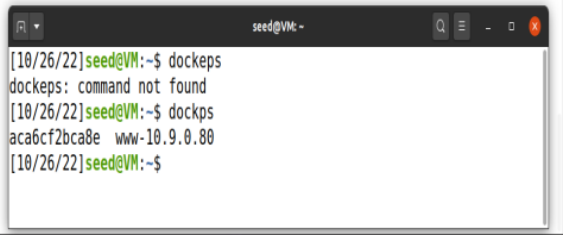
TASK 4 : Setting up server and setting up for certificate exchange

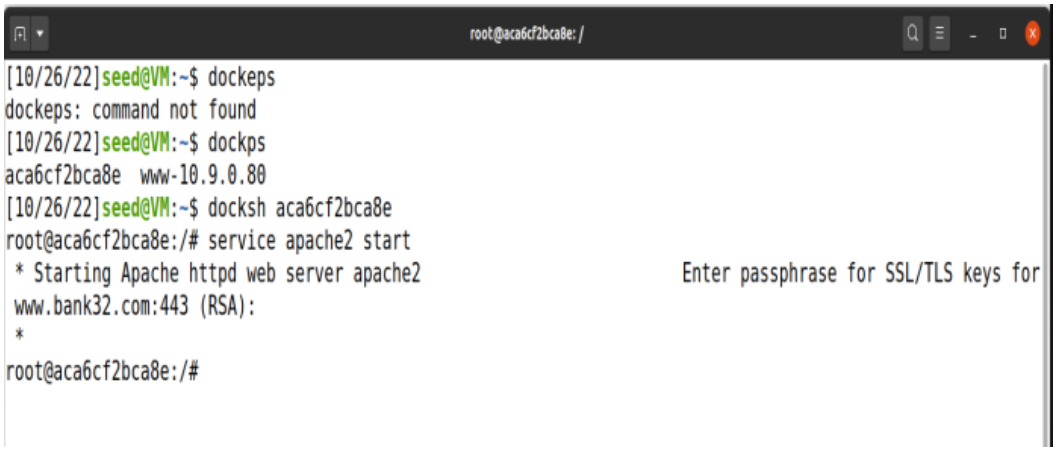
Screenshots and Observations :

```
[10/26/22]seed@VM:~/PKILAB$ cp *.crt ./Labsetup/image_www/certs/
[10/26/22]seed@VM:~/PKILAB$ cp server.key ./Labsetup/image_www/certs/
[10/26/22]seed@VM:~/PKILAB$ cd Labsetup/
[10/26/22]seed@VM:~/.../Labsetup$ ls
docker-compose.yml  image_www  volumes
[10/26/22]seed@VM:~/.../Labsetup$ cd image_www/
[10/26/22]seed@VM:~/.../image_www$ cd certs/
[10/26/22]seed@VM:~/.../certs$ ls
bank32.crt  bank32.key  ca.crt  modelCA.crt  README.txt  server.crt  server.key
[10/26/22]seed@VM:~/.../certs$ rm -r bank32.*
[10/26/22]seed@VM:~/.../certs$ ls
ca.crt  modelCA.crt  README.txt  server.crt  server.key
[10/26/22]seed@VM:~/.../certs$ rm -r modelCA.crt
[10/26/22]seed@VM:~/.../certs$ cat README.txt

bank32.crt: Bank32's public-key certificate, signed by ModelCA.

bank32.key: Bank32's private key
            The password used for protecting the private key is "dees".

modelCA.crt: ModelCA's public-key certificate
[10/26/22]seed@VM:~/.../certs$ rm -r README.txt
[10/26/22]seed@VM:~/.../certs$ ls
ca.crt  server.crt  server.key
[10/26/22]seed@VM:~/.../certs$ mv server.
server.crt  server.key
[10/26/22]seed@VM:~/.../certs$ mv server.crt bank32.crt
[10/26/22]seed@VM:~/.../certs$ mv server.key bank32.key
[10/26/22]seed@VM:~/.../certs$ mv ca.crt modelCA.crt
```

- Above we copy over the certicated for Bank32.com and CA certificates over to the docker file in the apace installtion.

```
[10/26/22]seed@VM:~/.../Labsetup$ docker-compose build
Building web-server
Step 1/7 : FROM handsonsecurity/seed-server:apache-php
 ---> 2365d0ed3ad9
Step 2/7 : ARG WWWDIR=/var/www/bank32
 ---> Using cache
 ---> bd5e2bdd6edb
Step 3/7 : COPY ./index.html ./index_red.html $WWWDIR/
 ---> Using cache
 ---> ff05e9db35bf
Step 4/7 : COPY ./bank32_apache_ssl.conf /etc/apache2/sites-available
 ---> Using cache
 ---> c046c55826f8
Step 5/7 : COPY ./certs/bank32.crt ./certs/bank32.key  /certs/
 ---> 26605e0b3ea2
Step 6/7 : RUN  chmod 400 /certs/bank32.key      && chmod 644 $WWWDIR/index.html      && chmod 644 $WWWDIR/index_red.html      && a2ensit
e bank32_apache_ssl
 ---> Running in 63b9b6bc9949
Enabling site bank32_apache_ssl.
To activate the new configuration, you need to run:
  service apache2 reload
Removing intermediate container 63b9b6bc9949
 ---> 618b02d859ba
Step 7/7 : CMD  tail -f /dev/null
 ---> Running in e44d590f1f29
Removing intermediate container e44d590f1f29
 ---> 6fb84bdd7f22

Successfully built 6fb84bdd7f22
Successfully tagged seed-image-www-pki:latest
[10/26/22]seed@VM:~/.../Labsetup$ docker-compose up
Recreating www-10.9.0.80 ... done
Attaching to www-10.9.0.80
```

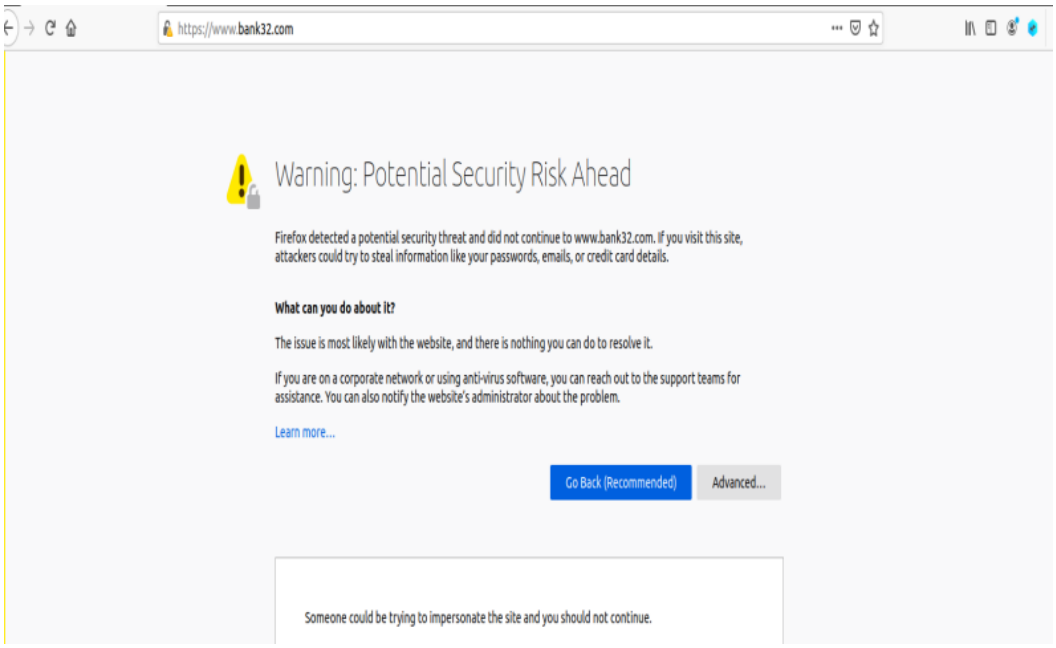




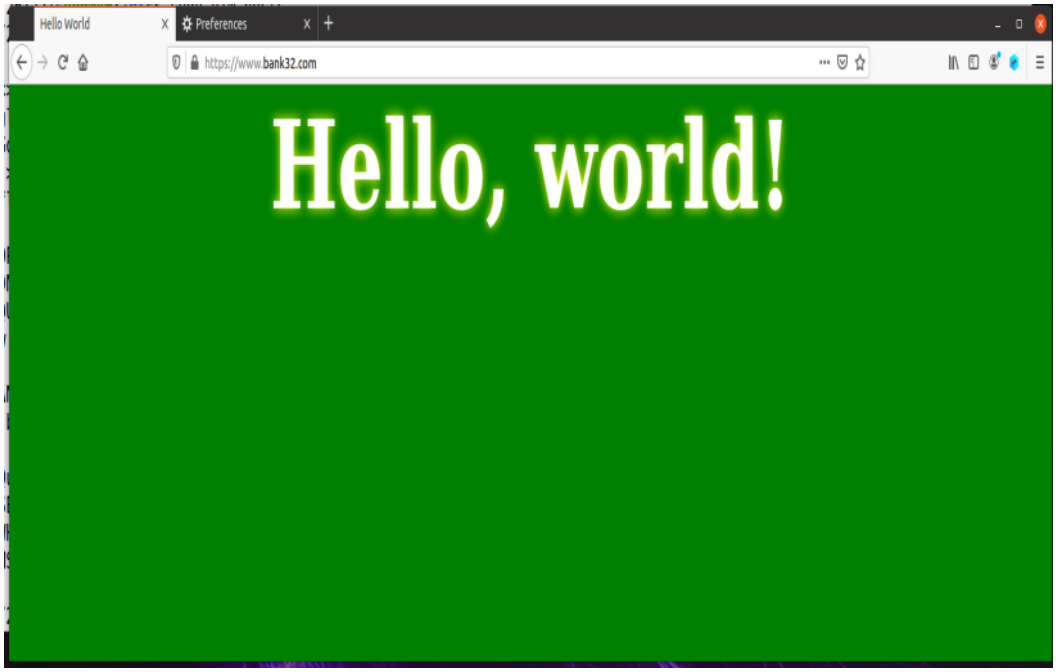
- Above we simply build the docker image and start it.

```
# For XSS Lab
10.9.0.5        www.xsslabelgg.com
10.9.0.5        www.example32a.com
10.9.0.5        www.example32b.com
10.9.0.5        www.example32c.com
10.9.0.5        www.example60.com
10.9.0.5        www.example70.com

# For CSRF Lab
10.9.0.5        www.csrflabelgg.com
10.9.0.5        www.csrflab-defense.com
10.9.0.105      www.csrflab-attacker.com

# For Shellshock Lab
10.9.0.80       www.seedlab-shellshock.com

# Bank32 for PKI lab
10.9.0.80 www.bank32.com
"hosts" 34L, 825C
```

- Above we are modifying the hosts file for bank32.com such that when queried from dns, we will get a local ip address and resolved address.

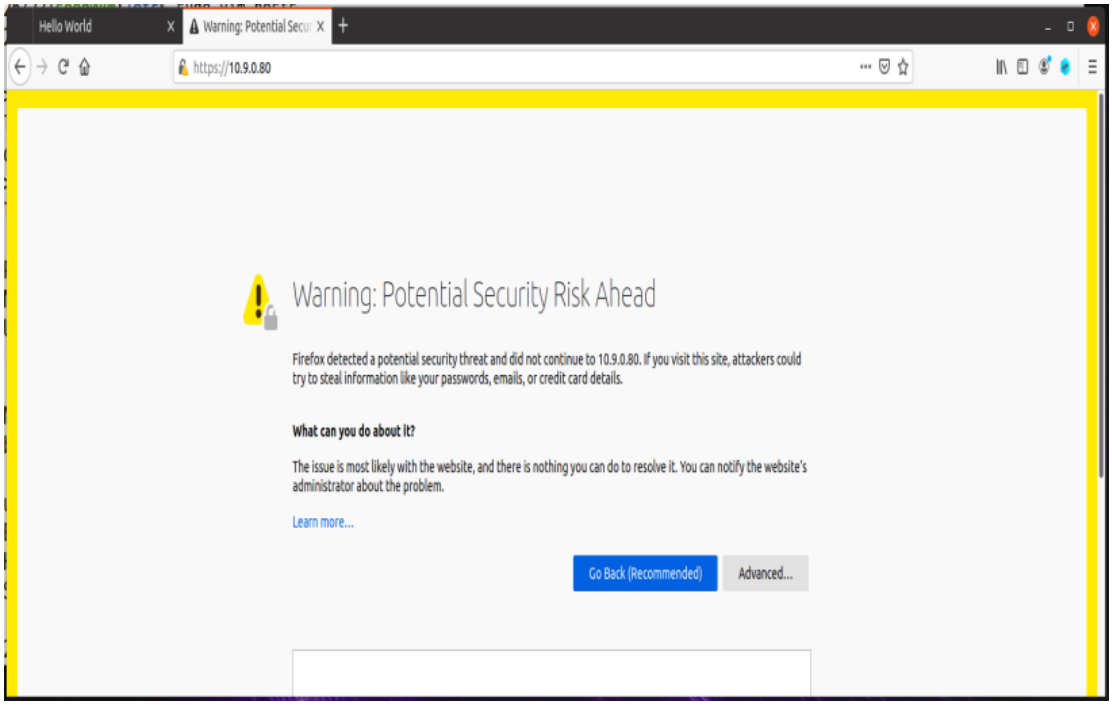


- Above we see that firefox gives a warning that the CA is not a trusted one and may be self signed.



- Above we see no such warning as the modelCA.crt certificate was added to the trusted list of certs in firefox.

Question :



- Above firefox shows an error code of bad cert domain, why?
- This is bacuse if we see the command that we did to create bank32.com's certificate we used 3 domains, one main domain and 2 alias domains. None of them were the

ip address of the server. Firefox verifies if the cerficated are fit by comparing hosts, which in this case is not.

- The three domain this certificate is fit for is : "www.bank32.com","www.bank32A.com","www.bank32B.com"

TASK 5 : MITM attack

Screenshots and Observations :

```
[10/26/22]seed@VM:~/PKILAB$ openssl req -newkey rsa:2048 -sha256 -keyout server.key -out server.csr -subj "/CN=www.example.com/O
=example Inc./C=US" -passout pass:dees
Generating a RSA private key
.....+++++
.....+++++
writing new private key to 'server.key'
-----
[10/26/22]seed@VM:~/PKILAB$ openssl ca -config openssl.cnf -policy policy_anything \
> -md sha256 -days 3650 \
> -in server.csr -out server.crt -batch \
> ^Cert ca.crt -keyfile ca.key
[10/26/22]seed@VM:~/PKILAB$ openssl ca -config openssl.cnf -policy policy_anything -md sha256 -days 3650 -in server.csr -out ser
ver.crt -batch -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: 4370 (0x1112)
        Validity
```

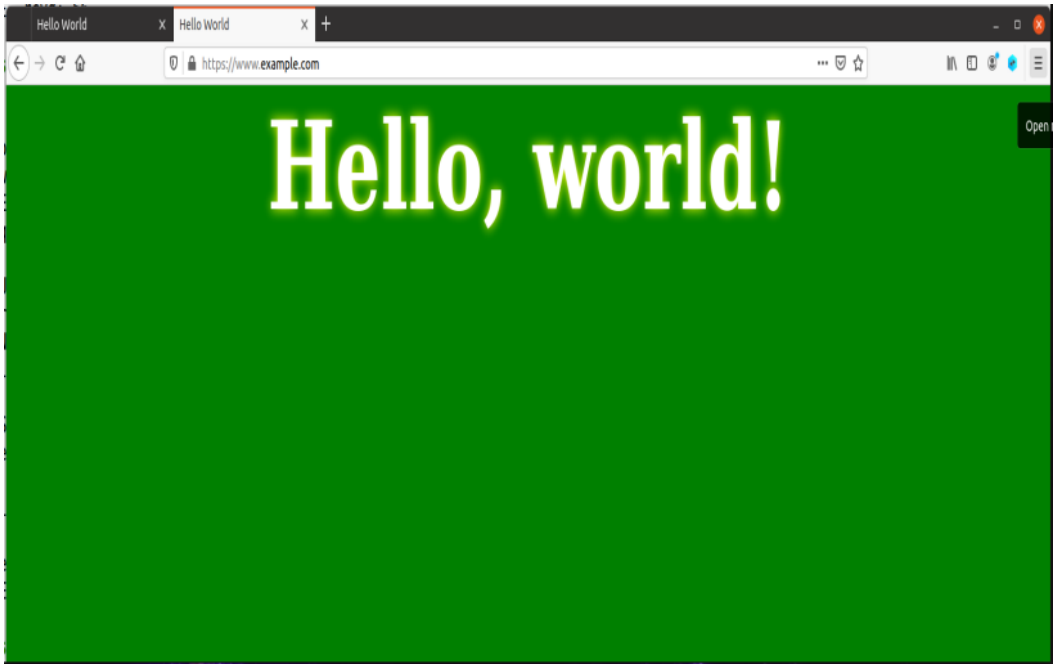
- Above we are acting as a CA and creating certificates for www.example.com using csr and crt

```
[10/26/22]seed@VM:~/PKILAB$ mv example.* ./Labsetup/image_www/certs/
[10/26/22]seed@VM:~/PKILAB$ dockps
ca2751d650cb  www-10.9.0.80
[10/26/22]seed@VM:~/PKILAB$ docksh ca2751d650cb
root@ca2751d650cb:/# service apache2 start
 * Starting Apache httpd web server apache2
Enter passphrase for SSL/TLS keys for www.example.com:443 (RSA):
 *
root@ca2751d650cb:/#
```

- Above we rebuild the docker image with new ssl conf and certificates, one that can serve www.example.com

```
# For PKI lab
10.9.0.80 www.example.com
~
"hosts" 34L, 819C written
```

- Above we are creating fake entries in the hosts dns record file. Usually for MITM attack, this change has to be done programmatically and this rightfully need elevated permissions.



- Above we visit www.example.com in the host machine, as we have added ourselves as trusted CA in the firefox list, we see no warnings come up. The unaware hosts never knows that is isn't the original site. Hence our MITM attack has been executed successfully.