EAST WEST UNIVERSITY

CSE487

Cyber Security, Law & Ethics

Project title

Securing a networked system with Public Key Infrastructure Implementing Transport Layer Security on HTTP for https:// connection

Section: 01

Submitted to: Rashedul Amin Tuhin

Senior Lecturer

Department of Computer Science & Engineering

East West University

Submitted by: Md. Faisal Ahmed Ridoy

Id: 2019-1-60-109

Iftakhir Mozumder Nibir

Id: 2019-1-60-113

Md. Mosaddakul Islam

Id: 2019-1-60-057

In this project, we secure a networked system with Public Key Infrastructure. Here we provide our procedure step by step. We complete our project using the ubuntu virtual machine. We do these things using the command line. Firstly, we secure our web system and configure the DNS system. Also, configure the firewall for necessary ports. After that, we revoked our certificates. Here we provide the necessary screenshots, instructions, and commands that will helps others to do the project.

Set-up Environment:

Move to root

Sudo –i

Find tree inside the root

Tree

To create directory

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

Check the folder by using tree

tree ca

Changing the root ca and sub ca private folder

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

Creating file index of root ca and sub ca

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

Check the tree

tree ca

Hexadecimal random number of 16 character by the following comment

openssl rand -hex 16

Serial number of root ca

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

Serial number of sub ca

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

See the tree

tree ca

Moving to ca

cd ca

Generating private key for root ca, sub ca and server:

Public Key for rootCA

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

Public Key for SubCA

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

Public Key for server

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

See the change

```
tree
```

Certificate Generation

Root-CA

Creating root ca.config

```
vim root-ca/root-ca.conf
```

Insert the following code into thr root-ca.config

```
[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 = EWUBD
[ 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
```

Save and exit

```
:wq
```

See the tree

```
tree
```

Move to root-CA

```
Cd root-ca
```

Generate root ca certificate

```
openssl req -config root-ca.conf -key private/ca.key -new -x509 -days 7305

-sha256 -extensions v3_ca -out certs/ca.crt
```

Ensure the certificate

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

Back to sub-CA

```
cd /sub-ca
```

Sub-CA

Config Sub-ca

```
vim sub-ca.conf
```

Enter the previous code

```
[ca]

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

#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 = EWUBD

[ 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
```

Save the code

```
:wq
```

See the tree

```
tree
```

Request the sub ca certificate signing

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

Go to previous folder

cd –

Request of sub CA by root CA

openssl ca -config root-ca.conf -extensions v3_intermediate_ca -days 365 -notext -in ../sub-ca/csr/sub-ca.csr -out ../sub-ca/certs/sub-ca.crt

To confirm press

"y"

See the tree

.pem has been generated

tree

See signing

cat index

Ensure certificate

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

Configuring Server

Move to server

cd ../server

Signing request for server

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

Move to sub ca

cd /sub-ca

Signing certificate request for server

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

Moving to certs folder

cd /server/certs/

See the directory

ls

Concating sub-ca.crt and server.crt and naming the new file chained.crt

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

Back to server-directory

cd ..

echo "127.0.0.1 www.verysecureserver.com" >> /etc/hosts

ping www.verysecureserver.com

Turning on ssl port

openssl s_server -accept 443 -www -key private/server.key -cert

certs/server.crt -CAfile ../sub-ca/certs/sub-ca.crt

In new terminal

sudo –i

See the port

ss –ntl

sudo apt update

Install curl

apt install curl

Copy the certificate folder to ca certificate folder

cp ca/root-ca/certs/ca.crt /usr/local/share/ca-certificates/

Update ca certificate folder

update-ca-certificates –v

See the html file

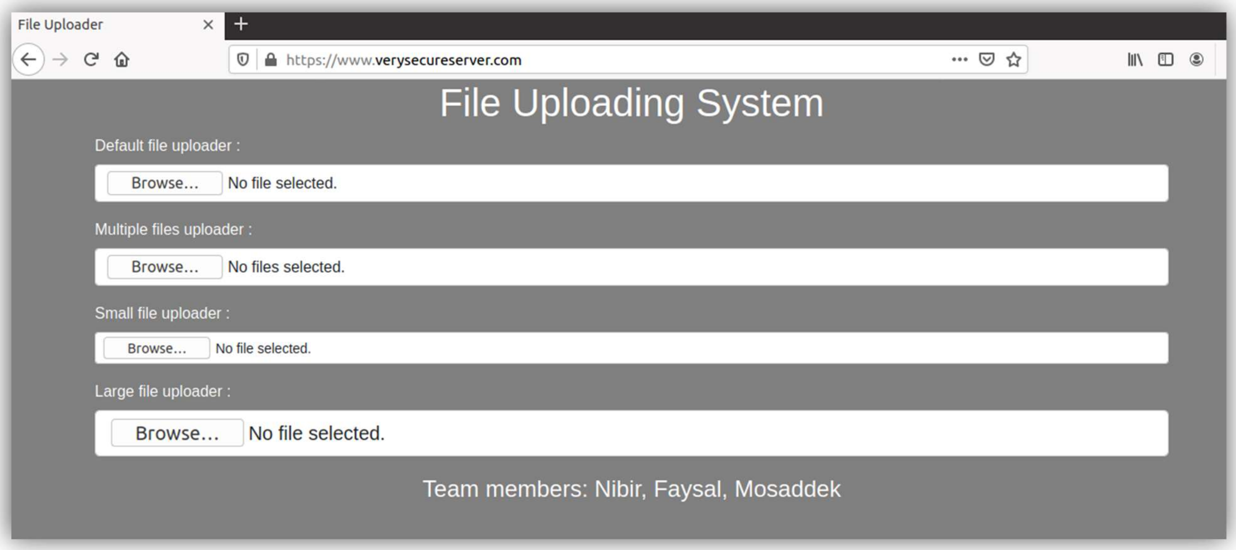
curl https://www.verysecureserver.com

After that we install apache in our Ubuntu machine and set the path of the certificate. We also edit the html page for the project.

In new terminal
Sudo –i
Copy to newcerts directory

cp /root/ca/root-ca/newcerts/.pem ~iftakhir/

After that, we got the padlock icon on our website. Showing the picture of our website is given below,



File Uploading System

Default file uploader :

Browse... No file selected.

Multiple files uploader :

Browse... No files selected.

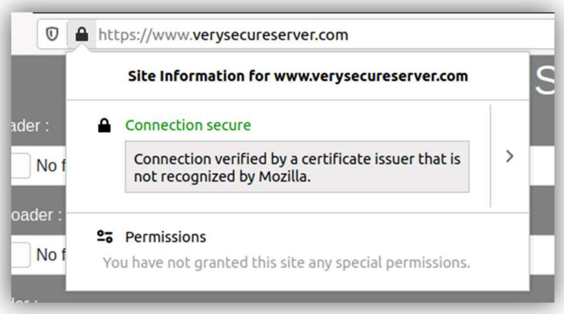
Small file uploader :

Browse... No file selected.

Large file uploader :

Browse... No file selected.

Team members: Nibir, Faysal, Mosaddek



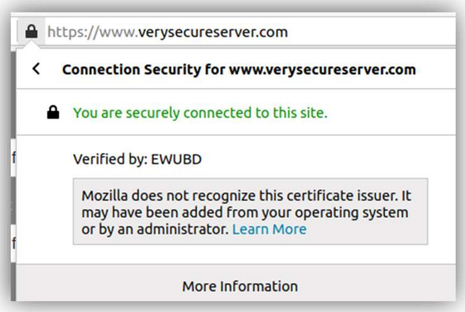
Site Information for www.verysecureserver.com

Connection secure

Connection verified by a certificate issuer that is not recognized by Mozilla.

Permissions

You have not granted this site any special permissions.



Connection Security for www.verysecureserver.com

You are securely connected to this site.

Verified by: EWUBD

Mozilla does not recognize this certificate issuer. It may have been added from your operating system or by an administrator. Learn More

More Information

Firewall Configuration

After that, we are working with firewall configuration. For our requirement, we need to allow the necessary ports (53, 80, 443) only. The configuration procedure is given below.

Install ufw package

apt install ufw

Check ufw

systemctl status ufw

Default rules for ufw firewall

ufw default allow outgoing

Enable ssh

ufw allow ssh

Again Checking the Status

ufw status

ufw enable

ufw status

Allow port 53,80,443

ufw allow 53

ufw allow 80

ufw allow 443

ufw allow http/tcp

Ufw status

After seeing the status, It showing us the allowed port. In this way, we configure the firewall.

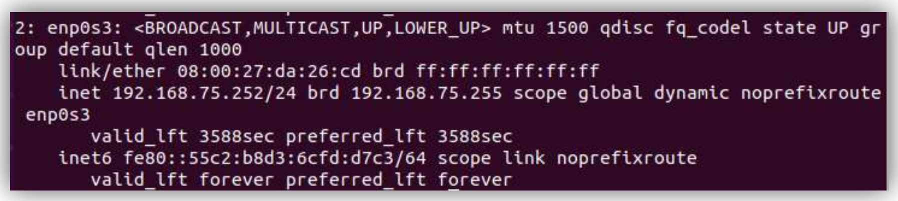
<u>DNS Configuration</u>

For DNS configuration, we consider our ubuntu machine as a server and windows machine as a client. Here we give the step-by-step procedure for configuring DNS in both client and server pc.

In server pc,

For checking the ip address in our machine,

ip addr



```
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP gr
oup default qlen 1000
    link/ether 08:00:27:da:26:cd brd ff:ff:ff:ff:ff:ff
    inet 192.168.75.252/24 brd 192.168.75.255 scope global dynamic noprefixroute
 enp0s3
       valid_lft 3588sec preferred_lft 3588sec
    inet6 fe80::55c2:b8d3:6cfd:d7c3/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

so we use ip address: 192.168.75.252

installing packages

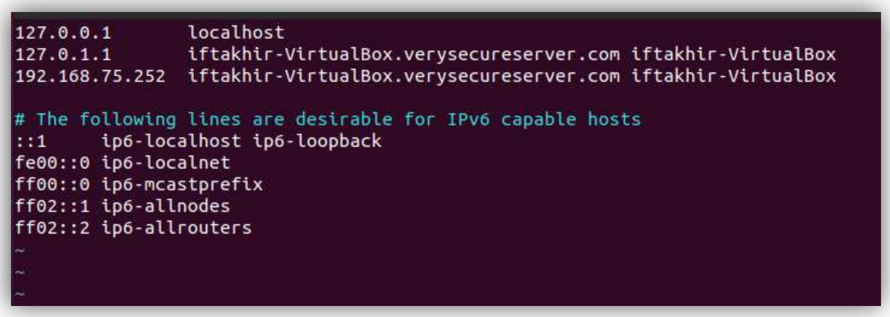
sudo apt install bind

going to the path

cd /etc/bind

assigning ip address, hostname, domainname

sudo vim /etc/hosts

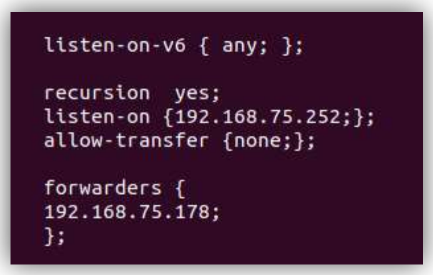


```
127.0.0.1       localhost
127.0.1.1       iftakhir-VirtualBox.verysecureserver.com iftakhir-VirtualBox
192.168.75.252  iftakhir-VirtualBox.verysecureserver.com iftakhir-VirtualBox

# 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
~
~
~
```

Editing named.conf.options

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

sudo vim named.conf.options



```
listen-on-v6 { any; };

recursion  yes;
listen-on {192.168.75.252;};
allow-transfer {none;};

forwarders {
192.168.75.178;
};
```

Changing This part. Here we assign our ip address and default gateway. Then we edit db.named.conf.local and verified it.

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

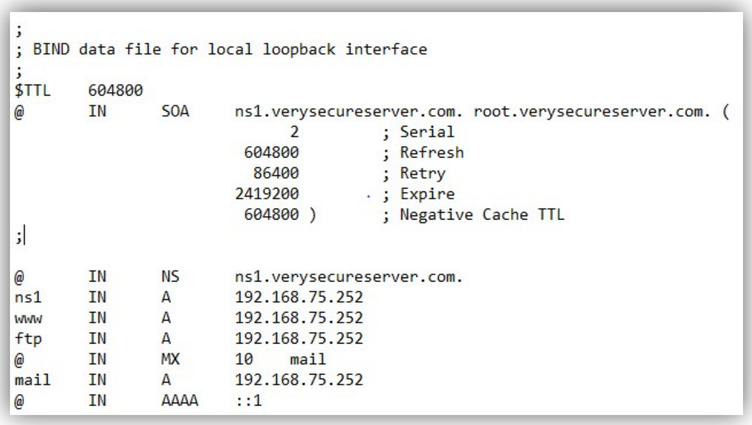
sudo gedit named.conf.local

named-checkconf

after that we edit db.local and verified it

sudo cp db.local db.verysecureserver.com

sudo gedit db.verysecureserver.com



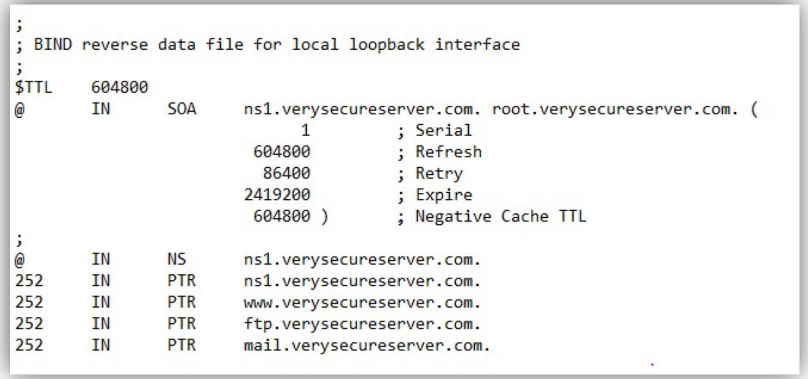
```
;
; BIND data file for local loopback interface
;
$TTL    604800
@       IN      SOA     ns1.verysecureserver.com. root.verysecureserver.com. (
                              2         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;

@       IN      NS      ns1.verysecureserver.com.
ns1     IN      A       192.168.75.252
www     IN      A       192.168.75.252
ftp     IN      A       192.168.75.252
@       IN      MX      10    mail
mail    IN      A       192.168.75.252
@       IN      AAAA    ::1
```

named-checkzone verysecureserver.com db.verysecureserver.com

After that we edit and verified reverse ip address

sudo cp db.127 db.75.168.192

sudo gedit db.75.168.192



```
;
; BIND reverse data file for local loopback interface
;
$TTL    604800
@       IN      SOA     ns1.verysecureserver.com. root.verysecureserver.com. (
                              1         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      ns1.verysecureserver.com.
252     IN      PTR     ns1.verysecureserver.com.
252     IN      PTR     www.verysecureserver.com.
252     IN      PTR     ftp.verysecureserver.com.
252     IN      PTR     mail.verysecureserver.com.
```

named-checkzone 75.168.192.in-addr.arpa db.75.168.192

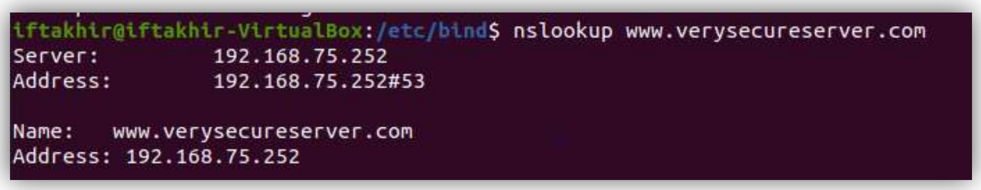
named-checkconf

Lastly we need to restart bind9 and checking the status

sudo service bind9 restart

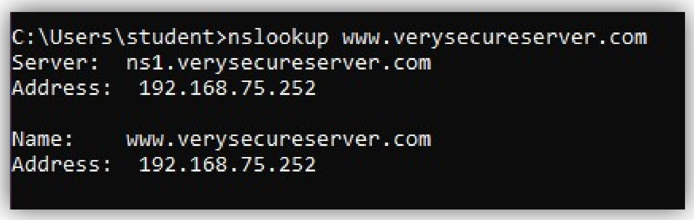
sudo service bind9 status

Finally checking nslookup command to verify



```
iftakhir@iftakhir-VirtualBox:/etc/bind$ nslookup www.verysecureserver.com
Server:         192.168.75.252
Address:        192.168.75.252#53

Name:   www.verysecureserver.com
Address: 192.168.75.252
```

Also checking this commend in the client pc



```
C:\Users\student>nslookup www.verysecureserver.com
Server:  ns1.verysecureserver.com
Address:  192.168.75.252

Name:    www.verysecureserver.com
Address:  192.168.75.252
```

In this way, we configured our DNS server.

Certificate revocation

Here, we are giving the procedure to revoke our certificate.

login as root user:
sudo -i

Revoking server certificate using openssl:
directory->**sub-ca**:
crlnumber was not created during certificate generation
touch crlnumber

directory->**ca**:

revoking certificate of server:
openssl ca -config sub-ca/sub-ca.conf -revoke server/certs/server.crt

verifying intermediate database located in index file:
cat sub-ca/index.txt

Notice the first column of first row i.e. R for Revoked. So certificate has been revoked

for generating certificate revocation list (crl), in sub-ca.conf the path of sub-ca.crl we have to define path for crl
crl = **$dir/crl/sub-ca.crl**

generating crl:
openssl ca -config sub-ca/sub-ca.conf -gencrl -out sub-ca/crl/sub-ca.crl

check the revoked certificate list in crl:
openssl crl -in sub-ca/crl/sub-ca.crl -text -noout

Verify revocation:
creating a temporary root-ca by merging the crl file with the root-ca, sub-ca certificate
cat root-ca/newcerts/<root-cert-name>.pem sub-ca/newcerts/<sub-cert-name>.pem sub-ca/crl/sub-ca.crl > /tmp/test.pem

openssl verify -extended_crl -verbose -CAfile /tmp/test.pem -crl_check server/certs/server.crt

After that the result shows that the certificate revoked successfully.