



Project: Configuration of Certification Authority and Implementation of Transport Layer Security over HTTP

Course: Cybersecurity, law and Ethics

Code: CSE487

Sec: 01.

Semester: Summer 2022.

Submitted By,

Iftakhar Ahmed Mahin (2019-1-60-206)

Nahian Niger Siddiqua (2019-2-60-074)

Submitted to,

Rashedul Amin Tuhin

Senior Lecturer, Department of Computer Science and
Engineering, East West University.

Date of Submission: 21-08-2022

Apache web server configuration:

- `sudo apt-get update`
- `Sudo apt-get install apache2`
- To continue enter y

To see Ip address write

- `ifconfig`

To start apache2 server write

- `sudo systemctl start apache2`

To start apache2 server write

- `sudo systemctl stop apache2`

To restart apache2 server write

- `sudo systemctl restart apache2`

To edit index.html

- `cd var/www`
- `sudo nano index.html`

DNS server configuration:

- `sudo apt install bind9`

To see all the file, write:

- `ls`

To configure IP address

- `sudo nano/etc/hosts`

To verify file content

- cat /etc/hosts

To check host name

- hostname

To see dns domain name

- Dnsdomainname

We have to config the“named.conf.options” file

- sudo nano named.conf.options

Then insert the code below

```
21     'dnssec-validation auto;
22
23     listen-on-v6 { any; };
24
25     recursion yes;
26     listen-on {192.168.28.96;};
27     allow-transfer {none;};
28
29     forwarders {
30         8.8.8.8;
31         8.8.4.4;
32     };
33
34
35 };
36
```

We have to config the “named.conf.local” file

- sudo nano named.conf.local

```
1 //
2 // Do any local configuration here
3 //
4
5 // Consider adding the 1918 zones here, if they are not used in your
6 // organization
7 //include "/etc/bind/zones.rfc1918";
8 //forward lookup zone
9 zone "acme1.com" IN{
10     type master;
11     file "/etc/bind/db.nahian.local";
12 };
13
14 //reverse lookup zone
15
16 zone "28.168.192.in-addr.arpa" IN{
17     type master;
18     file "/etc/bind/db.2.0.10";
19 };
```

To check the configurations

- Named-checkconf

Next we have to config the “db.nahian.local” file

- sudo nano db.nahian.local

```
1 ;
2 ; BIND data file for local loopback interface
3 ;
4 $TTL      604800
5 @         IN      SOA      ns1.acme1.com. root.acme1.com. (
6                                     2           ; Serial
7                                     604800      ; Refresh
8                                     86400       ; Retry
9                                     2419200     ; Expire
10                                    604800 )     ; Negative Cache TTL
11 ;
12 @         IN      NS       ns1.acme1.com.
13 ns1       IN      A        192.168.28.96
14 www       IN      A        192.168.28.96
15 @         IN      AAAA     fe80::a91d:b90a:4d15:5b8f|
```

Next, we have to config the “db.2.0.10” file

- `sudo nano db.2.0.10`

```
1 ;
2 ; BIND reverse data file for local loopback interface
3 ;
4 $TTL      604800
5 @         IN      SOA      ns1.acme1.com. root.acme1.com. (
6                               1          ; Serial
7                               604800     ; Refresh
8                               86400      ; Retry
9                               2419200    ; Expire
10                              604800 )    ; Negative Cache TTL
11 ;
12 @         IN      NS       ns1.acme1.com.
13 96        IN      PTR      ns1.acme1.com.
14 96        IN      PTR      www.acme1.com.
15
```

5

To restart the DNS server

- `Sudo service bind9 restart`

To check our DNS server is active or not

- `Sudo service bind9 status`

To find www.acme1.com

- `nslookup www.acme1.com`

To permanent the edit name server and resolv.conf file,we have to remove the resolv.conf file

- `sudo rm /etc/resolv.conf`

Then we will link a resolv.conf which is under systemd and result folder

- Sudo ln -sf /run/systemd/resolve/resolv.conf /etc/resolv.conf

Now we have to edit “resolv.conf” file under etc

- Sudo nano /etc/resolv.conf

If nameserver is not here then we have to add the nameserver

```
1 nameserver 192.168.28.96
2 search acme1.com
3 |
```

- nslookup www.acme1.com

Now we can see it resolved and it coming from our server

```
nahian@nahian:~$ nslookup www.acme1.com
Server:          192.168.28.96
Address:         192.168.28.96#53

Non-authoritative answer:
Name:   www.acme1.com
Address: 35.186.238.101
```

- ping www.acme1.com

```
nahian@nahian:~$ ping www.acme1.com
PING www.acme1.com (35.186.238.101) 56(84) bytes of data.
64 bytes from 101.238.186.35.bc.googleusercontent.com (35.186.238.101): icmp_seq=1 ttl=115 time=55.8 ms
64 bytes from 101.238.186.35.bc.googleusercontent.com (35.186.238.101): icmp_seq=2 ttl=115 time=153 ms
64 bytes from 101.238.186.35.bc.googleusercontent.com (35.186.238.101): icmp_seq=3 ttl=115 time=55.7 ms
64 bytes from 101.238.186.35.bc.googleusercontent.com (35.186.238.101): icmp_seq=4 ttl=115 time=61.8 ms
64 bytes from 101.238.186.35.bc.googleusercontent.com (35.186.238.101): icmp_seq=5 ttl=115 time=58.2 ms
64 bytes from 101.238.186.35.bc.googleusercontent.com (35.186.238.101): icmp_seq=6 ttl=115 time=57.7 ms
```

Firewall Configuration:

To enable uncomplicated firewall

- `sudo ufw enable`

To allow bind9

- `sudo ufw allow Bind9`

To check status

- `sudo ufw status`
- `sudo apt install ssh`

To allow some others ports

- `sudo ufw allow 21`
- `sudo ufw allow 10`
- `sudo ufw allow 20`

For generate certificates

Moving to the root using

- `sudo -i`

See tree of files inside the root:

- `tree`

Creating directory:

- `mkdir -p ca/{root-ca,sub-ca,server}/{private,certs,newcerts,crl,csr}`

Changing the root of ca and sub ca private folder

- `chmod -v 700 ca/{root-ca,sub-ca,server}/private`

Creating file index in both root ca and sub ca

- `touch ca/{root-ca,sub-ca}/index`

Generating hexadecimal random number of 16 character

writing serial number of root ca

- `openssl rand -hex 16 > ca/root-ca/serial`

writing serial number of sub ca

- `openssl rand -hex 16 > ca/sub-ca/serial`

moving to ca directory

- `cd ca`

1. Generating private key for root ca, sub ca and server

- `openssl genrsa -aes256 -out root-ca/private/ca.key 4096`
- `openssl genrsa -aes256 -out sub-ca/private/sub-ca.key 4096`
- `openssl genrsa -out server/private/server.key 2048`

Reviewing the change

- `tree`

Creating root ca.configd

- `sudo nano root-ca/root-ca.conf`

Inserting the code

- code: 1. root-ca.conf code

- Save and exit
- tree

Moving inside root-ca

- cd root-ca

2. Generating root certificates

- openssl req -config root-ca.conf -key private/ca.key -new -x509 -days 3650 -sha256 -extensions v3_ca -out certs/ca.crt

Ensuring that the certificate has been created

- properly** • openssl x509 -noout -in certs/ca.crt -text

Moving a step back and then to sub-ca

- cd ../sub-ca

Creating sub-ca.config

- sudo nano sub-ca.conf

Inserting the code into sub-ca.config file

- code: 2. sub-ca.conf code
- Saving and exiting

Seeing the directory once again

- tree

Requesting for sub ca certificate signing request.

- openssl req -config sub-ca.conf -new -key private/sub-ca.key -sha256 -out csr/sub-ca.csr

Moving to the previous folder

- cd -

Signing the request of sub ca by root ca

- `openssl ca -config root-ca.conf -extensions v3_intermediate_ca -days 3650 -notext -in ../sub-ca/csr/sub-ca.csr -out ../sub-ca/certs/sub-ca.crt`
- to confirm insert y

See directory

- Tree
- →.pem file has been generated

See the list of signing

- `cat index`
- →Root ca signed sub ca

Seeing detail

- `openssl x509 -noout -text -in ../sub-ca/certs/sub-ca.crt`

3. Configuring server

Moving to server

- `cd ../server`
- `sudo nano server.conf`

Generating certificate signing request from server

- `openssl req -config server.conf -key private/server.key -new -sha256 -out csr/server.csr`
- `cd ../sub-ca`

Sub ca signing certificate request of server

- `openssl ca -config sub-ca.conf -extensions server_cert -days 3650 -notext -in ../server/csr/server.csr -extensions req_ext -extfile ../server/server.conf -out ../server/certs/server.crt`

Seeing detail

- `cat index`
- `cd ../server/certs`

Now, concating sub-ca.crt and server.crt and naming the new file chained.crt

- `cat server.crt ../../sub-ca/certs/sub-ca.cert > chained.crt`
- `openssl x509 -noout -text -in server.crt`

Next, we have to config the 000-default.conf file

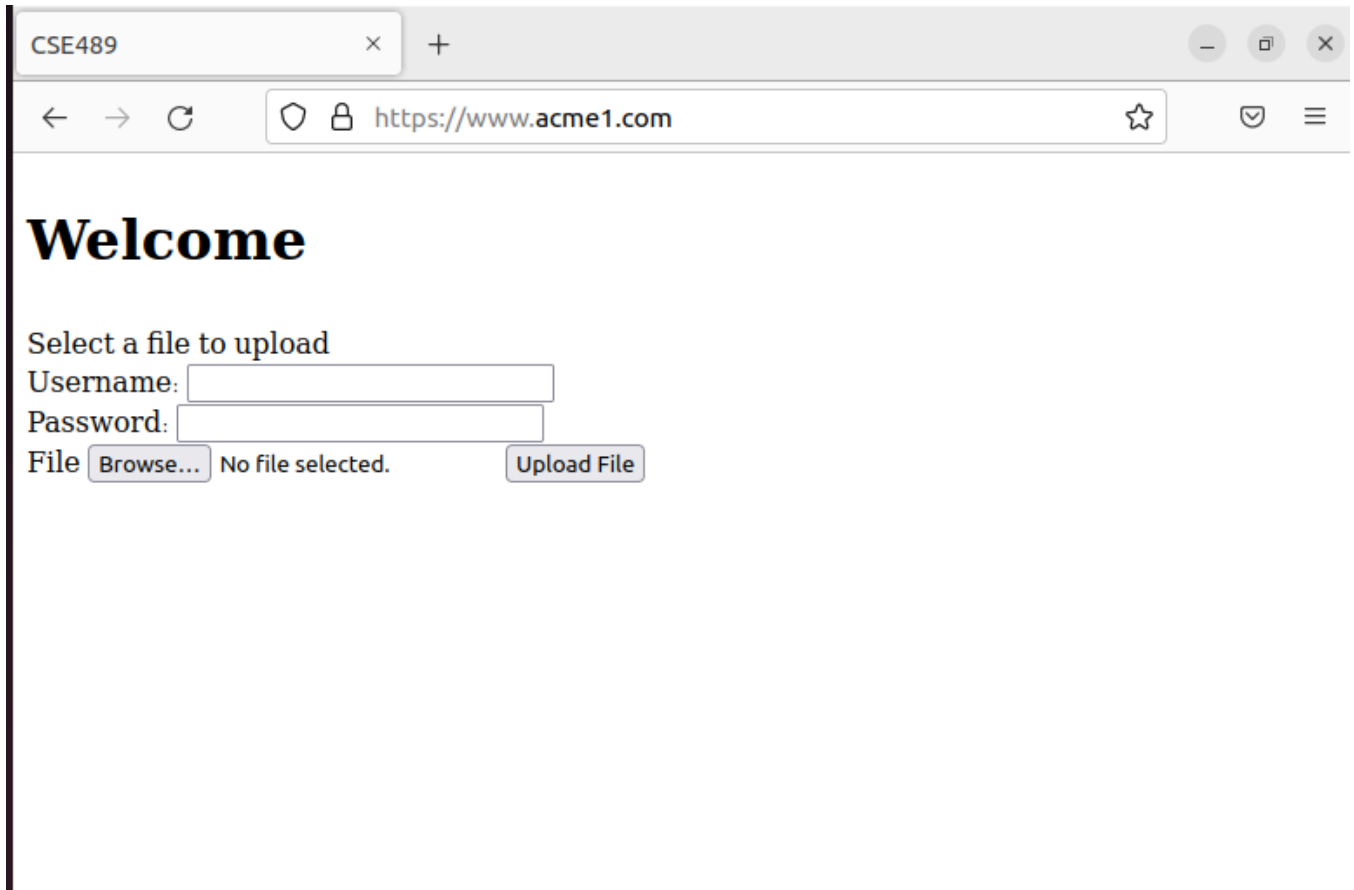
```
1 <VirtualHost *:443 *:333 *:480>
2     SSLEngine on
3     SSLCertificateFile /root/ca/server/certs/chained.crt
4     SSLCertificateKeyFile /root/ca/server/private/server.key
5     servername www.acme1.com
6     Documentroot /var/www/html
7 </VirtualHost>
8
9 <VirtualHost *:80>
10     servername www.acme1.com
11     #Redirect permanent / https://www.acme1.com
12 </VirtualHost>
13
14
```

Copying the “ca.crt” and “sub-ca.crt” into home

- `cd ca/root-ca/certs`
- `cp ca.crt /home/nahian/certs`






- cd
- cd ca/sub-ca/certs
- cp sub-ca.crt /home/nahian/certs

Now on the browser Settings → privacy and security → view certificate → authorities → import → select the file → open → select purpose → {view: to see the certificate} → OK



The screenshot shows a web browser window with a single tab titled 'CSE489'. The address bar displays 'https://www.acme1.com'. The page content includes a large 'Welcome' heading, followed by the text 'Select a file to upload'. Below this are input fields for 'Username:' and 'Password:'. At the bottom, there is a 'File' section with a 'Browse...' button, the text 'No file selected.', and an 'Upload File' button.

CSE489

← → ↻   https://www.acme1.com   

Welcome

Select a file to upload

Username:

Password:

File No file selected.

Codes of ROOT CA & SUB-CA

ROOT_CA:

[ca]

```
#/root/ca/root-ca/root-ca.conf  
#see man ca  
default_ca = CA_default
```

[CA_default]

```
dir = /root/ca/root-ca  
certs = $dir/certs  
crl_dir = $dir/crl  
new_certs_dir = $dir/newcerts
```

```
database = $dir/index  
serial = $dir/serial  
RANDFILE = $dir/private/.rand  
private_key = $dir/private/ca.key  
certificate = $dir/certs/ca.crt
```

```
crlnumber = $dir/crlnumber  
crl = $dir/crl/ca.crl  
crl_extensions = crl_ext  
default_crl_days = 30  
default_md = sha256
```

```
name_opt = ca_default  
cert_opt = ca_default  
default_days = 365
```

preserve = no
policy = policy_strict

[policy_strict]

countryName = supplied
stateOrProvinceName = supplied
organizationName = match
organizationalUnitName = optional
commonName = supplied
emailAddress = optional

[policy_loose]

countryName = optional
stateOrProvinceName = optional
localityName = optional
organizationName = optional
organizationalUnitName = optional
commonName = supplied
emailAddress = optional

[req]

Options for the req tool, man req.

default_bits = 2048
distinguished_name = req_distinguished_name
string_mask = utf8only
default_md = sha256

Extension to add when the -x509 option is used.
x509_extensions = v3_ca

[req_distinguished_name]

countryName = Country Name (2 letter code)
stateOrProvinceName = State or Province Name
localityName = Locality Name
0.organizationName = Organization Name
organizationalUnitName = Organizational Unit Name
commonName = Common Name
emailAddress = Email Address

```
countryName_default = BD
stateOrProvinceName_default = Dhaka
0.organizationName_default = NNS
commonName_default = RootCA
```

```
[ v3_ca ]
```

```
# Extensions to apply when creating root ca
# Extensions for a typical CA, man x509v3_config
```

```
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer
basicConstraints = critical, CA:true
keyUsage = critical, digitalSignature, cRLSign, keyCertSign
```

```
[ v3_intermediate_ca ]
```

```
# Extensions to apply when creating intermediate or sub-ca
```

```
# Extensions for a typical intermediate CA, same man as above
```

```
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer
```

```
#pathlen:0 ensures no more sub-ca can be created below an intermediate
```

```
basicConstraints = critical, CA:true, pathlen:0
keyUsage = critical, digitalSignature, cRLSign, keyCertSign
```

```
[ server_cert ]
```

```
# Extensions for server certificates
```

```
basicConstraints = CA:FALSE
nsCertType = server
nsComment = "OpenSSL Generated Server Certificate"
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid,issuer:always
keyUsage = critical, digitalSignature, keyEncipherment
extendedKeyUsage = serverAuth
```

SUB_CA:

[ca]

#/root/ca/sub-ca

#see man ca

default_ca = CA_default

[CA_default]

dir = /root/ca/sub-ca

certs = \$dir/certs

crl_dir = \$dir/crl

new_certs_dir = \$dir/newcerts

database = \$dir/index

serial = \$dir/serial

RANDFILE = \$dir/private/.rand

private_key = \$dir/private/sub-ca.key

certificate = \$dir/certs/sub-ca.crt

crlnumber = \$dir/crlnumber

crl = \$dir/crl/ca.crl

crl_extensions = crl_ext

default_crl_days = 30

default_md = sha256

name_opt = ca_default

cert_opt = ca_default

default_days = 365

preserve = no

policy = policy_loose

[policy_strict]

countryName = supplied
stateOrProvinceName = supplied
organizationName = match
organizationalUnitName = optional
commonName = supplied
emailAddress = optional

[policy_loose]

countryName = optional
stateOrProvinceName = optional

localityName = optional
organizationName = optional
organizationalUnitName = optional
commonName = supplied
emailAddress = optional

[req]

Options for the req tool, man req.

default_bits = 2048
distinguished_name = req_distinguished_name
string_mask = utf8only
default_md = sha256

Extension to add when the -x509 option is used.

x509_extensions = v3_ca

[req_distinguished_name]

countryName = Country Name (2 letter code)
stateOrProvinceName = State or Province Name
localityName = Locality Name
0.organizationName = Organization Name
organizationalUnitName = Organizational Unit Name

**commonName = Common Name emailAddress = Email
Address
countryName_default = BD
stateOrProvinceName_default = Dhaka
0.organizationName_default = NNS
commonName_default = SubCA**

[v3_ca]

**# Extensions to apply when createing root ca #
Extensions for a typical CA, man x509v3_config**

**subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer**

**basicConstraints = critical, CA:true
keyUsage = critical, digitalSignature, cRLSign, keyCertSign**

[v3_intermediate_ca]

**# Extensions to apply when creating intermediate or sub-ca #
Extensions for a typical intermediate CA, same man as above**

**subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer**

#pathlen:0 ensures no more sub-ca can be created below an intermediate

**basicConstraints = critical, CA:true, pathlen:0
keyUsage = critical, digitalSignature, cRLSign, keyCertSign**

[server_cert]

Extensions for server certificates

**basicConstraints = CA:FALSE
nsCertType = server
nsComment = "OpenSSL Generated Server Certificate"
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid,issuer:always
keyUsage = critical, digitalSignature, keyEncipherment
extendedKeyUsage = serverAuth**

Server

```
[ req ]  
default_bits = 2048  
distinguished_name = req_distinguished_name  
req_extensions = req_ext  
prompt = no
```

```
[ req_distinguished_name ]  
countryName = BD  
stateOrProvinceName = Dhaka  
organizationName = NNS  
commonName = www.acme1.com
```

```
[ req_ext ]  
subjectAltName = @alt_names
```

```
[ alt_names ]  
DNS.1 = nahian.local  
DNS.2 = www.acme1.com  
IP.1 = 192.168.28.96
```

