**East West University**

**Group Members**

MD.Amir Abdal Sobhani

ID: 2019-1-60-023

Sreya Deb Srestha

ID: 2019-1-60-032

Syeda Nawar Shamaem

ID: 2019-1-60-110

Submitted To:

Rashedul Amin Tuhin

Senior Lecturer

Department of Computer Science and Engineering

Submission date: 25/08/22

**Configuration of Certification Authority and Implementation of Transport Layer Security over HTTP**

Step1: install tree and curl in the linux machine.

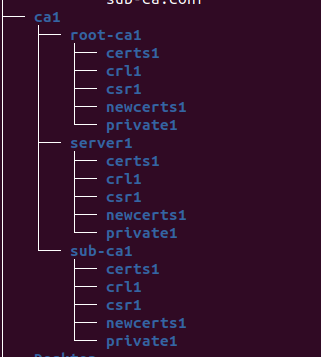
Step2: make the directories using:

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



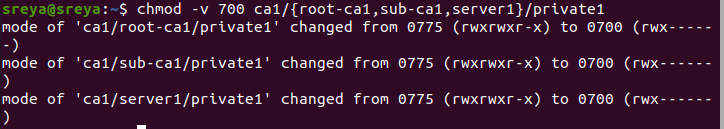
Step3: check if all the directories are created or not using:

tree ca



Step 4: Changing the root of ca and sub ca private folder using:

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



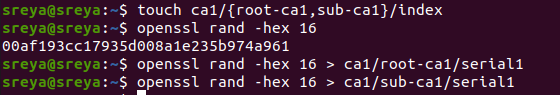
Step 5: Creating file index in both root ca and sub ca and serials for rrot-ca and sub-ca using:

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

openssl rand -hex 16

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

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



Step 6: Check if indexes are created for root-ca and sub-ca:

tree ca

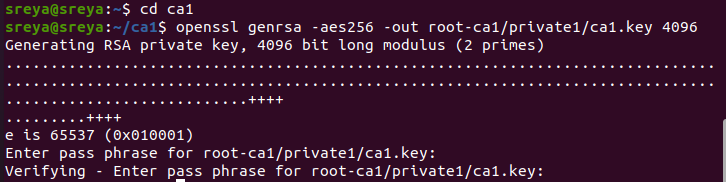


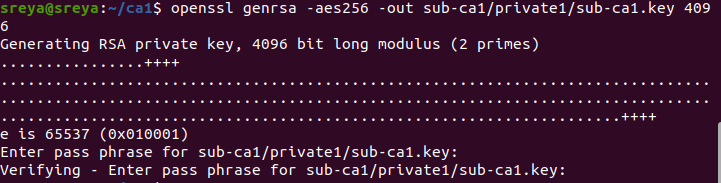
Step 7: Create private keys for ca, sub-ca and server using:

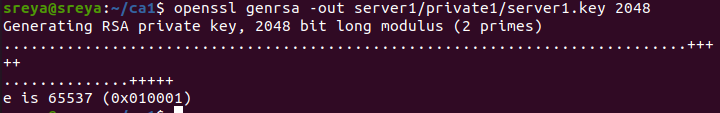
openssl genrsa -aes256 -out root-ca/private/ca.key 4096

openssl genrsa -aes256 -out sub-ca/private/sub-ca.key 4096

openssl genrsa -out server/private/server.key 2048

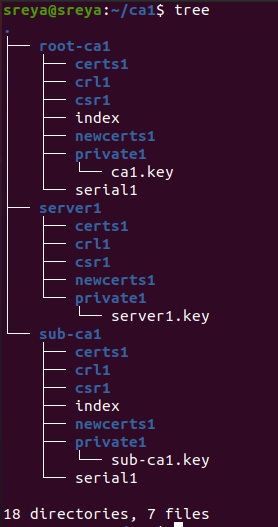






Step 8: check if the key were generated or not using:

tree ca



Step 9: create a root.conf file in root-ca folder:

vim root-ca/root-ca.conf



Step 10: Insert the following code in root-ca.conf:

[ca]

#/root/ca/root-ca/root-ca.conf

#see man ca

default\_ca = CA\_default

[CA\_default]

dir = /home/sobhani/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 = GB

stateOrProvinceName\_default = England

0.organizationName\_default = TheUrbanPenguin Ltd

[ 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

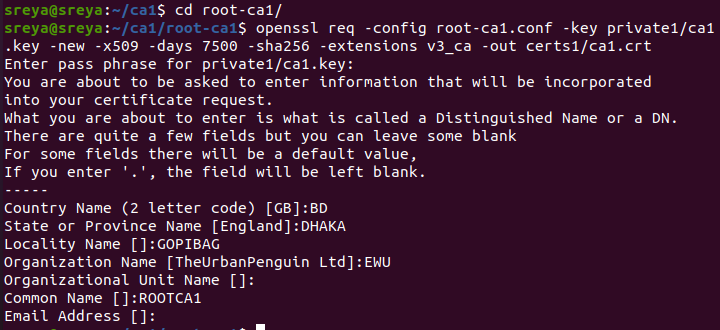
authorityKeyIdentifier = keyid,issuer:always

keyUsage = critical, digitalSignature, keyEncipherment

extendedKeyUsage = serverAuth

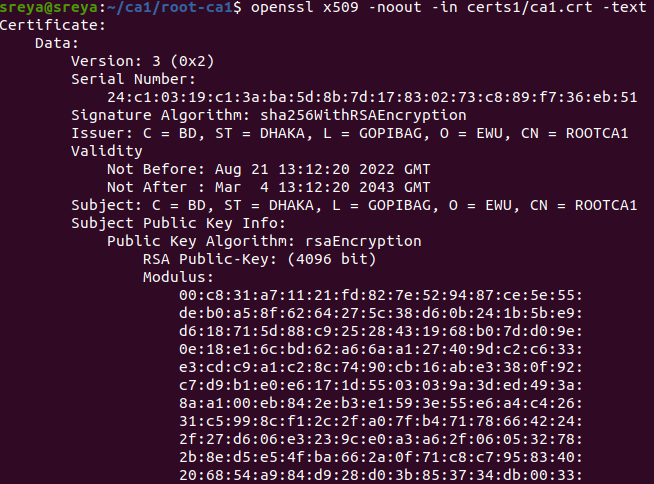
Step 11: Generate root certificate:

openssl req -config root-ca.conf -key private/ca.key -new -x509 -days 7500 -sha256 -extensions v3\_ca -out certs/ca.crt



Step 12: Check the ca.crt:

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



Step 12: create sub-ca.conf:

vim sub-ca.conf



Step 13: Import this code in sub-ca.conf:

[ca]

#/root/ca/root-ca/sub-ca.conf

#see man ca

default\_ca = CA\_default

[CA\_default]

dir = /home/sobhani/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 = GB

stateOrProvinceName\_default = England

0.organizationName\_default = TheUrbanPenguin Ltd

[ 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

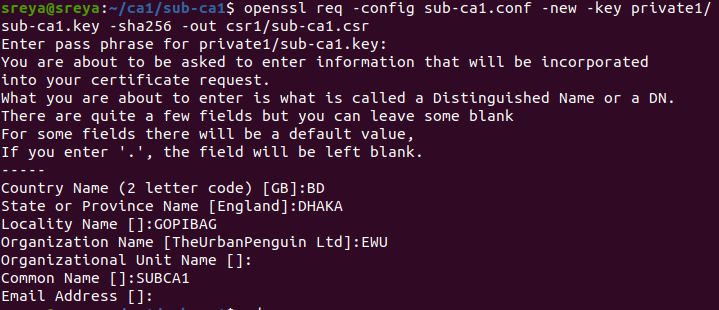
authorityKeyIdentifier = keyid,issuer:always

keyUsage = critical, digitalSignature, keyEncipherment

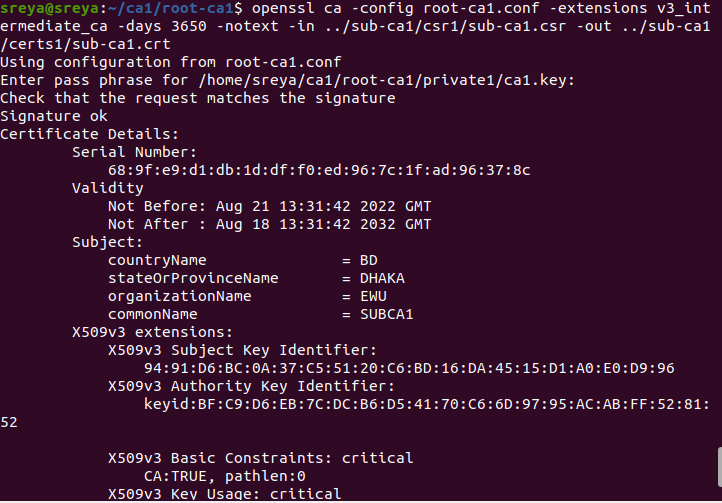
extendedKeyUsage = serverAuth

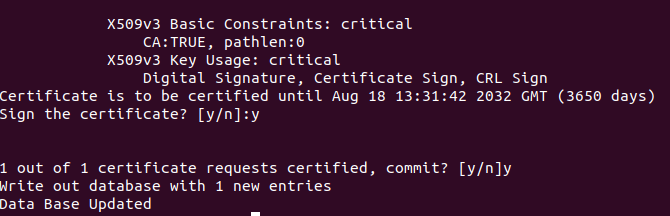
Step 14: Certificate signing request from RootCA to SubCA:

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

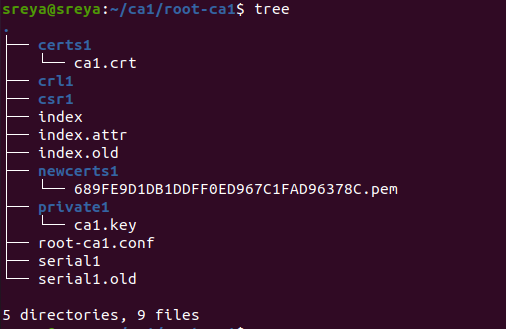


openssl ca -config root-ca.conf -extensions v3\_intermediate\_ca -days 3650 -notext -in ../sub-ca/csr/sub-ca.csr -out ../sub-ca/certs/sub-ca.crt





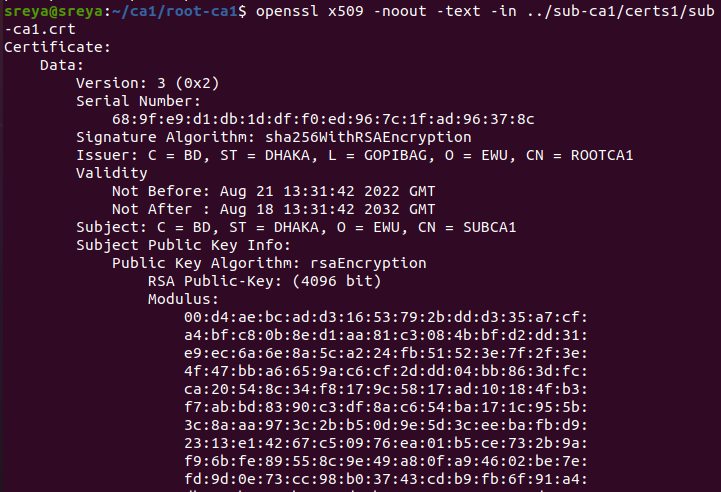
Step 15:check the certificate in .pem format





Step 16: Check sub-ca .crt:

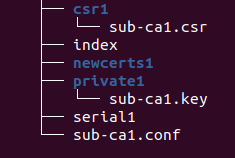
openssl x509 -noout -text -in ../sub-ca/certs/sub-ca.crt



Step 17: Check the folder using:

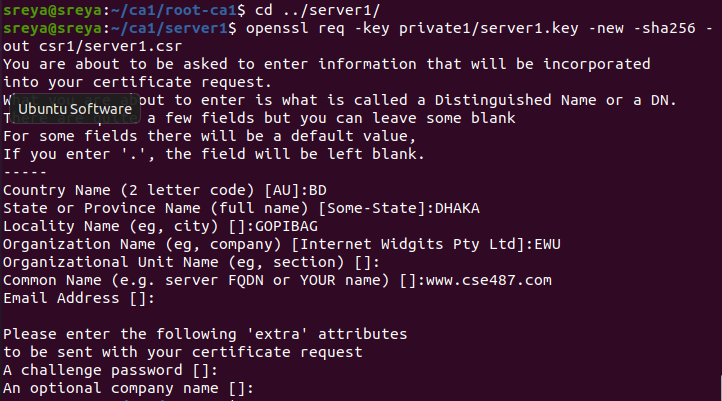
tree ca





Step 18: Server signing request

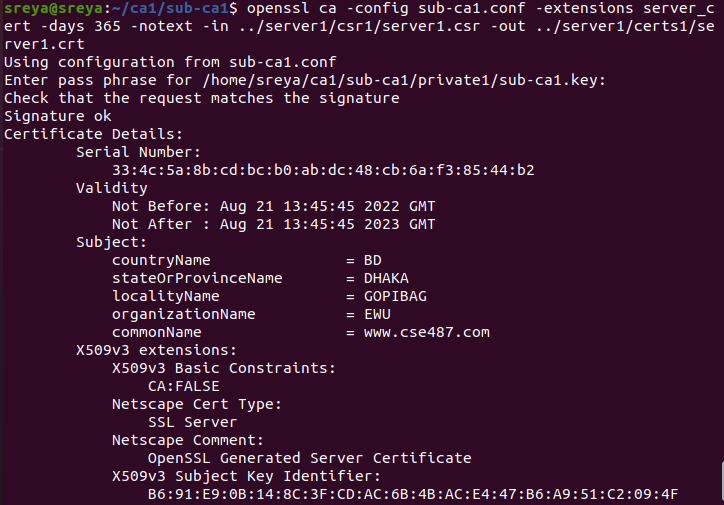
openssl req -key private/server.key -new -sha256 -out csr/server.csr

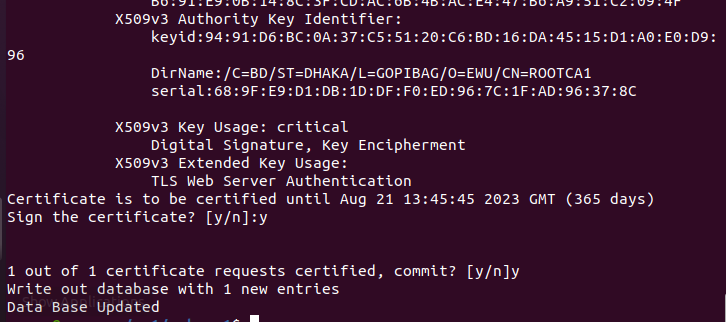




Step 19: Generating Server Certificate

openssl ca -config sub-ca.conf -extensions server\_cert -days 365 -notext -in ../server/csr/server.csr -out ../server/certs/server.crt



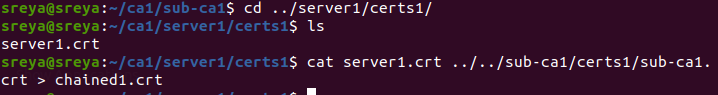


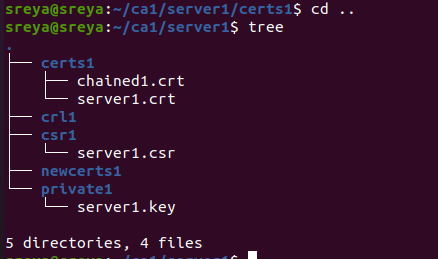




Step 20: Create chained certificate.

cat server.crt ../../sub-ca/certs/sub-ca.crt > chained.crt





Step 21: setting the ip for website in /etc/hosts

echo “192.168.0.105 acmecse487.com” >> /etc/hosts

Step 22: opening connection for certificate:

openssl s\_server -accept 443 -www -key private/server.key -cert certs/server.crt -CAfile ../sub-ca/certs/sub-ca.crt

Step 23: Nginx configuration:

To edit the file sudo su(root privilege) then nano /etc/nginx/nginx.conf and inside http insert below code.

# HTTPS server

#

server {

listen 443 ssl;

server\_name acmecse487.com;

ssl\_certificate /home/sobhani/ca/server/certs/chained.crt;

ssl\_certificate\_key /home/sobhani/ca/server/private/server.key;

ssl\_protocols TLSv1.2;

ssl\_session\_cache shared:SSL:1m;

ssl\_session\_timeout 5m;

ssl\_ciphers HIGH:!aNULL:!MD5;

ssl\_prefer\_server\_ciphers on;

location / {

root /srv/www/htdocs/;

index index.html index.htm;

}

}

include vhosts.d/\*.conf;

Step 24: In /srv/ folder make directory for htdocs:

mkdir –p www/htdocs

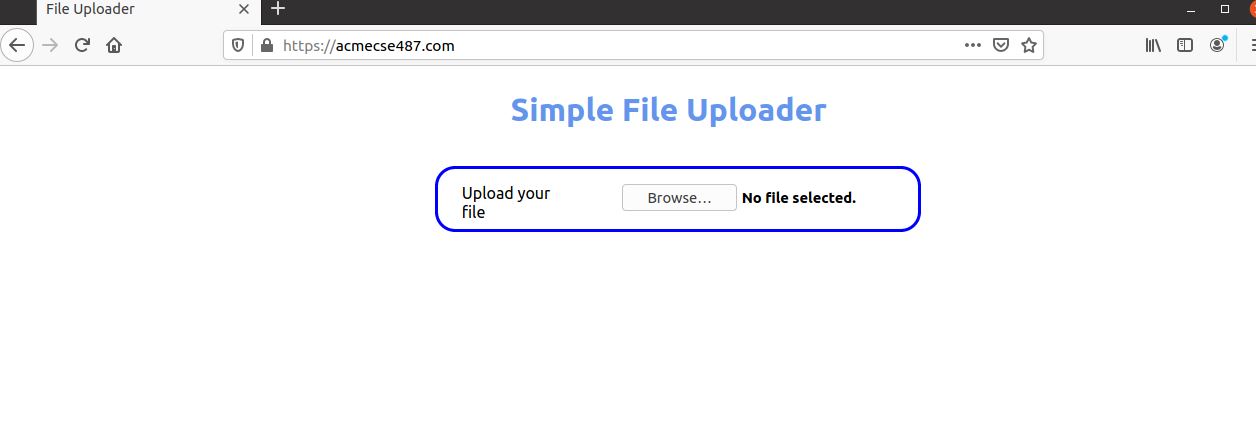
Step 25: Check website

curl https://acmecse487.com

Step 26: Import the certificate in the browser in privacy and security section

Import the ca/root-ca/newcerts/ .pem format certificate

Step 27: The padlock of https://acmecse487.com



Step 28: Certificate Revocation

First we need to move to the certificate directory.

In my case I want to revoke ca.crt

To do that import the following section of code in root-ca.conf

\*\* If you want to do it for sub-ca.crt then import the section in sub-ca.conf

\*\*Replace the command with sub-ca.conf if you want to do it for sub-ca.crt

\*\*OCSP server should be for port 80 that means http

[user\_cert]

authorityInfoAccess=OCSP;URI:http://acmecse487.com:8080

//

[v3\_OCSP]

basicConstraints= CA:FALSE

keyUsage= nonrepudiation,

digitalSignature, keyEncipherment

extendedKeyUsage= OCSPSigning

For configuration go to cd /etc/ssl

And edit the file openssl.conf

sudo gedit /etc/ssl/openssl.conf

and the change the directory to your certificate folder

In my case the folder was $dir= /home/sobhani/ca/root-ca

Change certificate directory $dir/certs/ca.crt

Change key directory $dir/private/ca.key

If you want a End user certificate then follow the next steps:

For end user key:

openssl genrsa –out certKey.key 1024

create user certificate:

openssl req –new –x509 –days 3650 –key /private/certKey.key –out certs/certificate.crt –config root-ca.conf

certificate signing request:

openssl x509 –x509toreq –in certs/certificate.crt –out CSR.csr –signkey private/certKey.key

sign the client certificate:

openssl ca –batch –startdate 150813080000Z –enddate –keyfile private/ca.key –cert certs/ca.crt –policy policy\_strict –config root-ca.conf –notext –out certs/ceretificate.crt –infile CSR.csr

Creating OCSP Server:

To host an OCSP server, an OCSP signing certificate has to be generated.

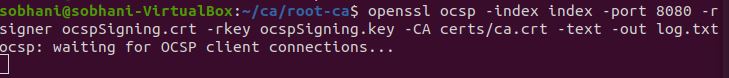
Run following 2 commands:

openssl req –new –nodes –out ocspSigning.csr –keyout ocspSigning.key

openssl ca –keyfile private/ca.key –cert certs/ca.crt –in ocspSigning.csr –out ocspSigning.crt –config root-ca.conf

To start OCSP server:

openssl ocsp -index index -port 8080 -rsigner ocspSigning.crt -rkey ocspSigning.key -CA certs/ca.crt -text -out log.txt

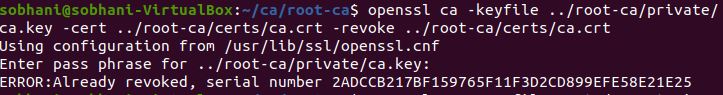


Verify certificate revocation:

openssl ocsp -CAfile certs/ca.crt -issuer certs/ca.crt -cert certs/certificate.crt -url http://acmecse487:8080 -resp\_text –noverify

For revocation:  
openssl ca -keyfile ../root-ca/private/ca.key -cert ../root-ca/certs/ca.crt -revoke ../root-ca/certs/ca.crt

You will see the server certificate and the status will be good

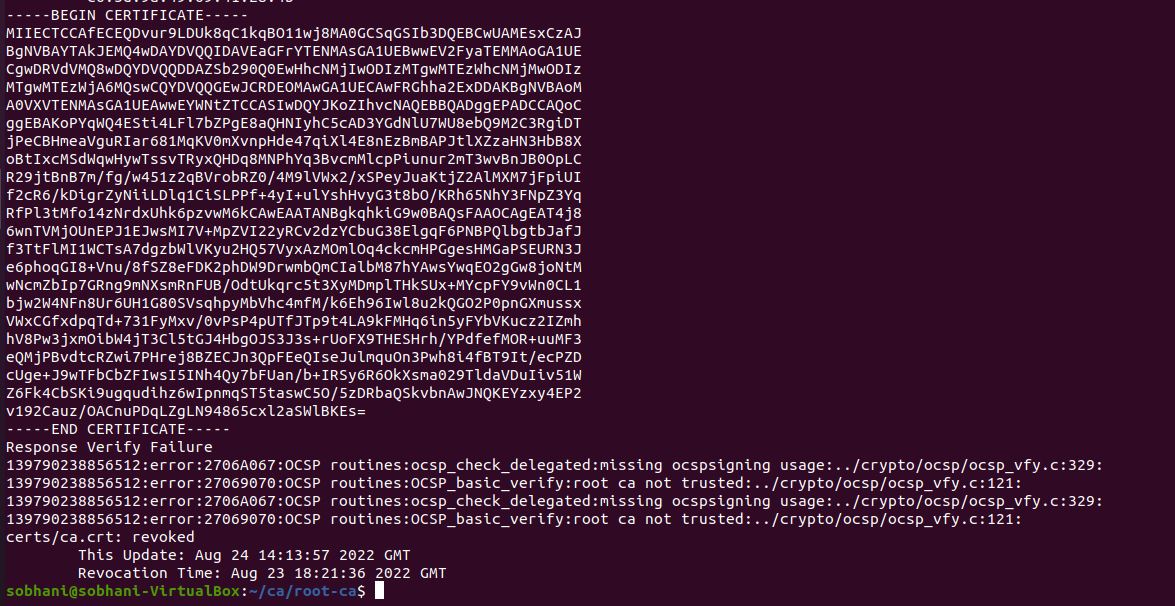
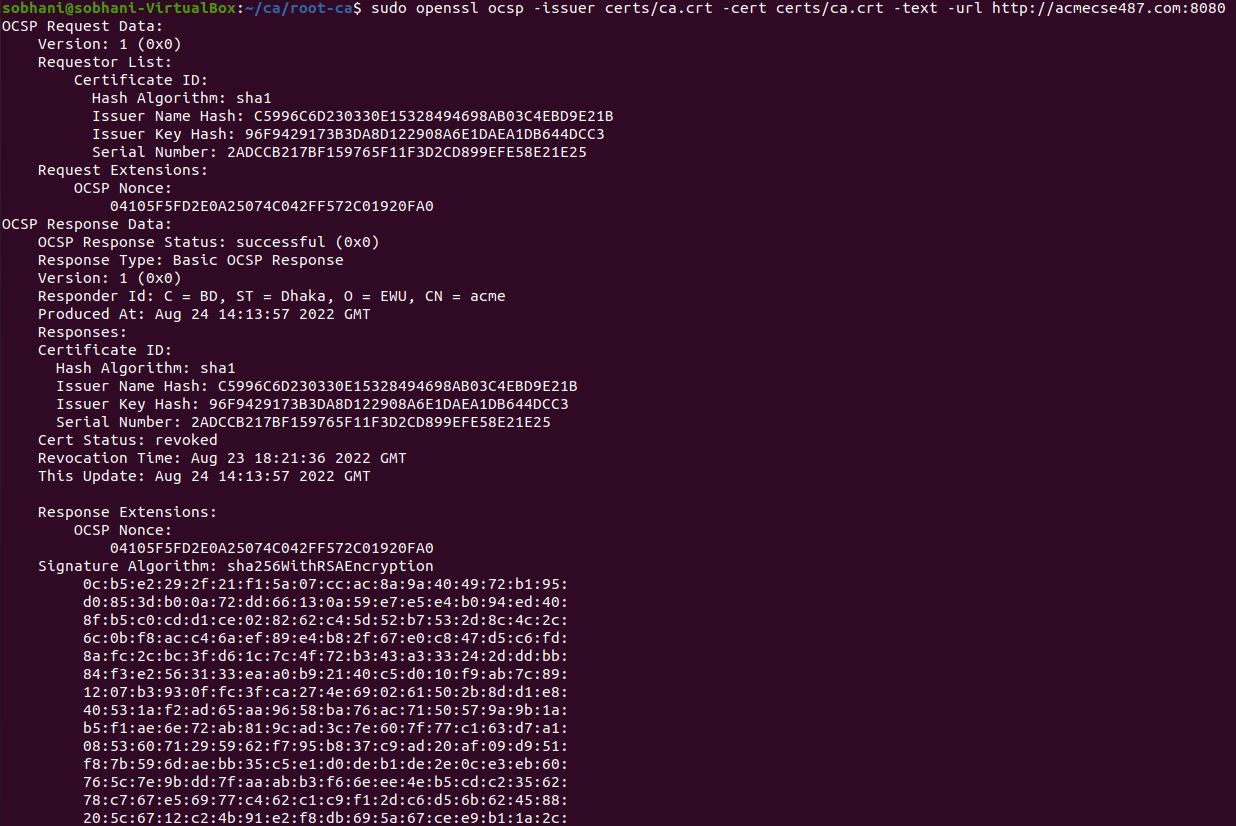


As I have already revoked the certificate that’s why it is showing error saying already revoked.

Don’t worry in your case you will be requiring pass phase and the certificate will be revoked.

To check the revoked status with server certificate:

sudo openssl ocsp -issuer certs/ca.crt -cert certs/ca.crt -text -url http://acmecse487.com:8080



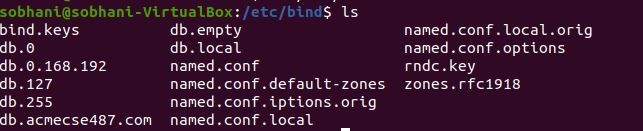
Now you can see ca.crt status is revoked.

Here verification failure saying root ca not trusted. To fix that just create ocspSigning certificate with extended key using –extension while creating certificate with ocspSigning.key.

**DNS:**

Step1: Folder location:

cd /etc/bind/



Step 2: Keep the the original files by copying into new file

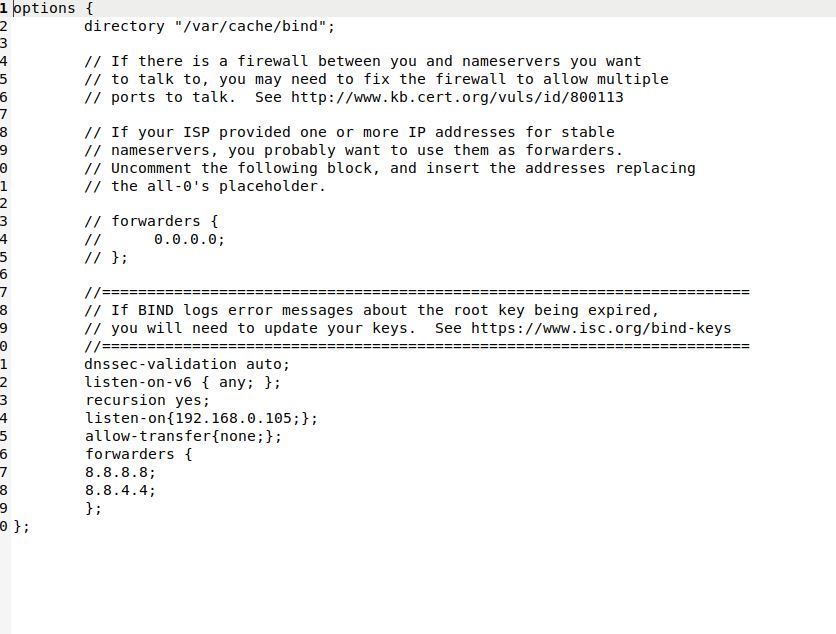
cp named.conf.options named.conf.options.orig

cp named.conf.local named.conf.local.orig

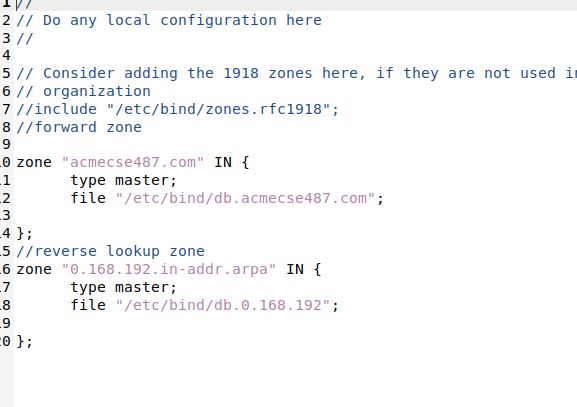
cp db.local db.acmecse487.com

cp db.127 db.0.168.192

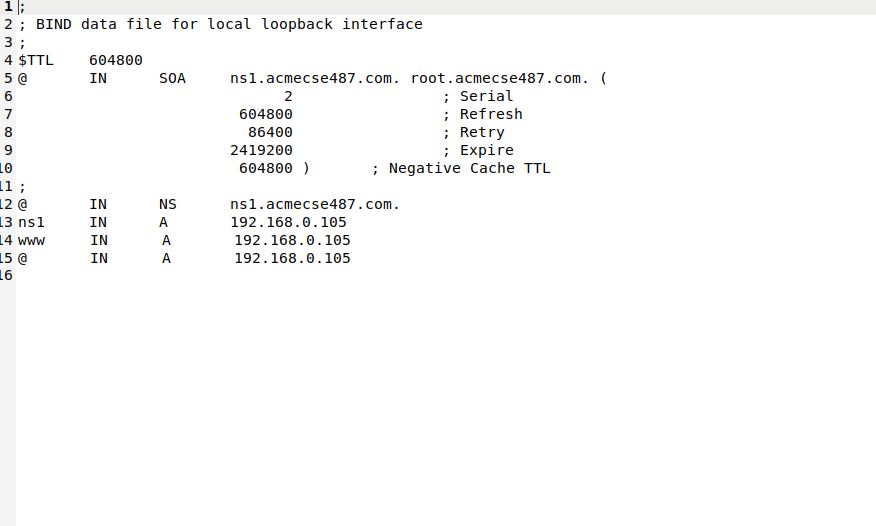
Step 3: Editing named.conf.options file



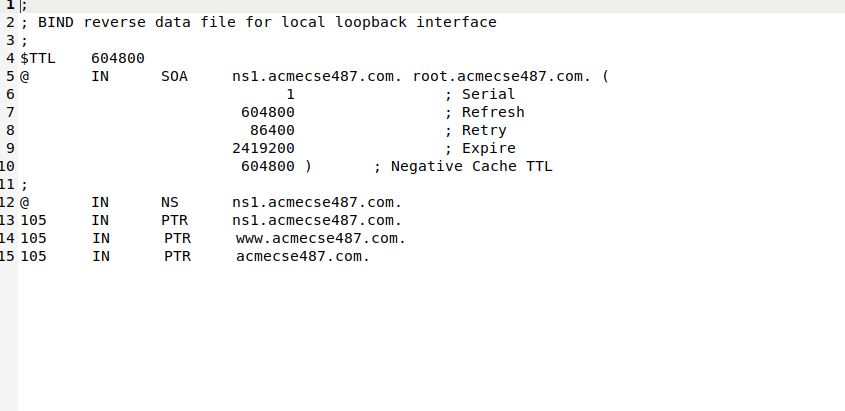
Step 4: Editing named.conf.local file



Step 5: Editing db.acmecse487.com file



Step 6: Editing db.0.168.192 file



Step 7: Check if all the configurations are ok or not:

named-checkconf

named-checkzone acmecse487.com db.acmecse487.com

named-checkzone db.0.168.192.in-addr.arpa db.0.168.192

Step 8: configure the resolv.conf file:

To do that before remove the previous resolv.conf file and create a new

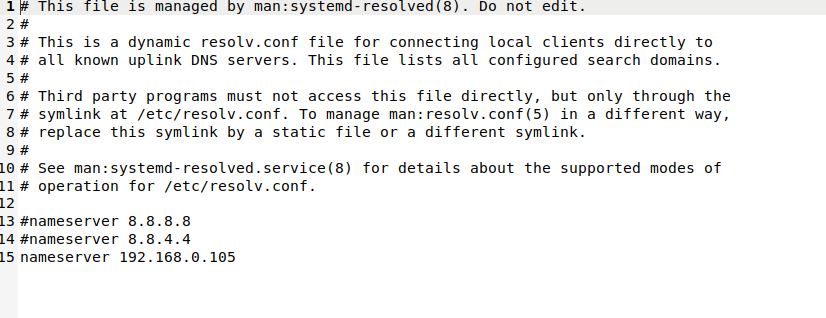
In /etc/bind/ folder

sudo rm /etc/resolv.conf

sudo ln –sf /run/system/resolve/resolv.conf /etc/resolv.conf

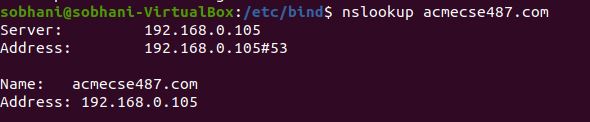
then

sudo gedit /etc/resolv.conf



Step 9: Lookup for website:

nslookup acmecse487.com



**Fire Wall:**

To start the file wall use command

sudo ufw enable

To allow the necessary ports

53: DNS Server

80: HTTP

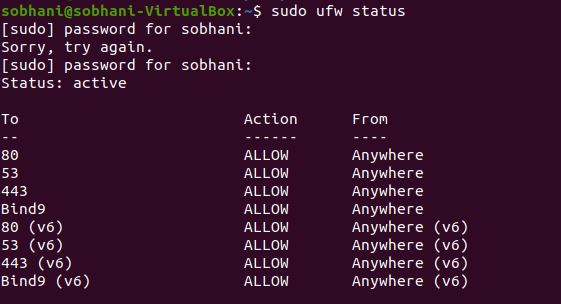
443: HTTPS

sudo ufw allow 53

sudo ufw allow 80

sudo ufw allow 443

Check Status: sudo ufw status



**IDS:** To check thedata leak in the system

Snort:

Checking if everything is ok in snort configuration

sudo snort -T -c /etc/snort/snort.conf -i enp0s3

To check the attack in the device

sudo snort -A console -q -u snort -g snort -c /etc/snort/snort.conf -i enp0s3

use nmap from different device to attack the ip

nmap 192.168.0.105