Shaun Young - N01173292
William Milligan – N00617078
24-NOV-2018
CIS-4360

Project 2 – Group 19

Step 1: Generate public/private key pair. The company needs to first create its own public/private key pair. We can run the following command to generate an RSA key pair (both private and public keys). You will also be required to provide a password to encrypt the private key (using the AES-128 encryption algorithm, as is specified in the command option). The keys will be stored in the file server.key:

$ openssl genrsa -aes128 -out server.key 1024

```
[11/24/18]seed@VM:~/demoCA$ openssl genrsa -aes128 -out server.key 1024
Generating RSA private key, 1024 bit long modulus
..++++++
.....................................................................................
.........................++++++
e is 65537 (0x10001)
Enter pass phrase for server.key:
Verifying - Enter pass phrase for server.key:
[11/24/18]seed@VM:~/demoCA$
```

The server.key is an encoded text file (also encrypted), so you will not be able to see the actual content, such as the modulus, private exponents, etc. To see those, you can run the following command:

$ openssl rsa -in server.key -text

```
[11/24/18]seed@VM:~/demoCA$ openssl rsa -in server.key -text
Enter pass phrase for server.key:
Private-Key: (1024 bit)
modulus:
    00:cd:d7:b0:16:f8:3a:f2:2a:43:4d:d4:55:70:d3:
    5b:4c:c2:75:44:b1:9f:db:02:ef:16:60:9c:5a:fb:
    ce:01:a9:f9:3c:d6:53:e8:a2:d4:76:5c:04:e9:23:
    12:03:3e:b7:d6:dd:8b:4d:4a:ae:32:e6:8d:58:4e:
    31:38:f1:21:6b:6e:ce:76:c8:fb:b1:74:9f:20:2a:
    60:7d:04:75:4c:aa:89:e5:d4:8b:81:c9:f9:be:4d:
    91:7d:3e:64:38:55:ea:a4:5a:79:dd:67:60:d8:5d:
    01:96:ce:db:9c:6a:f8:1f:72:58:9d:94:b4:a5:ae:
    35:78:f7:ac:be:3a:e1:23:4f
publicExponent: 65537 (0x10001)
privateExponent:
    7e:3c:c8:c1:4d:7a:d9:09:12:3d:a8:fa:bd:36:e1:
    c7:31:7a:b9:a6:35:63:1b:69:85:4f:ba:94:11:d9:
    45:2b:f8:ae:56:c3:1e:e8:bd:63:d9:0b:29:ef:58:
    1c:a5:5a:19:6b:c5:02:a1:ca:33:6d:31:41:f6:b2:
    39:39:a8:44:20:96:04:50:df:02:f9:25:9c:99:fa:
    8c:af:f6:76:11:a5:ef:5d:a6:eb:2a:28:76:6b:f2:
    dd:b5:08:eb:18:9d:7b:74:4d:1a:cc:ea:b5:33:4b:
    b8:20:4c:5a:08:46:1e:a2:aa:c2:8d:4c:3c:b6:e8:
    80:2c:74:6e:0e:08:e1
prime1:
    00:f4:35:ec:2c:b7:30:31:f5:c8:96:3f:30:4a:6d:
    de:39:9c:c8:a9:3c:78:81:40:67:61:e1:3f:7c:0c:
    db:23:05:bb:9d:b5:53:52:0b:af:0a:67:ff:3e:2b:
    94:a5:e3:8f:58:ec:db:ba:7f:c1:f1:b1:65:f3:5a:
    7f:4f:a6:c1:d1
prime2:
    00:d7:c7:97:d0:a0:e9:de:f0:22:2e:12:60:d6:8b:
    43:fe:9b:94:2d:9c:d7:09:01:b1:49:97:50:41:b0:
    06:90:98:f5:ab:2f:3e:e9:45:a8:50:f6:aa:71:ad:
    9e:df:c7:ad:98:f4:14:a7:90:8b:aa:57:b2:ad:f9:
    35:ef:5c:bb:1f
exponent1:
    00:97:14:44:e7:e4:ec:8b:fb:e1:fc:11:5b:6e:b0:
    49:aa:9a:b4:08:fe:c0:0e:36:29:bf:03:fa:a2:63:
    d1:ad:f8:fd:f8:b3:14:b8:cd:41:9c:ce:79:b9:9c:
    7c:79:a4:2e:2a:75:5d:fa:22:cf:79:16:a6:5b:c6:
    c0:bc:26:0e:01
exponent2:
    66:ad:f1:ce:33:64:56:d4:0a:4f:c6:2b:12:ac:be:
    b4:5f:b0:49:6c:42:df:64:50:ca:e6:18:28:c9:ec:
    b5:ce:33:c9:3b:f8:41:e5:05:cd:51:33:96:58:ba:
    2b:5e:c7:7a:eb:5b:10:c9:b5:cc:5f:63:05:6a:7b:
    81:e5:ac:9d
coefficient:
    0f:b4:6e:98:d0:e7:24:ac:17:3d:d6:a1:48:76:10:
    8c:6d:2a:8d:c4:db:1b:c3:4c:2f:21:81:95:f8:3a:
    20:11:84:99:24:1f:69:06:99:d1:5f:f2:8f:c7:43:
    87:34:81:8d:a6:67:88:24:34:ea:a6:1f:ae:d9:d8:
    e1:a5:fb:41
writing RSA key
-----BEGIN RSA PRIVATE KEY-----
MIICWgIBAAKBgQDN17AW+DryKkNN1FVw01tMwnVEsZ/bAu8WYJxa+84Bqfk81lPo
otR2XATpIxIDPrfW3YtNSq4y5o1YTjE48SFrbs52yPuxdJ8gKmB9BHVMqonl1IuB
yfm+TZF9PmQ4VeqkWnndZ2DYXQGWztucavgfclidlLSlrjV496y+OuEjTwIDAQAB
An9+PMjBTXrZCRI9qPq9NuHHMXq5pjVjG2mFT7qUEdlFK/iuVsMe6L1j2Qsp71gc
pVoZa8UCocozbTFB9rI5OahEIJYEUN8C+SWcmfqMr/Z2EaXvXabrKih2a/LdtQjr
GJ17dE0azOq1M0u4IExaCEYeoqrCjUw8tuiALHRuDgjhAkEA9DXsLLcwMfXIlj8w
Sm3eOZzIqTx4gUBnYeE/fAzbIwW7nbVTUguvCmf/PiuUpeOPWOzbun/B8bFl81p/
T6bB0QJBANfHl9Cg6d7wIi4SYNaLQ/6blC2c1wkBsUmXUEGwBpCY9asvPulFqFD2
qnGtnt/HrZj0FKeQi6pXsq35Ne9cux8CQQCXFETn5OyL++H8EVtusEmqmrQI/sAO
Nim/A/qiY9Gt+P34sxS4zUGcznm5nHx5pC4qdV36Is95FqZbxsC8Jg4BAkBmrfHO
M2RW1ApPxisSrL60X7BJbELfZFDK5hgoyey1zjPJO/hB5QXNUTOWWLorXsd661sQ
ybXMX2MFanuB5aydAkAPtG6Y0OckrBc91qFIdhCMbSqNxNsbw0wvIYGV+DogEYSZ
JB9pBpnRX/KPx0OHNIGNpmeIJDTqph+u2djhpftB
-----END RSA PRIVATE KEY-----
```

Shaun Young - N01173292
William Milligan – N00617078
24-NOV-2018
CIS-4360

Step 2: Generate a Certificate Signing Request (CSR). Once the company has the key file, it should generate a Certificate Signing Request (CSR), which basically includes the company's public key. The CSR will be sent to the CA, who will generate a certificate for the key (usually after ensuring that identity information in the CSR matches with the server's true identity). Please use SEEDPKILab2018.com as the common name of the certificate request.

$ openssl req -new -key server.key -out server.csr -config openssl.cnf

```
[11/24/18]seed@VM:~/demoCA$ openssl req -new -key server.key -out server.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]:us
State or Province Name (full name) [Some-State]:fl
Locality Name (eg, city) []:jacksonville
Organization Name (eg, company) [Internet Widgits Pty Ltd]:unf
Organizational Unit Name (eg, section) []:unf
Common Name (e.g. server FQDN or YOUR name) []:unf
Email Address []:n01173292@ospreys.unf.edu

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:pass1
An optional company name []:pass1
[11/24/18]seed@VM:~/demoCA$
```

Shaun Young - N01173292
William Milligan – N00617078
24-NOV-2018
CIS-4360

Step 3: Generating Certificates. The CSR file needs to have the CA's signature to form a certificate. In the real world, the CSR files are usually sent to a trusted CA for their signature. In this lab, we will use our own trusted CA to generate certificates. The following command turns the certificate signing request (server.csr) into an X509 certificate (server.crt), using the CA's ca.crt and ca.key:

$ openssl ca -in server.csr -out server.crt -cert ca.crt -keyfile ca.key \ -config openssl.cnf

```
[11/27/18]seed@VM:~/demoCA$ openssl ca -in server.csr -out server.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: 4096 (0x1000)
        Validity
            Not Before: Nov 27 19:21:16 2018 GMT
            Not After : Nov 27 19:21:16 2019 GMT
        Subject:
            countryName               = us
            stateOrProvinceName       = fl
            localityName              = jacksonville
            organizationName          = unf
            organizationalUnitName    = unf
            commonName                = unf
            emailAddress              = n01173292@ospreys.unf.edu
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                F4:38:94:39:47:C9:87:69:18:AF:1A:29:87:50:C3:35:BF:7E:CB:69
            X509v3 Authority Key Identifier:
                keyid:70:06:A7:8A:28:1E:C6:3B:18:C3:DE:6C:99:99:E4:34:50:31:F6:14

Certificate is to be certified until Nov 27 19:21:16 2019 GMT (365 days)
Sign the certificate? [y/n]:
```

If OpenSSL refuses to generate certificates, it is very likely that the names in your requests do not match with those of CA. The matching rules are specified in the configuration file (look at the [policy match] section). You can change the names of your requests to comply with the policy, or you can change the policy. The configuration file also includes another policy (called policy anything), which is less restrictive. You can choose that policy by changing the following line:

"policy = policy_match" change to "policy = policy_anything".

```
# A few difference way of specifying how similar the request should look
# For type CA, the listed attributes must be the same, and the optional
# and supplied fields are just that :-)
policy          = policy_anything
```

Shaun Young - N01173292
William Milligan – N00617078
24-NOV-2018
CIS-4360

Step 1: Configuring DNS. We choose SEEDPKILab2018.com as the name of our website. To get our computers recognize this name, let us add the following entry to /etc/hosts; this entry basically maps the hostname SEEDPKILab2018.com to our localhost (i.e., 127.0.0.1):

127.0.0.1 SEEDPKILab2018.com

```
[11/27/18]seed@VM:/etc$ cat hosts
127.0.0.1       localhost
127.0.1.1       VM

# 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
127.0.0.1       User
127.0.0.1       Attacker
127.0.0.1       Server
127.0.0.1       www.SeedLabSQLInjection.com
127.0.0.1       www.xsslabelgg.com
127.0.0.1       www.csrflabelgg.com
127.0.0.1       www.csrflabattacker.com
127.0.0.1       www.repackagingattacklab.com
127.0.0.1       www.seedlabclickjacking.com
[11/27/18]seed@VM:/etc$ sudo vim hosts
[sudo] password for seed:
```

Shaun Young - N01173292
William Milligan – N00617078
24-NOV-2018
CIS-4360

Step 2: Configuring the web server. Let us launch a simple web server with the certificate generated in the previous task. OpenSSL allows us to start a simple web server using the s server command:

 # Combine the secret key and certificate into one file
% cp server.key server.pem
% cat server.crt >> server.pem
 # Launch the web server using server.pem
% openssl s_server -cert server.pem -www

By default, the server will listen on port 4433. You can alter that using the -accept option. Now, you can access the server using the following URL: https://SEEDPKILab2018.com:4433/. Most likely, you will get an error message from the browser. In Firefox, you will see a message like the following:

"seedpkilab2018.com:4433 uses an invalid security certificate. The certificate is not trusted because the issuer certificate is unknown".

```
s_server -cert server.pem -www
Secure Renegotiation IS supported
Ciphers supported in s_server binary
TLSv1/SSLv3:ECDHE-RSA-AES256-GCM-SHA384TLSv1/SSLv3:ECDHE-ECDSA-AES256-GCM-SHA384
TLSv1/SSLv3:ECDHE-RSA-AES256-SHA384  TLSv1/SSLv3:ECDHE-ECDSA-AES256-SHA384
TLSv1/SSLv3:ECDHE-RSA-AES256-SHA     TLSv1/SSLv3:ECDHE-ECDSA-AES256-SHA
TLSv1/SSLv3:SRP-DSS-AES-256-CBC-SHA  TLSv1/SSLv3:SRP-RSA-AES-256-CBC-SHA
TLSv1/SSLv3:SRP-AES-256-CBC-SHA      TLSv1/SSLv3:DH-DSS-AES256-GCM-SHA384
TLSv1/SSLv3:DHE-DSS-AES256-GCM-SHA384TLSv1/SSLv3:DH-RSA-AES256-GCM-SHA384
TLSv1/SSLv3:DHE-RSA-AES256-GCM-SHA384TLSv1/SSLv3:DHE-RSA-AES256-SHA256
TLSv1/SSLv3:DHE-DSS-AES256-SHA256    TLSv1/SSLv3:DH-RSA-AES256-SHA256
TLSv1/SSLv3:DH-DSS-AES256-SHA256     TLSv1/SSLv3:DHE-RSA-AES256-SHA
TLSv1/SSLv3:DHE-DSS-AES256-SHA       TLSv1/SSLv3:DH-RSA-AES256-SHA
TLSv1/SSLv3:DH-DSS-AES256-SHA        TLSv1/SSLv3:DHE-RSA-CAMELLIA256-SHA
TLSv1/SSLv3:DHE-DSS-CAMELLIA256-SHA  TLSv1/SSLv3:DH-RSA-CAMELLIA256-SHA
TLSv1/SSLv3:DH-DSS-CAMELLIA256-SHA   TLSv1/SSLv3:ECDH-RSA-AES256-GCM-SHA384
TLSv1/SSLv3:ECDH-ECDSA-AES256-GCM-SHA384TLSv1/SSLv3:ECDH-RSA-AES256-SHA384
TLSv1/SSLv3:ECDH-ECDSA-AES256-SHA384 TLSv1/SSLv3:ECDH-RSA-AES256-SHA
TLSv1/SSLv3:ECDH-ECDSA-AES256-SHA    TLSv1/SSLv3:AES256-GCM-SHA384
TLSv1/SSLv3:AES256-SHA256            TLSv1/SSLv3:AES256-SHA
TLSv1/SSLv3:CAMELLIA256-SHA          TLSv1/SSLv3:PSK-AES256-CBC-SHA
TLSv1/SSLv3:ECDHE-RSA-AES128-GCM-SHA256TLSv1/SSLv3:ECDHE-ECDSA-AES128-GCM-SHA256
TLSv1/SSLv3:ECDHE-RSA-AES128-SHA256  TLSv1/SSLv3:ECDHE-ECDSA-AES128-SHA256
TLSv1/SSLv3:ECDHE-RSA-AES128-SHA     TLSv1/SSLv3:ECDHE-ECDSA-AES128-SHA
TLSv1/SSLv3:SRP-DSS-AES-128-CBC-SHA  TLSv1/SSLv3:SRP-RSA-AES-128-CBC-SHA
TLSv1/SSLv3:SRP-AES-128-CBC-SHA      TLSv1/SSLv3:DH-DSS-AES128-GCM-SHA256
TLSv1/SSLv3:DHE-DSS-AES128-GCM-SHA256TLSv1/SSLv3:DH-RSA-AES128-GCM-SHA256
TLSv1/SSLv3:DHE-RSA-AES128-GCM-SHA256TLSv1/SSLv3:DHE-RSA-AES128-SHA256
TLSv1/SSLv3:DHE-DSS-AES128-SHA256    TLSv1/SSLv3:DH-RSA-AES128-SHA256
TLSv1/SSLv3:DH-DSS-AES128-SHA256     TLSv1/SSLv3:DHE-RSA-AES128-SHA
TLSv1/SSLv3:DHE-DSS-AES128-SHA       TLSv1/SSLv3:DH-RSA-AES128-SHA
TLSv1/SSLv3:DH-DSS-AES128-SHA        TLSv1/SSLv3:DHE-RSA-SEED-SHA
TLSv1/SSLv3:DHE-DSS-SEED-SHA         TLSv1/SSLv3:DH-RSA-SEED-SHA
TLSv1/SSLv3:DH-DSS-SEED-SHA          TLSv1/SSLv3:DHE-RSA-CAMELLIA128-SHA
TLSv1/SSLv3:DHE-DSS-CAMELLIA128-SHA  TLSv1/SSLv3:DH-RSA-CAMELLIA128-SHA
TLSv1/SSLv3:DH-DSS-CAMELLIA128-SHA   TLSv1/SSLv3:ECDH-RSA-AES128-GCM-SHA256
TLSv1/SSLv3:ECDH-ECDSA-AES128-GCM-SHA256TLSv1/SSLv3:ECDH-RSA-AES128-SHA256
TLSv1/SSLv3:ECDH-ECDSA-AES128-SHA256 TLSv1/SSLv3:ECDH-RSA-AES128-SHA
TLSv1/SSLv3:ECDH-ECDSA-AES128-SHA    TLSv1/SSLv3:AES128-GCM-SHA256
TLSv1/SSLv3:AES128-SHA256            TLSv1/SSLv3:AES128-SHA
TLSv1/SSLv3:SEED-SHA                 TLSv1/SSLv3:CAMELLIA128-SHA
TLSv1/SSLv3:PSK-AES128-CBC-SHA       TLSv1/SSLv3:ECDHE-RSA-RC4-SHA
TLSv1/SSLv3:ECDHE-ECDSA-RC4-SHA      TLSv1/SSLv3:ECDH-RSA-RC4-SHA
TLSv1/SSLv3:ECDH-ECDSA-RC4-SHA       TLSv1/SSLv3:RC4-SHA
TLSv1/SSLv3:RC4-MD5                  TLSv1/SSLv3:PSK-RC4-SHA
TLSv1/SSLv3:ECDHE-RSA-DES-CBC3-SHA   TLSv1/SSLv3:ECDHE-ECDSA-DES-CBC3-SHA
TLSv1/SSLv3:SRP-DSS-3DES-EDE-CBC-SHA TLSv1/SSLv3:SRP-RSA-3DES-EDE-CBC-SHA
TLSv1/SSLv3:SRP-3DES-EDE-CBC-SHA     TLSv1/SSLv3:EDH-RSA-DES-CBC3-SHA
TLSv1/SSLv3:EDH-DSS-DES-CBC3-SHA     TLSv1/SSLv3:DH-RSA-DES-CBC3-SHA
TLSv1/SSLv3:DH-DSS-DES-CBC3-SHA      TLSv1/SSLv3:ECDH-RSA-DES-CBC3-SHA
TLSv1/SSLv3:ECDH-ECDSA-DES-CBC3-SHA  TLSv1/SSLv3:DES-CBC3-SHA
TLSv1/SSLv3:PSK-3DES-EDE-CBC-SHA
---
Ciphers common between both SSL end points:
ECDHE-ECDSA-AES128-GCM-SHA256 ECDHE-RSA-AES128-GCM-SHA256 ECDHE-ECDSA-AES256-GCM-SHA384
ECDHE-RSA-AES256-GCM-SHA384 ECDHE-RSA-AES128-SHA       ECDHE-RSA-AES256-SHA
```

Shaun Young - N01173292
William Milligan – N00617078
24-NOV-2018
CIS-4360

Step 4. Testing our HTTPS website. Now, point the browser to https://SEEDPKILab2018.com: 4433. Please describe and explain your observations. Please also do the following tasks:

1. Modify a single byte of server.pem, and restart the server, and reload the URL. What do you observe? Make sure you restore the original server.pem afterward. Note: the server may not be able to restart if certain places of server.pem is corrupted; in that case, choose another place to modify.

```
[11/27/18]seed@VM:~/demoCA$ openssl s_server -cert server.pem -www
unable to load server certificate private key file
3070764736:error:0906D06C:PEM routines:PEM_read_bio:no start line:pem_lib.c:701:
Expecting: ANY PRIVATE KEY
[11/27/18]seed@VM:~/demoCA$
```

2. Since SEEDPKILab2018.com points to the localhost, if we use https://localhost:4433 instead, we will be connecting to the same web server. Please do so, describe and explain your observations.

Shaun Young - N01173292
William Milligan – N00617078
24-NOV-2018
CIS-4360

2.4 Task4: Deploying Certificate in an Apache-Based HTTPS Website

For this part of the lab we had to host our new website locally while ensuring that it could be accessed through HTTP. The first stop of this process is to ensure that the desired name www.SEEDPKILab2018.com is on the /etc/hosts file. This means that the CDN www.SEEDPKILab2018.com is an alias for localhost.

```
127.0.0.1       localhost
127.0.1.1       VM

# 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
127.0.0.1       User
127.0.0.1       Attacker
127.0.0.1       Server
127.0.0.1       www.SeedLabSQLInjection.com
127.0.0.1       www.xsslabelgg.com
127.0.0.1       www.csrflabelgg.com
127.0.0.1       www.csrflabattacker.com
127.0.0.1       www.repackagingattacklab.com
127.0.0.1       www.seedlabclickjacking.com
127.0.0.1       www.SEEDPKILab2018.com
127.0.0.1       www.FamousBank.com
127.0.0.1       www.Google.com
~
```

The next step in this process is the charge the /etc/apache2/sites-available/default-ssl.conf file to provide the apache sever with the necessary files. The configuration fields include the document root, which is the file system the server will consider ./, the Directory index which is the first file that will be shown when people visit www.SEEDPKILab2018.com, then the configuration for SSL including the locations of the SSL Certificate File of the website, and the same certificate's key.

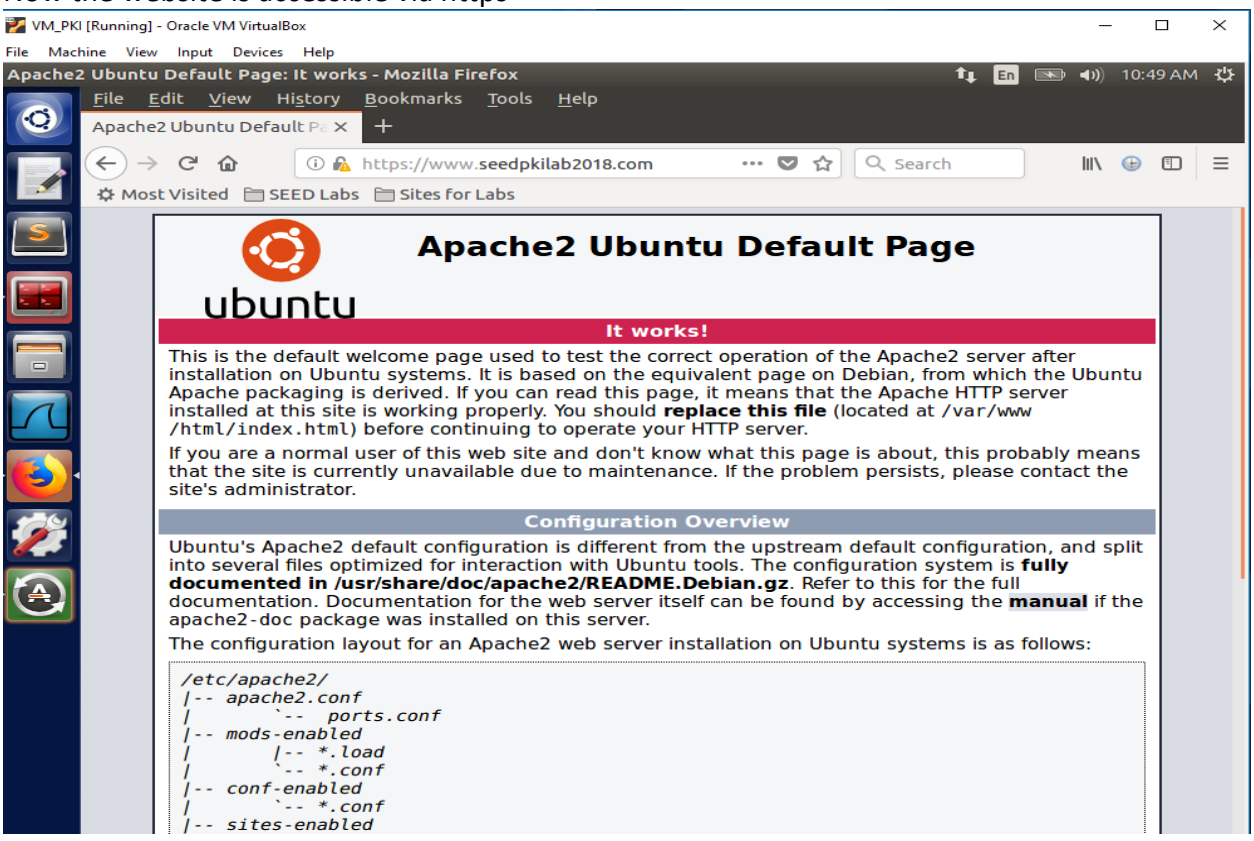
```
<VirtualHost *:443 >
        ServerName www.SEEDLABPKI2018.com
        DocumentRoot /var/www/html
        DirectoryIndex index.html

        SSLEngine On
        SSLCertificateFile /home/seed/demoCA/serverCertificate.crt
        SSLCertificateKeyFile /home/seed/demoCA/keyForServer.key
</VirtualHost>
```

Shaun Young - N01173292
William Milligan – N00617078
24-NOV-2018
CIS-4360

Once this file has been successfully configured a series of commands can be run, to enable SSL, the terminal command will restart the apache server and the user will be prompted to enter the private key of the certificate.

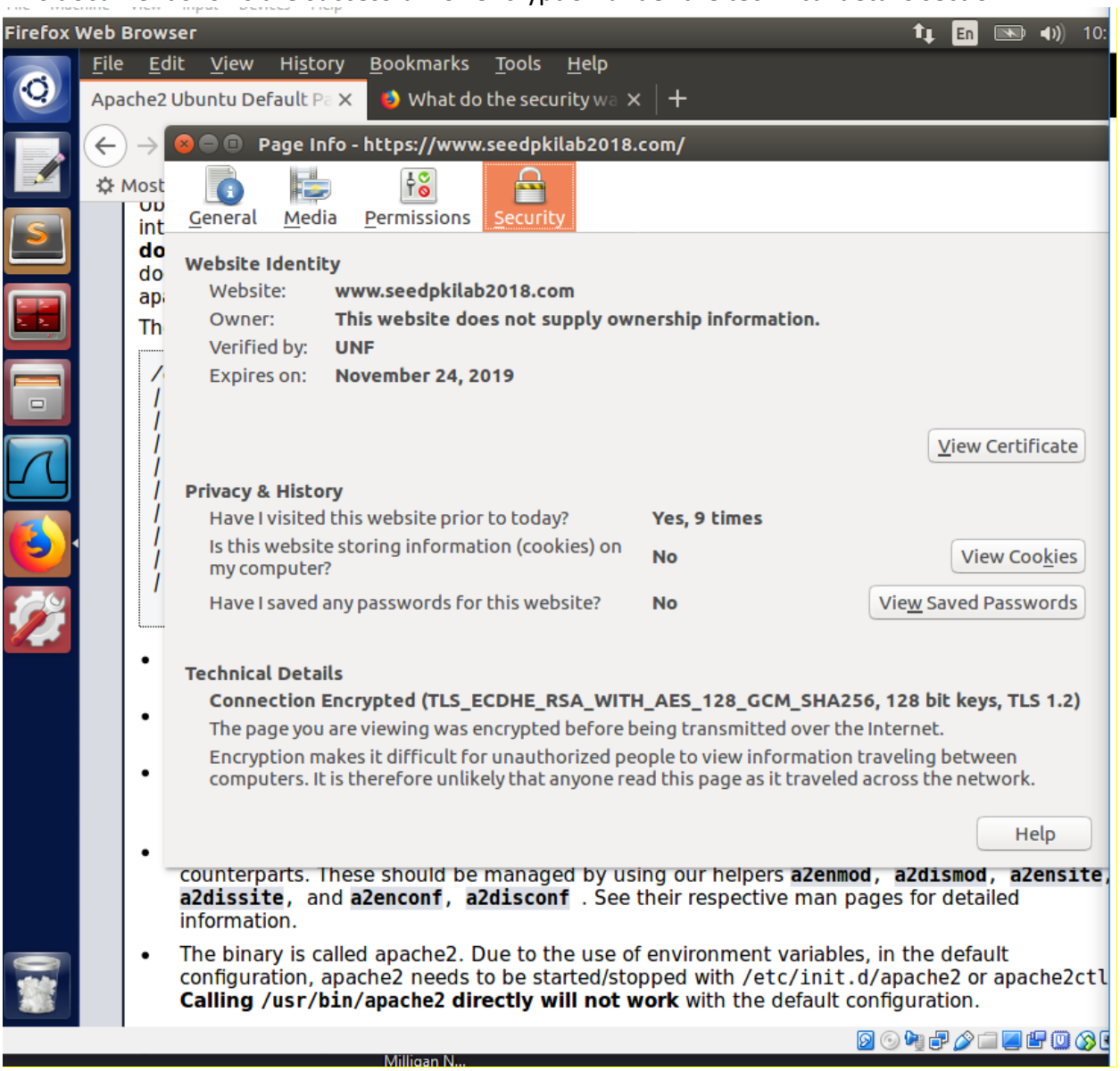
```
[11/30/18]seed@VM:.../sites-available$ sudo vim default-ssl.conf
[sudo] password for seed:
[11/30/18]seed@VM:.../sites-available$ sudo a2enmod ssl
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Module socache_shmcb already enabled
Module ssl already enabled
[11/30/18]seed@VM:.../sites-available$ sudo a2ensite default-ssl
Site default-ssl already enabled
[11/30/18]seed@VM:.../sites-available$ sudo service apache2 resta
rt

Enter passphrase for SSL/TLS keys for www.SEEDLABPKI2018.com:443
(RSA):
Enter passphrase for SSL/TLS keys for www.SEEDLABPKI2018.com:443
(RSA): ********
[11/30/18]seed@VM:.../sites-available$
```

Now the website is accessible via https



Shaun Young - N01173292
William Milligan – N00617078
24-NOV-2018
CIS-4360

This document shows the successful TSL encryption under the technical details section.



Shaun Young - N01173292
William Milligan – N00617078
24-NOV-2018
CIS-4360

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

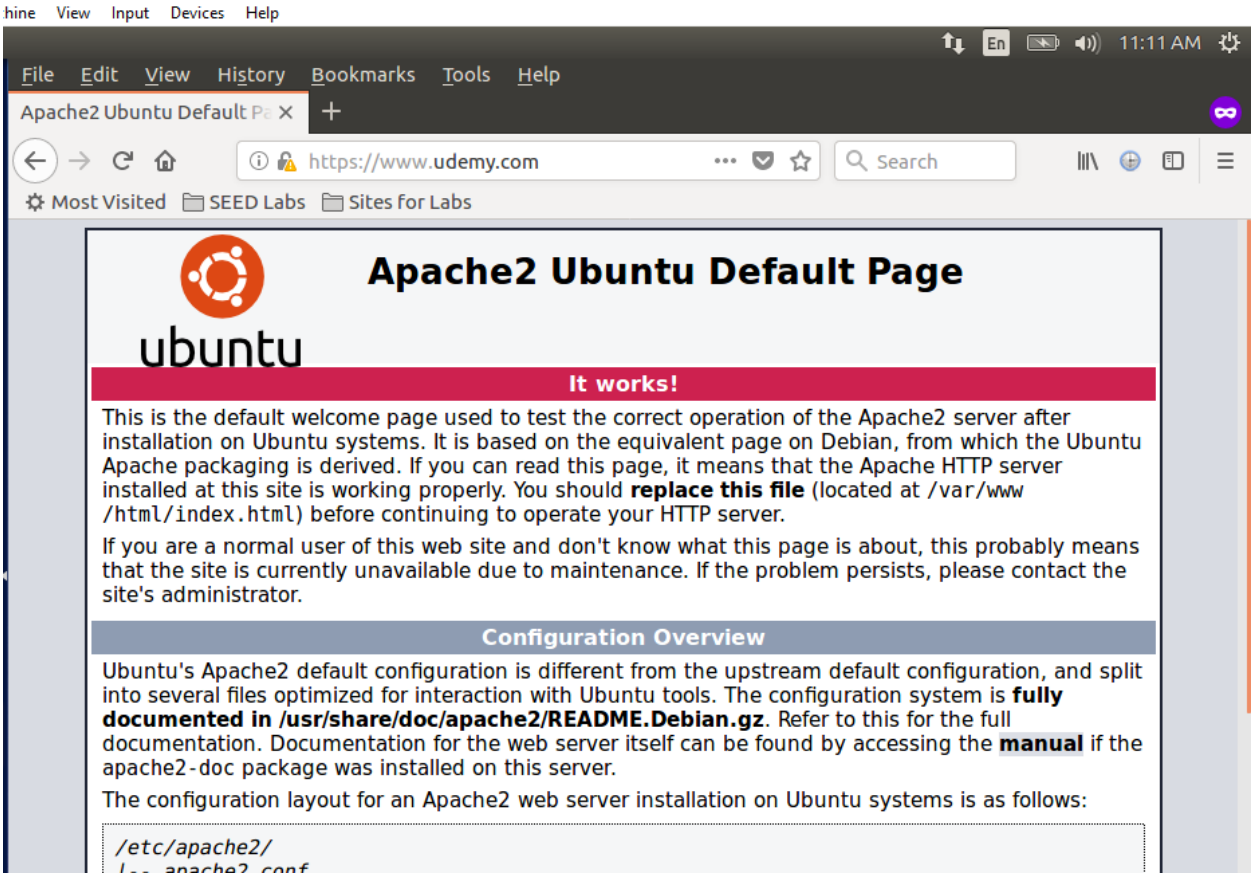
Here we are emulating a man in the middle attack. In order to do this we are going to “attack” the local user’s PC and make an entry I their /etc/hosts so that when they visit a site they would commonly visit then they will be redirected back our malicious site. This requires the modification of the hosts file and a certificate for the site they will be redirected to.

In Order to do this a new entry must be made in the Hosts file (the host file of the target) Here you can see the addition of Udemy.com

After that the Apace server is configured to have Udemy as a webpage that it servers.

Now whenever the target tries to go to Udemy they are instead sent to my application instead, though they think they are at Udemy . com As Seen Here :

Shaun Young - N01173292
William Milligan – N00617078
24-NOV-2018
CIS-4360



Shaun Young - N01173292
William Milligan – N00617078
24-NOV-2018
CIS-4360

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

In short if a CA's private key was to be compromised then anyone would be able to generate a certificate on their own using the compromised CA's private key. The first step in this experiment would be to design a CSR for a site that the attacker would like to impersonate, in this case Udemy.com.

```
CAuthority.key  index.txt.attr    serial
CertAuthor.crt  index.txt.old     serial.old
certs           keyForServer.key  serverCertificate.crt
crl             newcerts          server.csr
index.txt       openssl.conf      UServerPrivateKey.key
[11/30/18]seed@VM:~/demoCA$ openssl req -new -key UServerPrivateK
ey.key -out UServer.csr -config openssl.conf
Enter pass phrase for UServerPrivateKey.key:
You are about to be asked to enter information that will be incor
porated
into your certificate request.
What you are about to enter is what is called a Distinguished Nam
e 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]:Fl
Locality Name (eg, city) []:Jacksonville
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Udemy
Organizational Unit Name (eg, section) []:Udemy Online Services
Common Name (e.g. server FQDN or YOUR name) []:www.Udemy.com
Email Address []:nope

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:password
An optional company name []:Udemy
[11/30/18]seed@VM:~/demoCA$
```

Now we can make a copy of our CA's private key to show that it has been compromised and use that to sign the certificate. Normally these two steps would be done by the CA, but since we have compromised the key, we can do it ourselves. First we sign the csr using the compromised private key.

Shaun Young - N01173292
William Milligan – N00617078
24-NOV-2018
CIS-4360

```
[11/30/18]seed@VM:~/demoCA$ cp CAuthority.key CompromisedKey.key
[11/30/18]seed@VM:~/demoCA$ dir
CAuthority.key      crl            keyForServer.key  serial.old            UServerPrivateKey.key
CertAuthor.crt      index.txt      newcerts          serverCertificate.crt
certs               index.txt.attr openssl.conf      server.csr
CompromisedKey.key  index.txt.old  serial            UServer.csr
[11/30/18]seed@VM:~/demoCA$
```

```
[11/30/18]seed@VM:~$ openssl ca -in ./demoCA/UServer.csr -out Hack
edCert.cert -cert ./demoCA/CertAuthor.crt -keyfile ./demoCA/Compro
misedKey.key -config ./demoCA/openssl.conf
Using configuration from ./demoCA/openssl.conf
Enter pass phrase for ./demoCA/CompromisedKey.key:
Check that the request matches the signature
Signature ok
Certificate Details:
```

This allows for us to send a valid request to the CA. Now we get the response

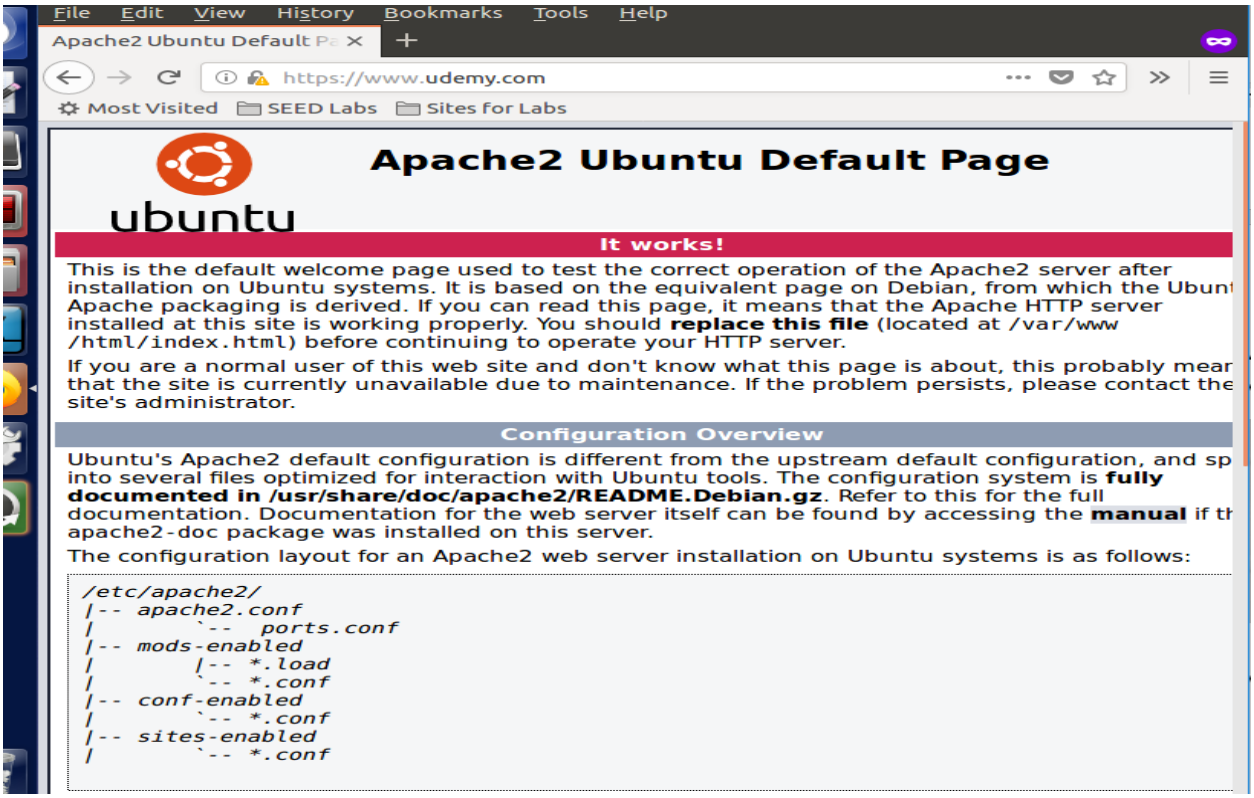
```
CAuthority.key  index.txt.attr    serial
CertAuthor.crt  index.txt.old     serial.old
certs           keyForServer.key  serverCertificate.crt
crl             newcerts          server.csr
index.txt       openssl.conf      UServerPrivateKey.key
[11/30/18]seed@VM:~/demoCA$ openssl req -new -key UServerPrivateK
ey.key -out UServer.csr -config openssl.conf
Enter pass phrase for UServerPrivateKey.key:
You are about to be asked to enter information that will be incor
porated
into your certificate request.
What you are about to enter is what is called a Distinguished Nam
e 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]:Fl
Locality Name (eg, city) []:Jacksonville
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Udemy
Organizational Unit Name (eg, section) []:Udemy Online Services
Common Name (e.g. server FQDN or YOUR name) []:www.Udemy.com
Email Address []:nope

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:password
An optional company name []:Udemy
[11/30/18]seed@VM:~/demoCA$
```

So we have created a fake certificate using the private key of the CA!

Now we change our server to serve this new key.

```
        <VirtualHost *:443 >
                ServerName www.Udemy.com
                DocumentRoot /var/www/html
                DirectoryIndex index.html

                SSLEngine On
                SSLCertificateFile /home/seed/HackedCert.cert
                SSLCertificateKeyFile /home/seed/demoCA/keyForServ
er.key
        </VirtualHost>
```

Now when we restart apache and try to access this site we get this response.



Showing the browser was not able to tell the certificate used to validate the site is in fact from a compromised CA.

Shaun Young - N01173292
William Milligan – N00617078
24-NOV-2018
CIS-4360