EAST WEST UNIVERSITY

Mini Project-1

Course Code: CSE487

Course Title: Cyber Security, Law and Ethics

Section: 1

Submitted By:

Name	ID
Tasnia Afrin Chowdhury	2018-2-60-133
Tasnim Mozumder Anannya	2018-2-60-032
Somaiya Akter Sania	2018-2-60-069

Submitted To:

Rashedul Amin Tuhin

Senior Lecturer

Department of Computer Science and Engineering

East West University

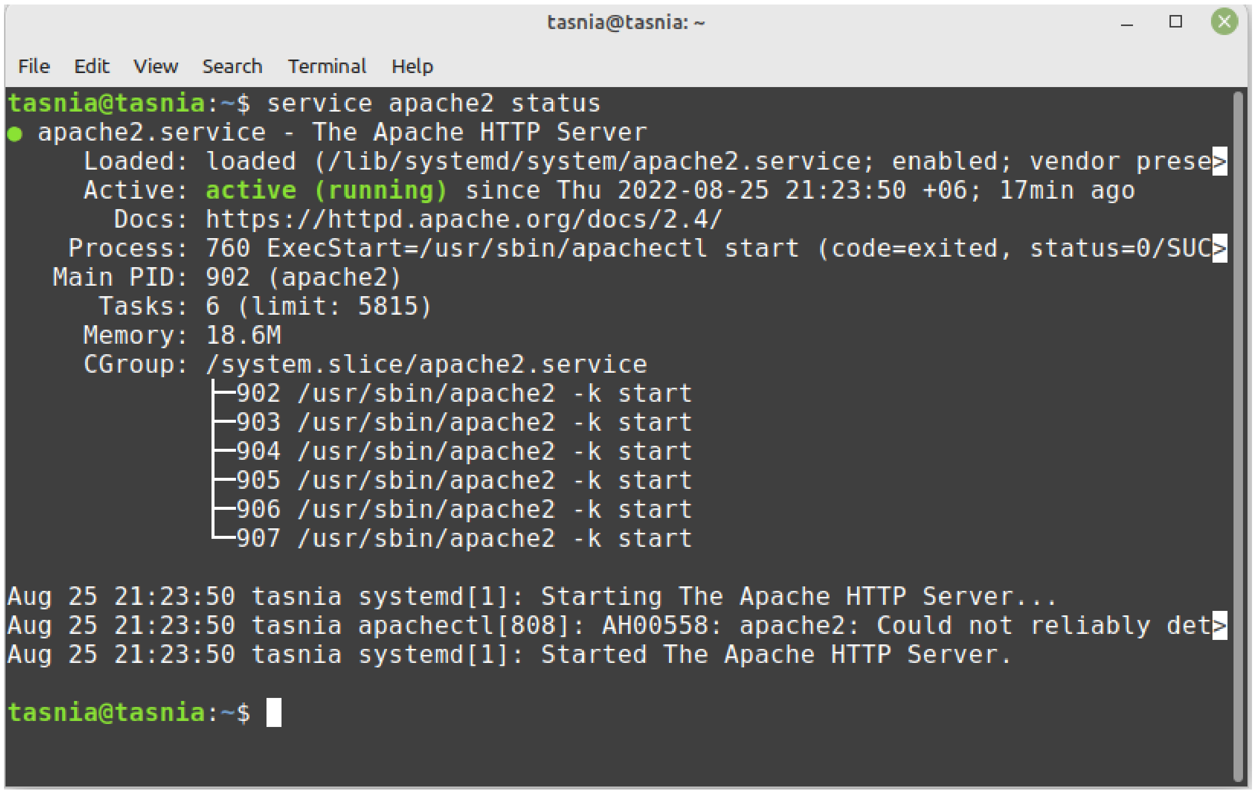
Project Title: Securing a networked system with Public Key Infrastructure

(Implementing Transport Layer Security on HTTP for https:// connection)

Description: The work of this project was done on a Linux Mint (Ubuntu) virtual machine using OpenSSL tool and bind9 and dnsutils libraries. Let us assume we have a server on the local address 127.0.0.1, which we want to turn into a website named "acmesecureserver.com" and have the site SSL encrypted and secure. For this purpose, we need to configure 2 main things, 1. A local name resolving DNS server, 2. Creating signed SSL certificates to ensure security.

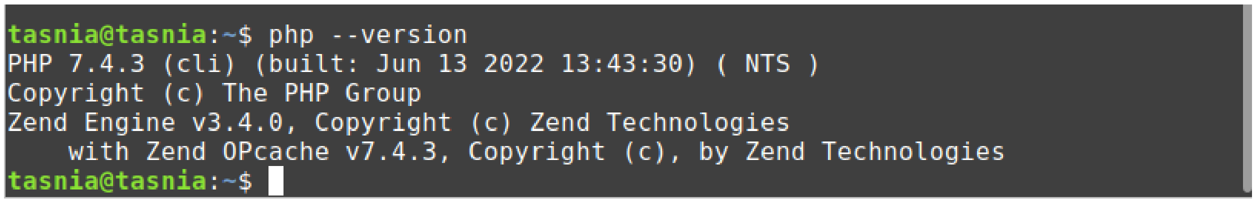
Step-1) Install Apache2 and PHP in Linux: For installing apache2 and php in linux we have to run the following commands:

```
sudo apt install apache2
service apache2 status
```

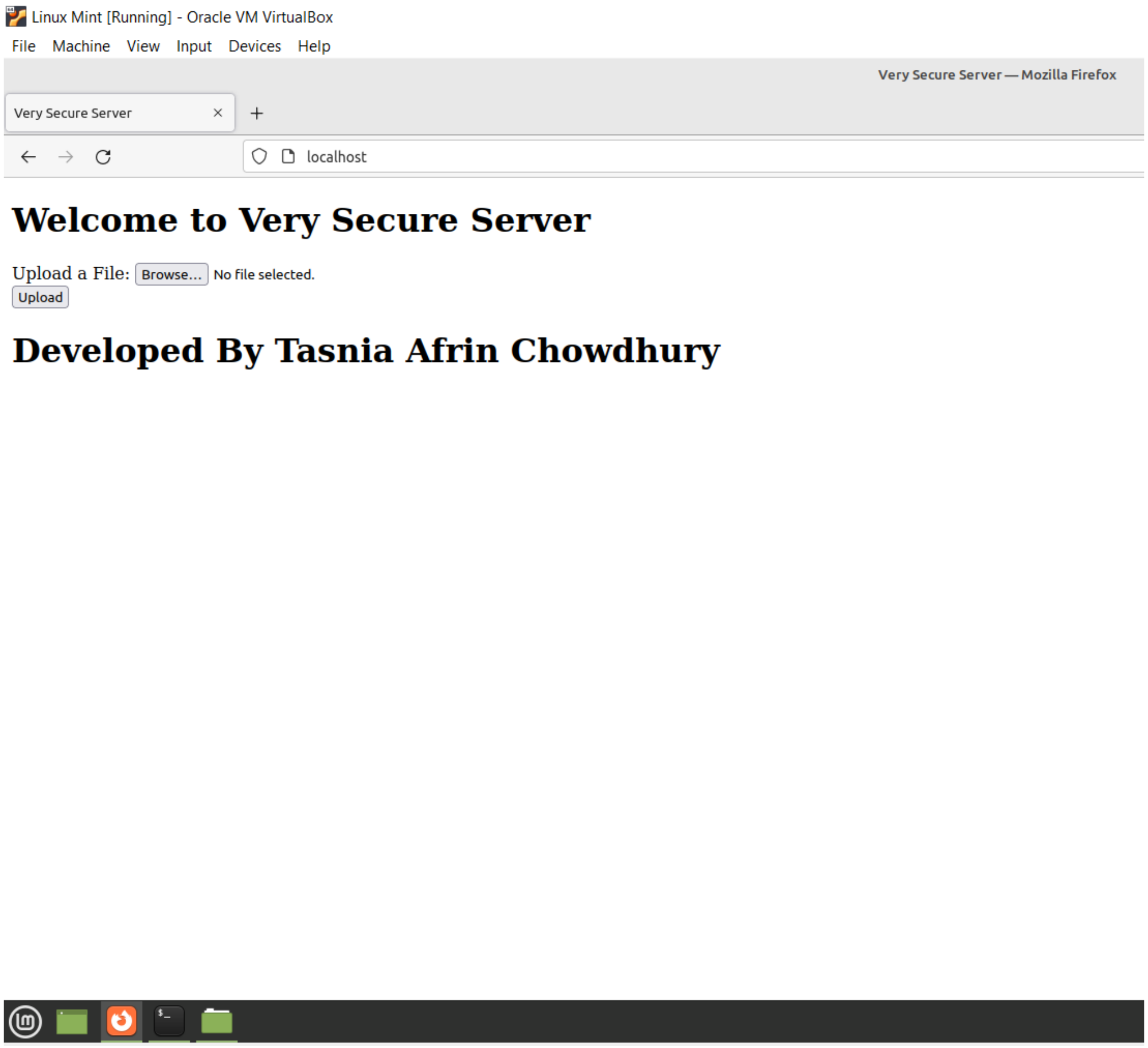


```
tasnia@tasnia:~$ service apache2 status
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor prese>
     Active: active (running) since Thu 2022-08-25 21:23:50 +06; 17min ago
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 760 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUC>
   Main PID: 902 (apache2)
      Tasks: 6 (limit: 5815)
     Memory: 18.6M
     CGroup: /system.slice/apache2.service
             ├─902 /usr/sbin/apache2 -k start
             ├─903 /usr/sbin/apache2 -k start
             ├─904 /usr/sbin/apache2 -k start
             ├─905 /usr/sbin/apache2 -k start
             ├─906 /usr/sbin/apache2 -k start
             └─907 /usr/sbin/apache2 -k start

Aug 25 21:23:50 tasnia systemd[1]: Starting The Apache HTTP Server...
Aug 25 21:23:50 tasnia apachectl[808]: AH00558: apache2: Could not reliably det>
Aug 25 21:23:50 tasnia systemd[1]: Started The Apache HTTP Server.

tasnia@tasnia:~$
```

```
sudo apt-get install php
```



```
tasnia@tasnia:~$ php --version
PHP 7.4.3 (cli) (built: Jun 13 2022 13:43:30) ( NTS )
Copyright (c) The PHP Group
Zend Engine v3.4.0, Copyright (c) Zend Technologies
    with Zend OPcache v7.4.3, Copyright (c), by Zend Technologies
tasnia@tasnia:~$
```


Linux Mint [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Very Secure Server — Mozilla Firefox
Very Secure Server
localhost

Welcome to Very Secure Server

Upload a File: Browse... No file selected.

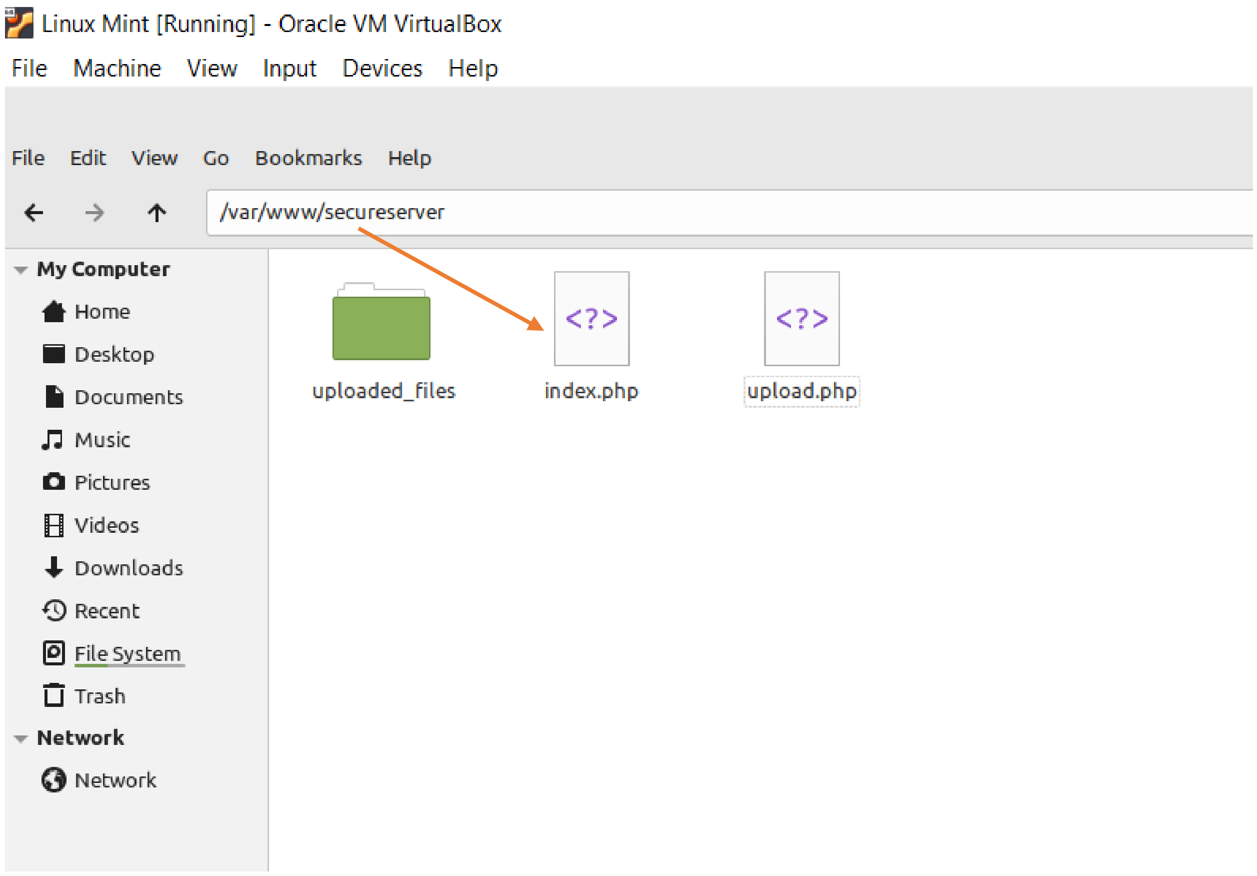
Upload

Developed By Tasnia Afrin Chowdhury

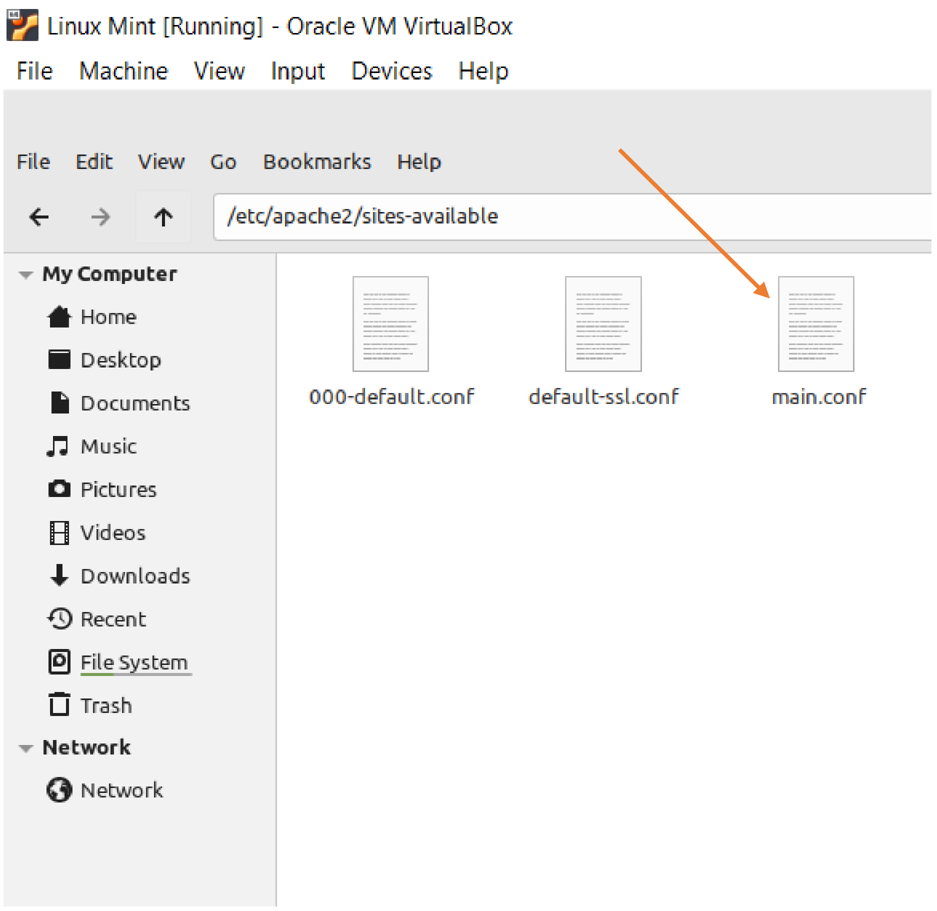
Step-2) Configure DNS Server in Linux Mint:

For this, we have to first go to the File System→ var→ www→ secureserver and then create a php file named index.php.

→



Then we have to disable the file from where our local host is running. For this again, we have to go the File System→ etc→ apache2→ sites available and create a conf file named main.conf.


Linux Mint [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
File Edit View Go Bookmarks Help
/etc/apache2/sites-available
My Computer
Home
Desktop
Documents
Music
Pictures
Videos
Downloads
Recent
File System
Trash
Network
Network
000-default.conf
default-ssl.conf
main.conf

To disable the default conf file we have to run the following commands

```
sudo a2dissite 000-default
```

Then we have to restart our server. For this we have to run the following command:

```
systemctl reload apache2
```

Then to enable the main file we have to run:

```
sudo a2ensite main
```

Then we have to reload our system control. For this we have to run:

```
systemctl reload apache2
```

Then we have to create another php file named upload.php and a folder named uploaded_files.

Then we have to run the following commands to configure the dns server:

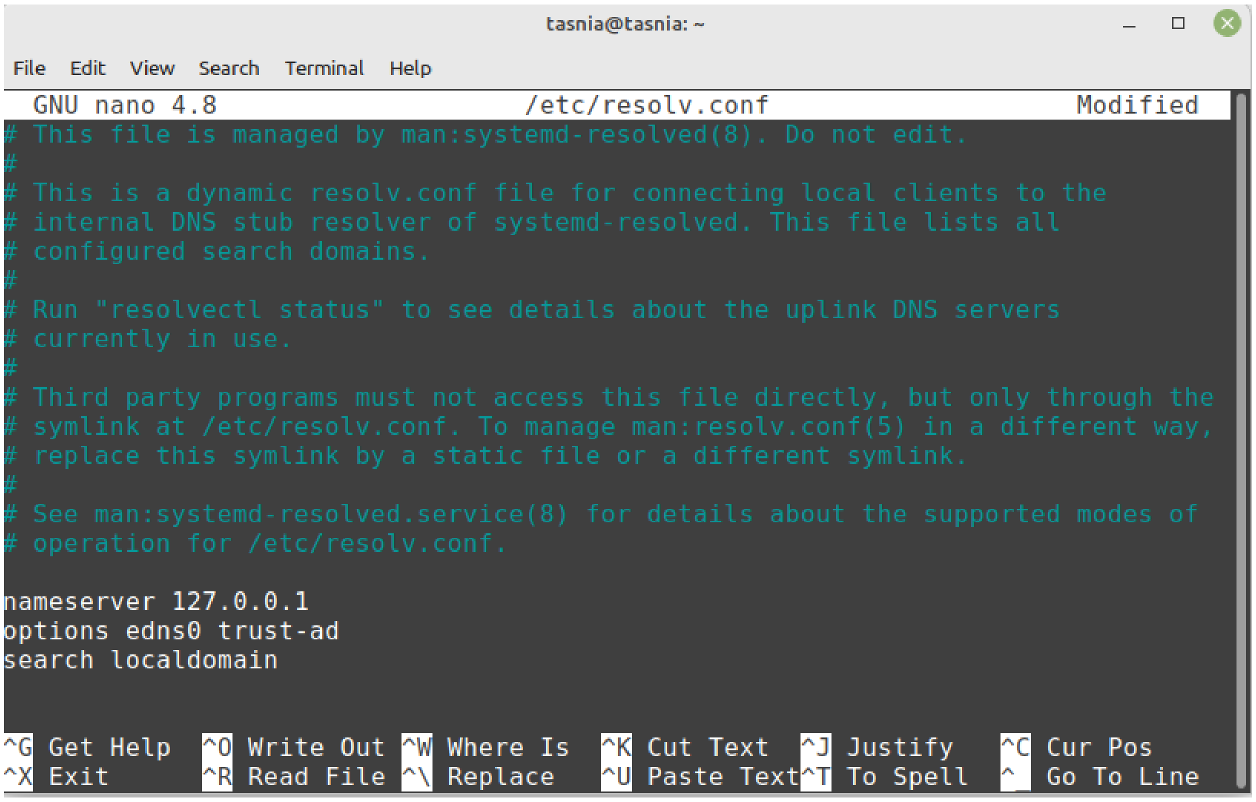
```
sudo apt install bind9
sudo apt install dnsutils
sudo systemctl restart bind9.service
```

sudo nano /etc/resolv.conf

nameserver 127.0.0.1

options edns0 trust-ad

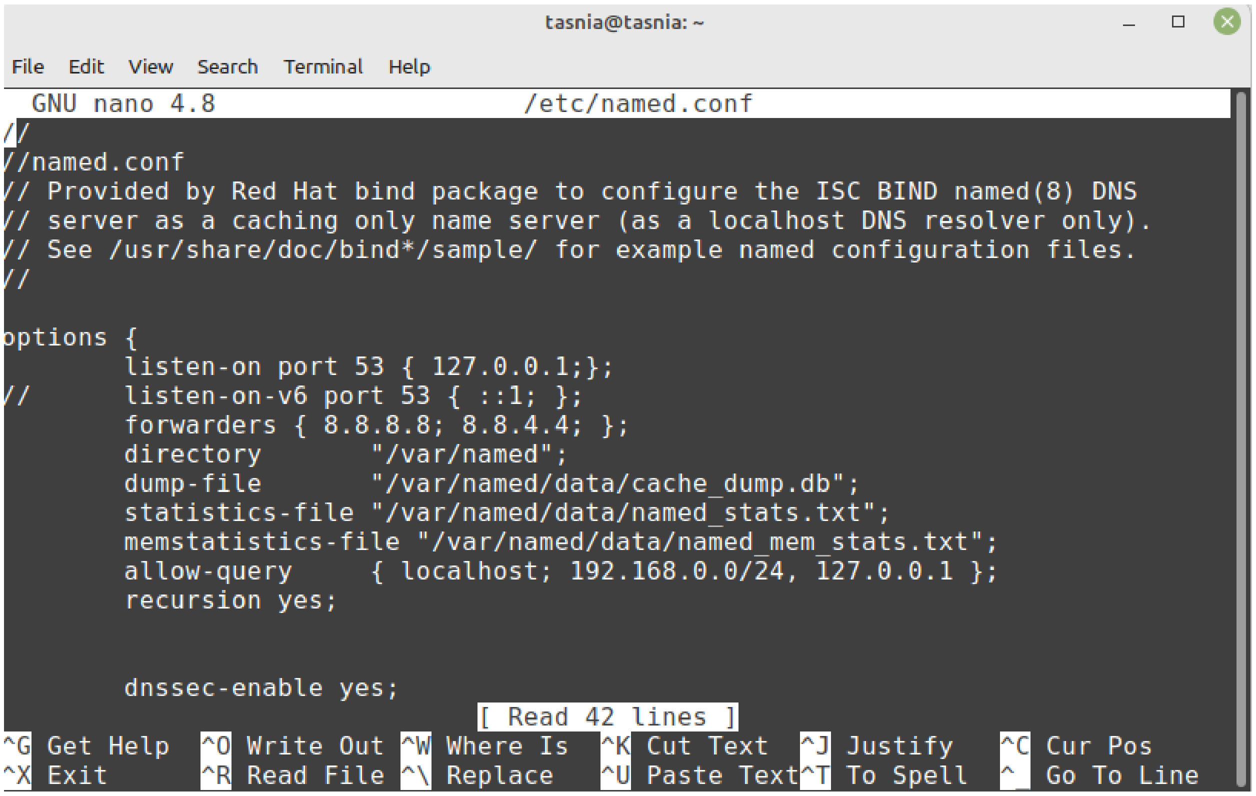
search localdomain



```
  GNU nano 4.8                    /etc/resolv.conf                    Modified
# This file is managed by man:systemd-resolved(8). Do not edit.
#
# This is a dynamic resolv.conf file for connecting local clients to the
# internal DNS stub resolver of systemd-resolved. This file lists all
# configured search domains.
#
# Run "resolvectl status" to see details about the uplink DNS servers
# currently in use.
#
# Third party programs must not access this file directly, but only through the
# symlink at /etc/resolv.conf. To manage man:resolv.conf(5) in a different way,
# replace this symlink by a static file or a different symlink.
#
# See man:systemd-resolved.service(8) for details about the supported modes of
# operation for /etc/resolv.conf.

nameserver 127.0.0.1
options edns0 trust-ad
search localdomain


^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Paste Text^T To Spell  ^_ Go To Line
```

Then we have to create a conf file named name, for this:

sudo nano /etc/named.conf

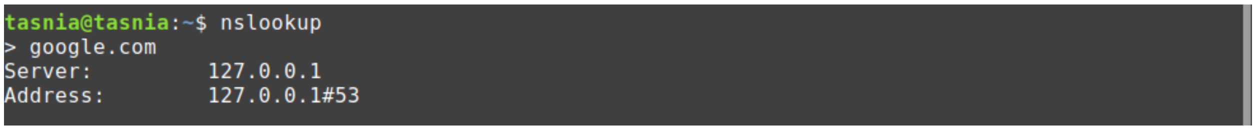


```
tasnia@tasnia: ~
File  Edit  View  Search  Terminal  Help
  GNU nano 4.8                        /etc/named.conf
//
//named.conf
// Provided by Red Hat bind package to configure the ISC BIND named(8) DNS
// server as a caching only name server (as a localhost DNS resolver only).
// See /usr/share/doc/bind*/sample/ for example named configuration files.
//

options {
        listen-on port 53 { 127.0.0.1;};
//      listen-on-v6 port 53 { ::1; };
        forwarders { 8.8.8.8; 8.8.4.4; };
        directory       "/var/named";
        dump-file       "/var/named/data/cache_dump.db";
        statistics-file "/var/named/data/named_stats.txt";
        memstatistics-file "/var/named/data/named_mem_stats.txt";
        allow-query     { localhost; 192.168.0.0/24, 127.0.0.1 };
        recursion yes;


        dnssec-enable yes;
                               [ Read 42 lines ]
^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos
^X Exit        ^R Read File   ^\ Replace     ^U Paste Text^T To Spell      ^_ Go To Line
```

Then we have to check if the dns of google is working or not, for this:

dig google.com

nslookup google.com



```
tasnia@tasnia:~$ nslookup
> google.com
Server:         127.0.0.1
Address:        127.0.0.1#53
```

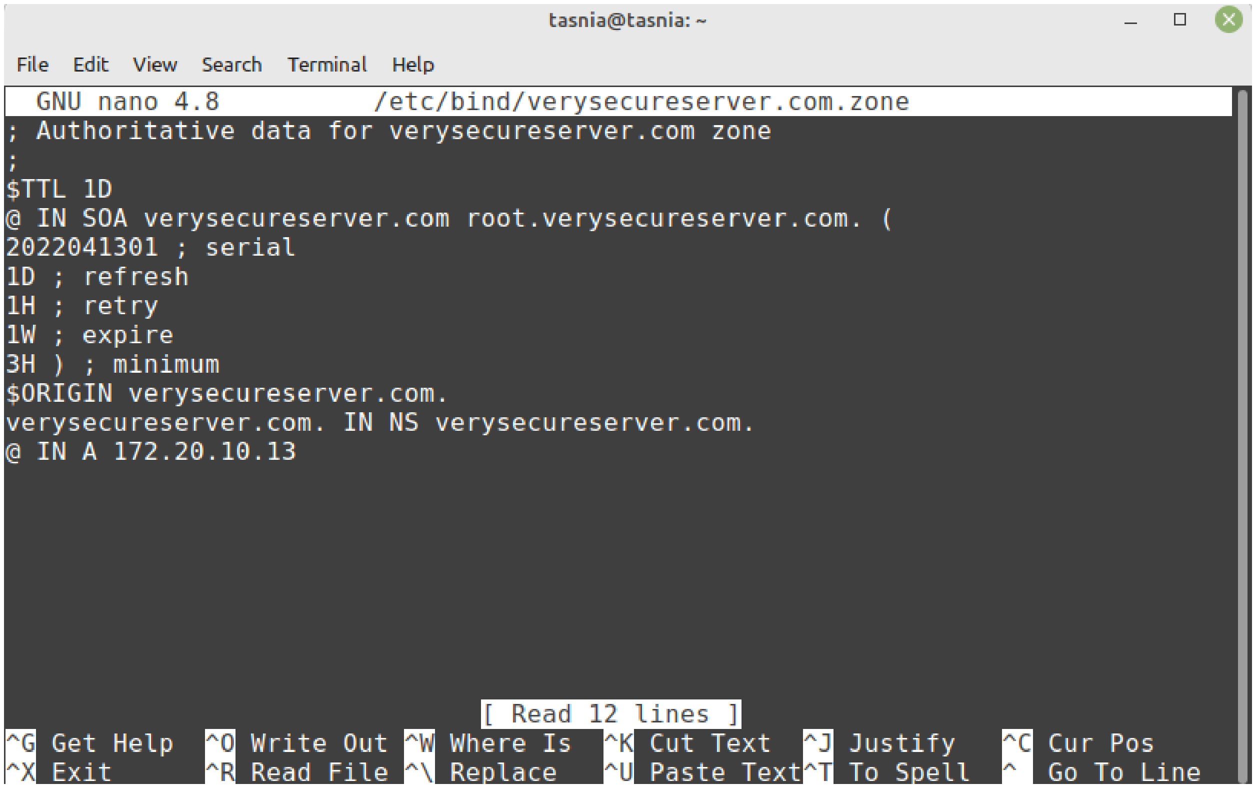
Then we have to enable the named, for this:

systemctl enable named

systemctl start named

Then we have to create a zone file for this:

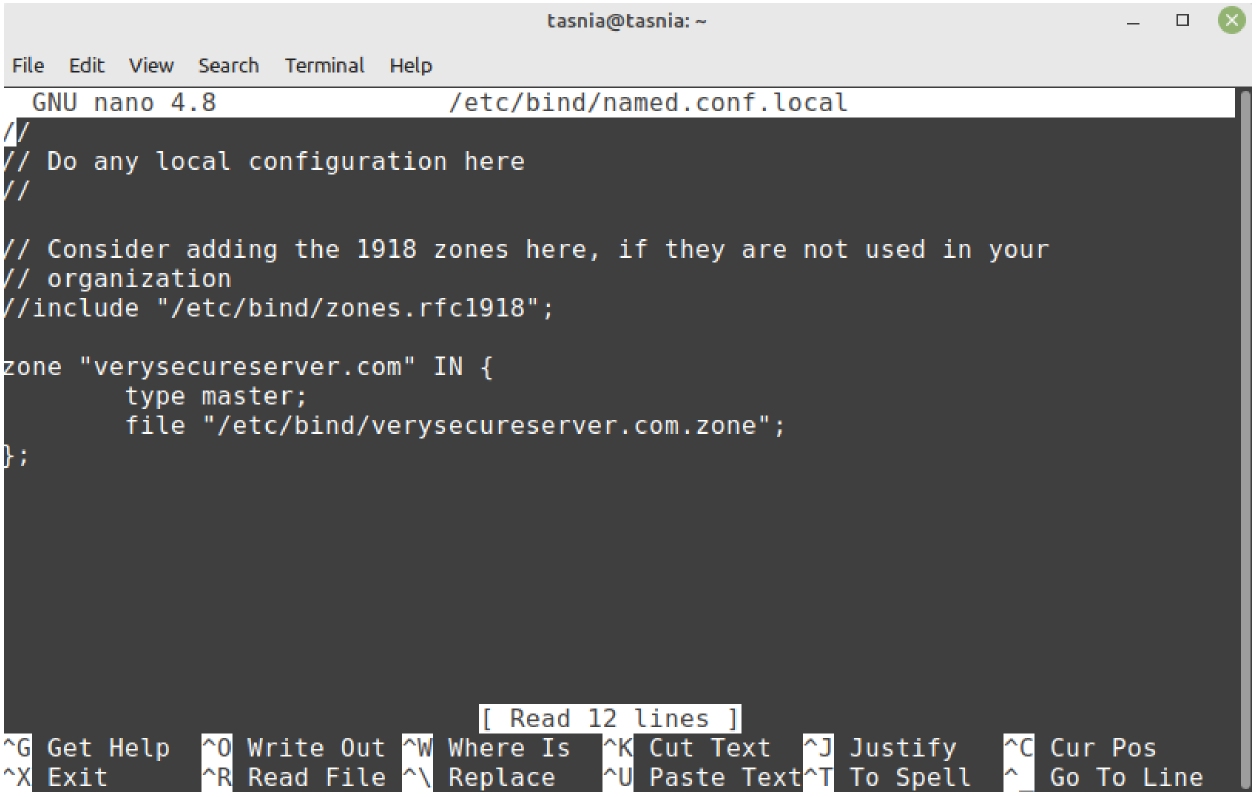
sudo nano /etc/bind/verysecureserver.com.zone



```
  GNU nano 4.8          /etc/bind/verysecureserver.com.zone
; Authoritative data for verysecureserver.com zone
;
$TTL 1D
@ IN SOA verysecureserver.com root.verysecureserver.com. (
2022041301 ; serial
1D ; refresh
1H ; retry
1W ; expire
3H ) ; minimum
$ORIGIN verysecureserver.com.
verysecureserver.com. IN NS verysecureserver.com.
@ IN A 172.20.10.13

                              [ Read 12 lines ]
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Paste Text^T To Spell  ^_ Go To Line
```

Then we have to open a local file, for this:

sudo nano /etc/bind/named.conf.local



```
  GNU nano 4.8               /etc/bind/named.conf.local
//
// Do any local configuration here
//

// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";

zone "verysecureserver.com" IN {
        type master;
        file "/etc/bind/verysecureserver.com.zone";
};

                              [ Read 12 lines ]
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Paste Text^T To Spell  ^_ Go To Line
```

Then we have to run the following commands:

```
systemctl enable named
systemctl start named
systemctl restart named
dig verysecureserver.com
nslookup verysecureserver.com
```

Create certificate ans sign this site with the certificate:

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

touch root-ca/index
touch sub-ca/index
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

cd root-ca
openssl req -config root-ca.conf -key private/ca.key -new -x509 -days 7200 -sha256 -extensions
v3_ca -out certs/ca.crt
#common name : Acme-RootCA

cd ../sub-ca/
38. openssl req -config sub-ca.conf -new -key private/sub-ca.key -sha256 -out csr/sub-ca.csr
#common name : Acme

cd ../root-ca
40. 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 -rand_serial
```

```
cd ../server
openssl req -config server.conf -key private/server.key -new -sha256 -out csr/server.csr
#common name : verysecureserver.com

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

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

Revoke certificate:

```
cd sub-ca
openssl ca -config sub-ca.conf -revoke ../server/certs/server.crt

# Add CRL to server
cd sub-ca
nano crlnumber
#type: 1002

openssl ca -config sub-ca.conf -gencrl -out crl/rev.crl
```

Appendix:

Main.conf:

```
<VirtualHost *:80>
```

```
    ServerName verysecureserver.com

    ServerAlias www.verysecureserver.com

    ServerAdmin webmaster@localhost

    DocumentRoot /var/www/secureserver

    ErrorLog ${APACHE_LOG_DIR}/error.log

    CustomLog ${APACHE_LOG_DIR}/access.log combined

</VirtualHost>

<VirtualHost *:443>

    ServerName verysecureserver.com

    ServerAlias www.verysecureserver.com

    ServerAdmin webmaster@localhost

    DocumentRoot /var/www/secureserver

    ErrorLog ${APACHE_LOG_DIR}/error.log

    CustomLog ${APACHE_LOG_DIR}/access.log combined
    SSLEngine on

        SSLCertificateFile /home/tasnia/openssl/server/certs/server.crt

        SSLCertificateKeyFile /home/tasnia/openssl/server/private/server.key

        SSLCertificateChainFile /home/tasnia/openssl/chained.crt

</VirtualHost>
```

Upload.php:

```
<?php
session_start();

$message = '';
if (isset($_POST['uploadBtn']) && $_POST['uploadBtn'] == 'Upload')
{
  if (isset($_FILES['uploadedFile']) && $_FILES['uploadedFile']['error'] === UPLOAD_ERR_OK)
  {
    // get details of the uploaded file
    $fileTmpPath = $_FILES['uploadedFile']['tmp_name'];
    $fileName = $_FILES['uploadedFile']['name'];
    $fileSize = $_FILES['uploadedFile']['size'];
    $fileType = $_FILES['uploadedFile']['type'];
    $fileNameCmps = explode(".", $fileName);
    $fileExtension = strtolower(end($fileNameCmps));

    // sanitize file-name
    $newFileName = md5(time() . $fileName) . '.' . $fileExtension;


    // check if file has one of the following extensions
    $allowedfileExtensions = array('jpg', 'jpeg','gif', 'png', 'zip', 'txt', 'xls', 'doc');


    if (in_array($fileExtension, $allowedfileExtensions))
    {
      // directory in which the uploaded file will be moved
      $uploadFileDir = './uploaded_files/';
      $dest_path = $uploadFileDir . $newFileName;


      if(move_uploaded_file($fileTmpPath, $dest_path))
      {
        $message ='File is successfully uploaded.';
      }
      else
      {
        $message = 'There was some error moving the file to upload directory. Please make sure the
upload directory is writable by web server.';
      }
    }
    else
    {
      $message = 'Upload failed. Allowed file types: ' . implode(',', $allowedfileExtensions);
    }
  }
  else
  {
    $message = 'There is some error in the file upload. Please check the following error.<br>';
    $message .= 'Error:' . $_FILES['uploadedFile']['error'];
  }
}
$_SESSION['message'] = $message;
header("Location: index.php");
```

OpenSSL root-ca.conf:

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

[CA_default]
dir     = /home/tasnia/openssl/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  = supplied
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 = Acme


[ 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
crlDistributionPoints           = @crl_dist_points

[ server_cert ]
# Extensions for server certificates
basicConstraints  = CA:FALSE
nsComment   =  "OpenSSL Generated Server Certificate"
subjectKeyIdentifier  = hash
authorityKeyIdentifier  = keyid,issuer:always
keyUsage   =  nonRepudiation, digitalSignature, keyEncipherment
extendedKeyUsage  = serverAuth
subjectAltName                                                    = @alt_names



[alt_names]
DNS.1 = verysecureserver.com
DNS.2 = www.verysecureserver.com

[crl_dist_points]
URI.0 = http://localhost:8086/rev.crl
```

OpenSSL server.conf:

```
[ca]
#C:/openssl/root-ca/root-ca.conf
```

```
#see man ca
default_ca    = CA_default

[CA_default]
dir     = /home/tasnia/openssl/server
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  = supplied
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 = Acme

[ 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
crlDistributionPoints          = @crl_dist_points
```

```
[ server_cert ]
# Extensions for server certificates
basicConstraints  = CA:FALSE
nsComment   =  "OpenSSL Generated Server Certificate"
subjectKeyIdentifier  = hash
authorityKeyIdentifier  = keyid,issuer:always
keyUsage   =  nonRepudiation, digitalSignature, keyEncipherment
extendedKeyUsage  = serverAuth
subjectAltName                                        = @alt_names


[alt_names]
DNS.1 = verysecureserver.com
DNS.2 = www.verysecureserver.com

[crl_dist_points]
URI.0 = http://localhost:8086/rev.crl
```

OpenSSL sub-ca.conf:

```
[ca]
#C:/openssl/root-ca/root-ca.conf
#see man ca
default_ca    = CA_default

[CA_default]
dir     = /home/tasnia/openssl/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
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  = supplied
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 = Acme

[ 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
crlDistributionPoints           = @crl_dist_points

[ server_cert ]
# Extensions for server certificates
basicConstraints  = CA:FALSE
nsComment   =  "OpenSSL Generated Server Certificate"
subjectKeyIdentifier  = hash
authorityKeyIdentifier  = keyid,issuer:always
keyUsage   =  nonRepudiation, digitalSignature, keyEncipherment
extendedKeyUsage  = serverAuth
```

```
subjectAltName                    = @alt_names


[alt_names]
DNS.1 = verysecureserver.com
DNS.2 = www.verysecureserver.com

[crl_dist_points]
URI.0 = http://localhost:8086/rev.crl
```

For Client Server:

We have to run the following commands:

```
sudo nano /etc/netplan/1-network-manager-all.yaml
sudo netplan try
sudo resolvectl status
Let NetworkManager manage all devices on this system
network:
version: 2
renderer: NetworkManager
ethernets:
enp0s3:
dhcp4: no
addresses: [192.168.0.103/24]
routes:
- to: default
via: 192.168.0.1
nameservers:
addresses: [192.168.0.104]
search: [verysecureserver.com]
```

192.168.0.103