

# Public Key Infrastructure (PKI)

## Lab 8 (Task 5 and 6)

Himanshu Sharma - 202117006

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

#### Step 1: Setting up the malicious website

We are trying to set up a new malicious website. Here 'hello.com' is used to setup on Apache web server. We modified default-ssl.cnf and 000-default.cnf file to add hello.com website to our Apache web server.

000-default.cnf

```
<VirtualHost *:80>
    ServerName hello.com
    DocumentRoot /var/www/hello
    DirectoryIndex index.html
</VirtualHost>
```

default-ssl.cnf

```
<VirtualHost *:443>
    ServerName hello.com
    DocumentRoot /var/www/Example_Two
    DirectoryIndex index.html
    SSLEngine On
    SSLCertificateFile /var/www/Example_Two/server.crt
    SSLCertificateKeyFile /var/www/Example_Two/server.pem
</VirtualHost>
```

#### Step 2: Becoming the man in the middle

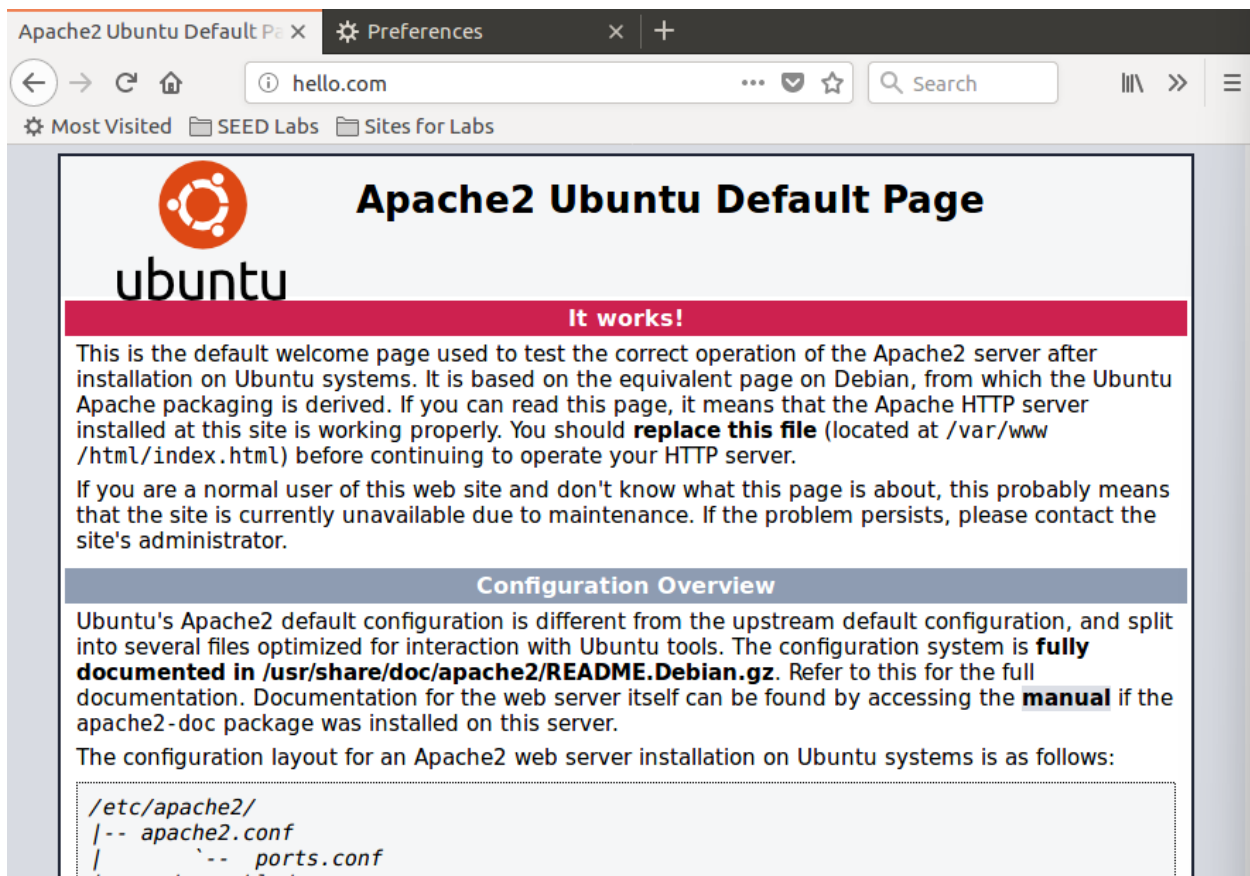
We modify the /etc/hosts file so that the traffic host is redirect to our website causing DNS cache poisoning attack.

```
127.0.0.1    www.seedlabclickjacking.com
127.0.0.1    SEEDPKILab2020.com
127.0.0.1    hello.com
```

#### Step 3: Browse the target website

Now everything is setup. We try to browse the target website i.e., 'hello.com'. The host computer redirects our request to access hello.com using local DNS and redirects it to our Apache server.

Everything seems fine.



Continued ...

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

It is already mentioned that the private key of the root CA has been compromised. So, using that private key we can generate certificate for our Target website. The steps will be similar to which we performed in Lab6 (Task 1-4).

Generating certificate signing request for hello.com

```
[04/22/22]seed@VM:~/lab7$ openssl req -new -key server.key -out hello.csr -config openssl.cnf
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]:IN
State or Province Name (full name) [Some-State]:RAJASTHAN
Locality Name (eg, city) []:JAIPUR
Organization Name (eg, company) [Internet Widgits Pty Ltd]:HTT
Organizational Unit Name (eg, section) []:SOFTWARE
Common Name (e.g. server FQDN or YOUR name) []:hello.com
Email Address []:hello@hello.com
```

Producing hello.crt certificate using ca.crt, ca.key and hello.csr

```
[04/22/22]seed@VM:~/lab7$ openssl ca -in hello.csr -out hello.crt -cert ca.crt -keyfile ca.key -config openssl.cnf
Using configuration from openssl.cnf
Enter pass phrase for ca.key:
Check that the request matches the signature
Signature ok
Certificate Details:
    Serial Number: 4098 (0x1002)
    Validity
        Not Before: Apr 22 12:33:41 2022 GMT
        Not After : Apr 22 12:33:41 2023 GMT
    Subject:
        countryName           = IN
        stateOrProvinceName   = RAJASTHAN
        organizationName      = HTT
        organizationalUnitName = Software
        commonName            = hello.com
        emailAddress          = hello@hello.com
    X509v3 extensions:
        X509v3 Basic Constraints:
            CA:FALSE
    Netscape Comment:
```

Combining the key, crt and generate pem. Finally transfer it to DocumentRoot of our web server.

```
[04/22/22]seed@VM:~/lab7$ cp server.key hello.pem
[04/22/22]seed@VM:~/lab7$ cat hello.crt >> hello.pem
```

```
[04/22/22]seed@VM:~/lab7$ sudo cp hello.crt hello.pem /var/www/Example_Two/
```

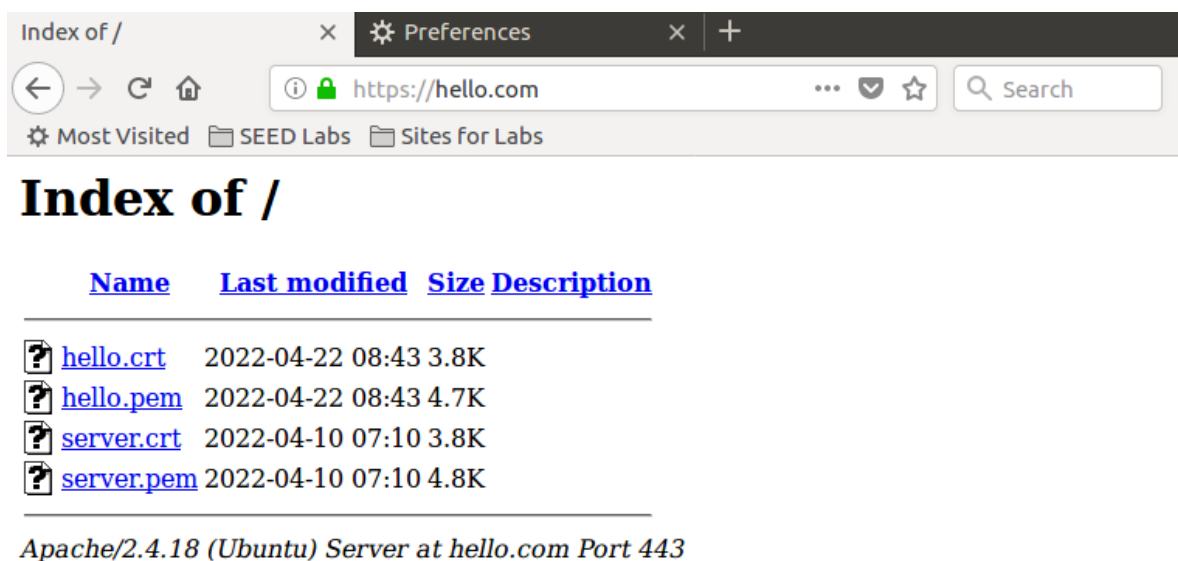
As we want to impersonate as original server, we will use same private key (server.pem) and our hello.crt certificate file.

```
<VirtualHost *:443>
    ServerName hello.com
    DocumentRoot /var/www/Example_Two
    DirectoryIndex index.html
    SSLEngine On
    SSLCertificateFile /var/www/Example_Two/hello.crt
    SSLCertificateKeyFile /var/www/Example_Two/server.pem
</VirtualHost>
```

Testing and restarting our Apache web server.

```
[04/22/22]seed@VM:~/lab7$ sudo apachectl configtest
AH00112: Warning: DocumentRoot [/var/www/seedlabclickjacking] does not exist
[Fri Apr 22 08:44:29.299272 2022] [core:error] [pid 2901] (EAI 2)Name or service not known: AH00547: Could not resolve host name *.80 -- ignoring!
AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1. Set the 'ServerName' directive globally to suppress this message
Syntax OK
[04/22/22]seed@VM:~/lab7$ sudo service apache2 restart
Enter passphrase for SSL/TLS keys for hello.com:443 (RSA): *****
```

Our website is up and running as we intended.



Index of /

Preferences

https://hello.com

Search

Name	Last modified	Size	Description
<a href="#">hello.crt</a>	2022-04-22 08:43	3.8K	
<a href="#">hello.pem</a>	2022-04-22 08:43	4.7K	
<a href="#">server.crt</a>	2022-04-10 07:10	3.8K	
<a href="#">server.pem</a>	2022-04-10 07:10	4.8K	

Apache/2.4.18 (Ubuntu) Server at hello.com Port 443

The screenshot displays a web browser window with the address bar showing `https://hello.com`. The main content area shows an "Index of /" directory listing with the following files:

Name	Last modified	Size	Description
<a href="#">hello.crt</a>	2022-04-22 08:43	3.8K	
<a href="#">hello.pem</a>	2022-04-22 08:43	4.7K	
<a href="#">server.crt</a>	2022-04-10 07:10	3.8K	
<a href="#">server.pem</a>	2022-04-10 07:10	4.8K	

Below the listing, it states "Apache/2.4.18 (Ubuntu) Server at hello.com Port 443".

On the right, the "Certificate Viewer: 'hello.com'" window is open, showing the following details:

- General** tab is selected.
- This certificate has been verified for the following purposes:** SSL Server Certificate.
- Issued To:**
  - Common Name (CN): **hello.com**
  - Organization (O): HTTP
  - Organizational Unit (OU): Software
  - Serial Number: 10:02
- Issued By:**
  - Common Name (CN): HIMANSHU
  - Organization (O): HTTP
  - Organizational Unit (OU): SOFTWARE
- Period of Validity:**
  - Begins On: April 22, 2022
  - Expires On: April 22, 2023
- Fingerprints:**
  - SHA-256 Fingerprint: 72:B3:54:35:80:33:BE:E6:D1:35:BD:22:...
  - SHA1 Fingerprint: F4:C7:2A:09:81:8C:7E:99:...

At the bottom, the "Page Info - https://hello.com/" window is open, showing the "Security" tab. It displays the following information:

- Website Identity:**
  - Website: **hello.com**
  - Owner: **This website does not supply own**
  - Verified by: **HTTP**
  - Expires on: **April 22, 2023**
- Privacy & History:**
  - Have I visited this website prior to today?
  - Is this website storing information (cookies) on my computer?
  - Have I saved any passwords for this website?

Checking the certificate information of hello.com, we can clearly see **hello.com certificate** is verified by the HTTP. Viewing the detailed certificate information, we can verify the certificate we issued by Root-CA (HIMANSHU-HTTP) to hello.com.

Thus, we can easily impersonate as original webserver and perform Man-In-The-Middle attack.