EAST WEST UNIVERSITY

Department of CSE

Project Report

Course Title: Cyber Security, Law and Ethics.

Course Code: CSE487

Course Instructor: Rashedul Amin Tuhin (RDA),

Senior Lecturer, Department of Computer Science and Engineering.

Section: 1

Semester: Summer 2022

Submitted by:

Name	ID
Asif Mahmud	2018-3-60-074
MD Nazmus Sakib	2019-1-60-251
MD Mahir Bin Morshed	2019-1-60-260

Date of Submission: August 30, 2022

Configuration Specification:

VirtualBox:

Version 6.1.36 r152435 (Qt5.6.2)

Ubuntu:

Ubuntu 22.04.1 LTS

Bind9:

Version 9.18.1-1ubuntu1.1-Ubuntu (Stable Release)

Nginx:

Version 1.22.0

Openssl:

Version 3.0.2 15 Mar 2022 (Library: OpenSSL 3.0.2 15 Mar 2022)

Snort:

Version 2.9.15.1 GRE (Build 15125)

Wireshark Tool:

Version 3.6.2 (Git v3.6.2 packaged as 3.6.2-2)

Slowloris Tool:

Version 0.2.3

Server Ubuntu Machine Username:

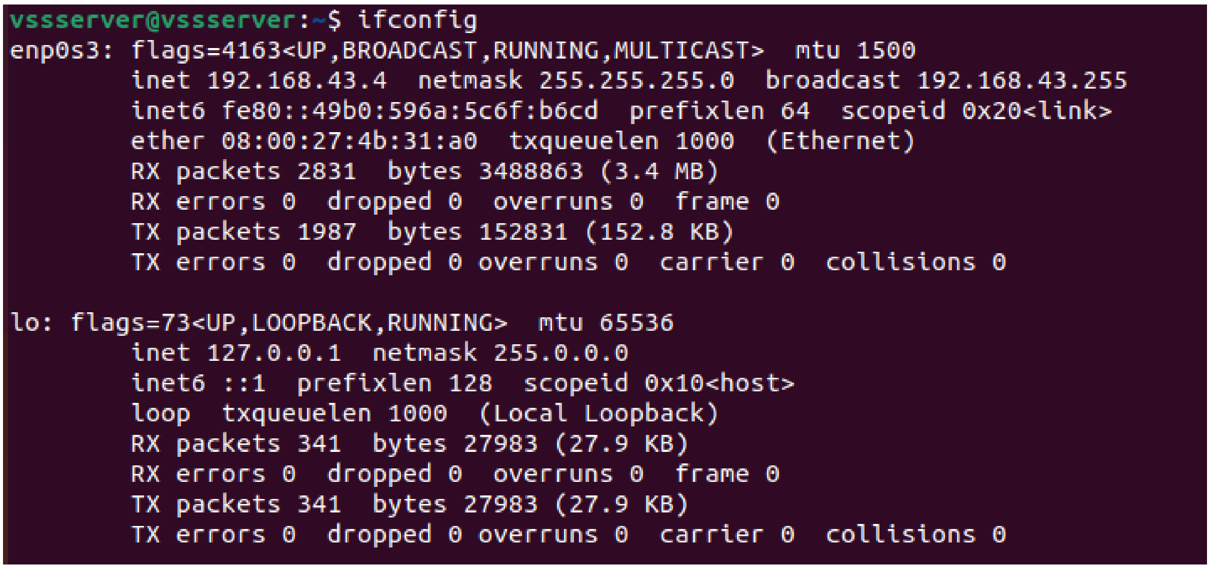
```
vssserver
```

Install Net Tools for finding IPv4 for Server Ubuntu machine:

```
sudo apt install net-tools
```

First, find the IPv4 for Server Ubuntu machine using the following command:

```
ifconfig
```



```
vssserver@vssserver:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.43.4  netmask 255.255.255.0  broadcast 192.168.43.255
        inet6 fe80::49b0:596a:5c6f:b6cd  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:4b:31:a0  txqueuelen 1000  (Ethernet)
        RX packets 2831  bytes 3488863 (3.4 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1987  bytes 152831 (152.8 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 341  bytes 27983 (27.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 341  bytes 27983 (27.9 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Here, the IPv4 is **192.168.43.4**, which will be the Server IP Address.

Configuring DNS using Bind9:

Install Bind9:

```
sudo apt install bind9
```

Set Bind9 to IPv4 mode:

```
sudo nano /etc/default/named
```

Modify the file:

```
OPTIONS="-u bind -4"
```

Configuring Bind9 Options:

```
sudo nano /etc/bind/named.conf.options
```

Modify the file:

```
options {
        recursion yes;
        allow-recursion { trusted; };
        listen-on { 192.168.43.4; };
        allow-query { 192.168.43.0/24; };
        allow-transfer { 192.168.43.0/24; };

        forwarders {
                8.8.8.8;
                8.8.4.4;
        };
}
```

Configuring Bind9 Local:

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

Modify the file:

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

zone "43.168.192.in-addr.arpa" IN {
        type master;
        file "/etc/bind/db.rev.verysecureserver.com";
};
```

Configuring forward zone for the server:

```
sudo nano /etc/bind/db.fwd.verysecureserver.com
```

Modify the file:

```
$TTL    604800
@       IN      SOA     verysecureserver.com. root.verysecureserver.com. (
                          2         ; Serial
                     604800         ; Refresh
                      86400         ; Retry
                    2419200         ; Expire
                     604800 )       ; Negative Cache TTL
;
@     IN    NS     ns.verysecureserver.com.
@     IN    A      192.168.43.4
ns      IN      A       192.168.43.4
www     IN      A       192.168.43.4
```

Configuring reverse zone for the server:

```
sudo nano /etc/bind/db.rev.verysecureserver.com
```

Modify the file:

```
$TTL    604800
@       IN      SOA     ns.verysecureserver.com. root.verysecureserver.com. (
                             1         ; Serial
                        604800         ; Refresh
                         86400         ; Retry
                       2419200         ; Expire
                        604800 )       ; Negative Cache TTL
;
@       IN      NS      ns.verysecureserver.com.
@       IN      PTR     verysecureserver.com.
ns      IN      A       192.168.43.4
4     IN      PTR     ns.verysecureserver.com.
```

Configuring resolver:

```
sudo nano /etc/resolv.conf
```

Modify the file:

```
nameserver 192.168.43.4
options edns0 trust-ad
search verysecureserver.com
```

After saving the file, set a flag that the resolver config file cannot be modified:

```
chattr +i /etc/resolv.conf
```

Configuring Primary DNS:

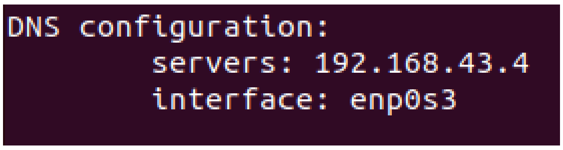
```
sudo nano /etc/systemd/resolved.conf
```

Modify the file:

```
DNS=192.168.43.4
```

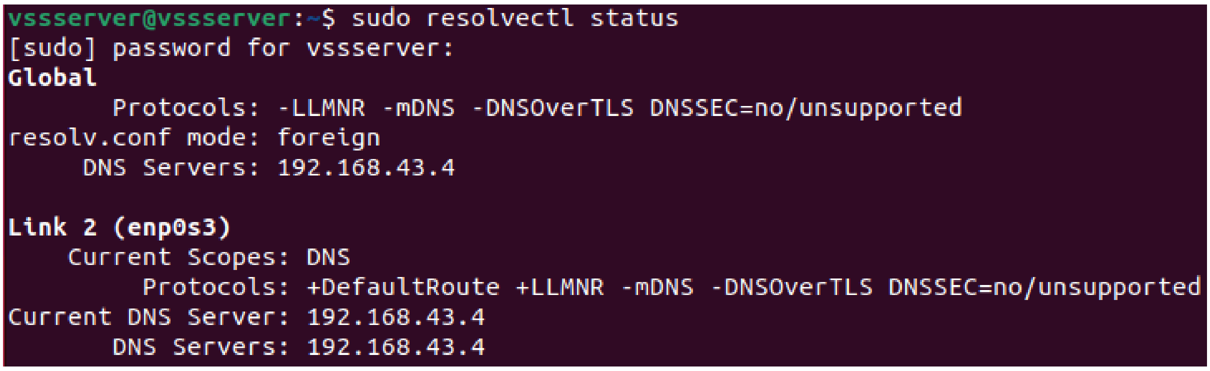
Reboot the Ubuntu machine. Then check the DNS status using the following commands:

```
nmcli
```



```
DNS configuration:
        servers: 192.168.43.4
        interface: enp0s3
```

```
sudo resolvectl status
```



```
vssserver@vssserver:~$ sudo resolvectl status
[sudo] password for vssserver:
Global
       Protocols: -LLMNR -mDNS -DNSOverTLS DNSSEC=no/unsupported
resolv.conf mode: foreign
     DNS Servers: 192.168.43.4

Link 2 (enp0s3)
    Current Scopes: DNS
         Protocols: +DefaultRoute +LLMNR -mDNS -DNSOverTLS DNSSEC=no/unsupported
Current DNS Server: 192.168.43.4
       DNS Servers: 192.168.43.4
```

Configuring the Webserver using Nginx:

Install Nginx:

```
sudo apt install nginx
```

Configuring the Webserver:

```
sudo nano /etc/nginx/sites-available/VSS
```

Modify the file:

```
server {
        listen 80;
        listen [::]:80;
        listen 443 ssl;
        listen [::]:443 ssl;
        server_name www.verysecureserver.com;

        ssl_certificate /home/vssserver/Server/pki/chained.pem;
        ssl_certificate_key /home/vssserver/Server/pki/server.key;

        ssl_protocols TLSv1.2 TLSv1.3;

        ssl_prefer_server_ciphers on;
        ssl_ciphers HIGH:!aNULL:!MD5;

        ssl_session_cache shared:SSL:10m;
        ssl_session_timeout 10m;

        ssl_ocsp on;
        ssl_crl /home/vssserver/Server/pki/sr-crl.pem;

        ssl_stapling on;
        resolver 192.168.43.4 valid=300s;
        resolver_timeout 30s;
        ssl_stapling_verify on;
        ssl_ocsp_cache shared:OCSPCache:20m;
        ssl_trusted_certificate /home/vssserver/Server/pki/root-ca.pem;

        location / {
                include proxy_params;
                root /srv/www/htdocs/;
                index index.html index.htm;
        }
}
```

Link Webserver from sites-available to sites-enabled:

```
sudo ln -s /etc/nginx/sites-available/VSS /etc/nginx/sites-enabled/VSS
```

Unlink default webserver in sites-enabled:

```
sudo unlink /etc/nginx/sites-enabled/default
```

Check the Nginx Configuration:

```
sudo nginx -t
```

Configuring Firewall:

Enable Firewall:

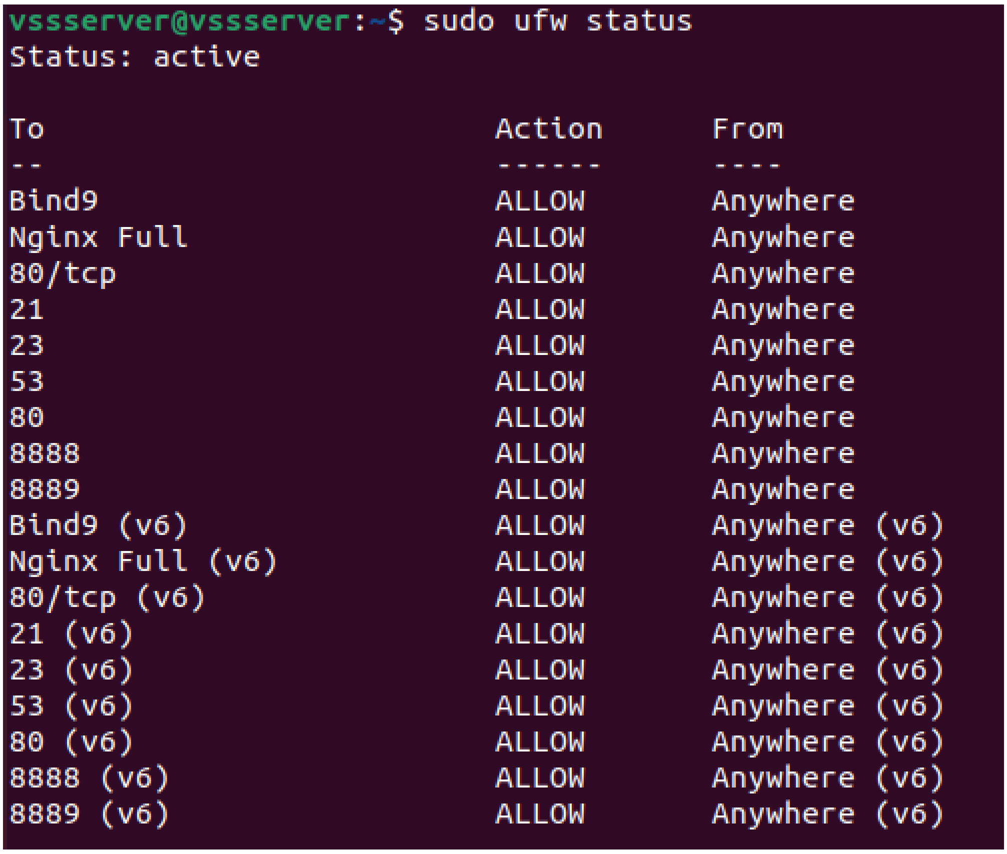
```
sudo ufw enable
```

Allow Firewall Port:

```
sudo ufw allow Bind9
sudo ufw allow 'Nginx Full'
sudo ufw allow http
sudo ufw allow 21
sudo ufw allow 23
sudo ufw allow 53
sudo ufw allow 80
sudo ufw allow 8888
sudo ufw allow 8889
```

Check Firewall Status:

```
sudo ufw status
```



```
vssserver@vssserver:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
Bind9                      ALLOW       Anywhere
Nginx Full                 ALLOW       Anywhere
80/tcp                     ALLOW       Anywhere
21                         ALLOW       Anywhere
23                         ALLOW       Anywhere
53                         ALLOW       Anywhere
80                         ALLOW       Anywhere
8888                       ALLOW       Anywhere
8889                       ALLOW       Anywhere
Bind9 (v6)                 ALLOW       Anywhere (v6)
Nginx Full (v6)            ALLOW       Anywhere (v6)
80/tcp (v6)                ALLOW       Anywhere (v6)
21 (v6)                    ALLOW       Anywhere (v6)
23 (v6)                    ALLOW       Anywhere (v6)
53 (v6)                    ALLOW       Anywhere (v6)
80 (v6)                    ALLOW       Anywhere (v6)
8888 (v6)                  ALLOW       Anywhere (v6)
8889 (v6)                  ALLOW       Anywhere (v6)
```

Generating Certificates:

Open the terminal and log in as the root user:

```
sudo su
```

Create all the necessary directories:

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

Give read, write, and execute permission to only the root user:

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

Create an index file for root-ca and sub-ca:

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

Generate a serial file for root-ca and sub-ca:

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

Generate private keys for root-ca, sub-ca, and server:

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

Create a root-ca configuration file:

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

Modify the file:

```
[ ca ]
# /root/ca/root-ca/root-ca.conf
# 'man ca'
# Used by the ca command
default_ca    = CA_default

[ CA_default ]
# Directory and file locations
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
# RANDFILE is for storing seed data for random number generation

# Root CA certificate and key locations
certificate       = $dir/certs/root-ca.crt
private_key         = $dir/private/root-ca.key

crlnumber          = $dir/crlnumber
crl             = $dir/crl/root-ca.crl
crl_extensions      = crl_ext
default_crl_days     = 30

# Default message digest, we'll opt for SHA2 256bits
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

[ req ]
# 'man req'
# Used by the req command
default_bits        = 2048
distinguished_name      = req_distinguished_name
string_mask         = utf8only
default_md          = sha256

# Extensions to use for -x509
x509_extensions         = server_cert

[ req_distinguished_name ]
# Certificate signing request
countryName             = Country Name (2 letter code)
stateOrProvinceName     = State or Province Name
localityName            = Locality Name
organizationName        = Organization Name
organizationalUnitName  = Organizational Unit Name
commonName              = Common Name
emailAddress            = Email Address

# Defaults
countryName_default             = BD
stateOrProvinceName_default     = Dhaka
organizationName_default        = Very Secure Server
commonName_default              = RootCA

[ crl_ext ]
# CRL extensions.
# Only issuerAltName and authorityKeyIdentifier make any sense in a CRL.

# issuerAltName=issuer:copy
authorityKeyIdentifier=keyid:always

[ v3_ca ]
# ' man x509v3_config'
# Extensions for root CA
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          = URI:http://www.verysecureserver.com/pki/root-ca.crl
authorityInfoAccess     = @ocsp_info

[ ocsp_info ]
caIssuers;URI = http://www.verysecureserver.com/pki/root-ca.crt
OCSP;URI = http://www.verysecureserver.com:8888

[ usr_cert ]
# `man x509v3_config`
# Extensions for client certificates
basicConstraints        = CA:FALSE
nsCertType              = client, email
nsComment               = "OpenSSL Generated Client Certificate"
subjectKeyIdentifier    = hash
authorityKeyIdentifier  = keyid,issuer
keyUsage                = critical, nonRepudiation, digitalSignature, keyEncipherment
extendedKeyUsage        = clientAuth, emailProtection

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

[ ocsp ]
# Extension for OCSP signing certificates (`man ocsp`).
basicConstraints = CA:FALSE
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid,issuer
keyUsage = critical, nonRepudiation, digitalSignature, keyEncipherment
extendedKeyUsage = critical, OCSPSigning
```

Change to the root-ca directory and generate a root-ca certificate with the root-ca configuration file and root-ca key file:

```
cd ca/root-ca

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

Check root-ca certificate:

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

Change to the sub-ca directory and create a sub-ca configuration file:

```
cd ../sub-ca

nano sub-ca.conf
```

Modify the file:

```
[ ca ]
# /root/ca/sub-ca/sub-ca.conf
# 'man ca'
# Used by the ca command
default_ca      = CA_default

[ CA_default ]
# Directory and file locations
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
# RANDFILE is for storing seed data for random number generation

# Root CA certificate and key locations
certificate             = $dir/certs/sub-ca.crt
private_key             = $dir/private/sub-ca.key

crlnumber               = $dir/crlnumber
crl                     = $dir/crl/sub-ca.crl
crl_extensions          = crl_ext
default_crl_days        = 30

# Default message digest, we'll opt for SHA2 256bits
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

[ req ]
# 'man req'
# Used by the req command
default_bits            = 2048
distinguished_name      = req_distinguished_name
string_mask             = utf8only
default_md              = sha256

# Extensions to use for -x509
x509_extensions         = server_cert

[ req_distinguished_name ]
# Certificate signing request
countryName             = Country Name (2 letter code)
stateOrProvinceName     = State or Province Name
```

```
localityName            = Locality Name
organizationName        = Organization Name
organizationalUnitName  = Organizational Unit Name
commonName              = Common Name
emailAddress            = Email Address

# Defaults
countryName_default             = BD
stateOrProvinceName_default     = Dhaka
organizationName_default        = Very Secure Server
commonName_default              = SubCA

[ crl_ext ]
# CRL extensions.
# Only issuerAltName and authorityKeyIdentifier make any sense in a CRL.

# issuerAltName=issuer:copy
authorityKeyIdentifier = keyid:always

[ v3_ca ]
# ' man x509v3_config'
# Extensions for root CA
subjectKeyIdentifier    = hash
authorityKeyIdentifier  = keyid:always,issuer
basicConstraints        = critical, CA:TRUE
keyUsage                = critical, digitalSignature, cRLSign, keyCertSign

[ usr_cert ]
# `man x509v3_config`
# Extensions for client certificates
basicConstraints        = CA:FALSE
nsCertType              = client, email
nsComment               = "OpenSSL Generated Client Certificate"
subjectKeyIdentifier    = hash
authorityKeyIdentifier  = keyid,issuer
keyUsage                = critical, nonRepudiation, digitalSignature, keyEncipherment
extendedKeyUsage        = clientAuth, emailProtection

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

[ ocsp ]
# Extension for OCSP signing certificates (`man ocsp`).
basicConstraints = CA:FALSE
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid,issuer
keyUsage = critical, nonRepudiation, digitalSignature, keyEncipherment
extendedKeyUsage = critical, OCSPSigning
```

Generate sub-ca Certificate Signing Request (CSR) using sub-ca configuration file and sub-ca key file:

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

Change to the root-ca directory and accept the sub-ca Certificate Signing Request (CSR) to generate the sub-ca certificate using the root-ca configuration file:

```
cd ../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
```

Check root-ca certificate:

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

Check root-ca index file:

```
cat index
```

Verify sub-ca certificate:

```
openssl verify -CAfile /root/ca/root-ca/certs/root-ca.crt /root/ca/sub-ca/certs/sub-ca.crt
```

Change to the server directory and create the server configuration file:

```
cd ../server

nano server.conf
```

Modify the file:

```
[ req ]
# 'man req'
# Used by the req command
default_bits            = 2048
distinguished_name      = req_distinguished_name
req_extensions          = req_ext
prompt                  = no

[ req_distinguished_name ]
# Certificate signing request
countryName             = BD
stateOrProvinceName     = Dhaka
organizationName        = Very Secure Server
commonName              = www.verysecureserver.com

[ req_ext ]
subjectAltName          = @alt_names
crlDistributionPoints          = URI:http://www.verysecureserver.com/pki/sub-ca.crl
authorityInfoAccess     = @ocsp_info

[ ocsp_info ]
caIssuers;URI = http://www.verysecureserver.com/pki/sub-ca.crt
OCSP;URI = http://www.verysecureserver.com:8889

[ alt_names ]
DNS.1   = verysecureserver.com
DNS.2   = www.verysecureserver.com
IP.1    = 192.168.43.4
```

Generate server Certificate Signing Request (CSR) using server configuration file and server key file:

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

Change to the sub-ca directory and accept the server Certificate Signing Request (CSR) to generate the server certificate using the root-ca and server configuration file:

```
cd ../sub-ca

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

Check server certificate:

```
openssl x509 -noout -text -in ../server/certs/server.crt
```

Check sub-ca index file:

```
cat index
```

Create a CRL number for root-ca and sub-ca:

```
echo 00 > /root/ca/root-ca/crlnumber
echo 00 > /root/ca/sub-ca/crlnumber
```

Generate a Certificate Revocation List (CRL) for root-ca and sub-ca:

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

Copy our root-ca certificate to the server system’s ca-certificate directory:

```
sudo cp /home/vssserver/Server/pki/root-ca.crt /usr/local/share/ca-certificates
```

Update the server system’s ca-certificate:

```
sudo update-ca-certificates -v
```

Generate root-ca certificate, sub-ca certificate, and server certificate in PEM format:

```
openssl x509 -in root-ca.crt -outform PEM -out root-ca.pem
openssl x509 -in sub-ca.crt -outform PEM -out sub-ca.pem
openssl x509 -in server.crt -outform PEM -out server.pem
```

Generate chained certificates in PEM format of the server certificate and the sub-ca certificate:

```
cat server.pem sub-ca.pem > chained.pem
```

Generate a Certificate Revocation List (CRL) in PEM format for root-ca and sub-ca:

```
openssl crl -in sub-ca.crl -outform PEM -out sub-ca-crl.pem
openssl crl -in root-ca.crl -outform PEM -out root-ca-crl.pem
```

Generate root-ca and sub-ca Certificate Revocation List (CRL) in PEM format:

```
cat sub-ca-crl.pem root-ca-crl.pem > sr-crl.pem
```

Finally, import the root-ca certificate file into the Firefox browser:

Go to Firefox browser’s Settings → Privacy & Security → View certificates → Authorities section → Click Import.

Import the root-ca certificate file and give permission to trust this CA to identify websites.

Online Certificate Status Protocol (OCSP) Configuration:

Change to the root-ca directory and create OCSP directory:

```
cd ca/root-ca
mkdir ocsp
```

Generate root OCSP Certificate Signing Request (CSR) and root OCSP key file using the root-ca configuration file:

```
openssl req -config root-ca.conf -extensions ocsp -new -nodes -out ocsp/root_ocsp.csr -keyout
ocsp/root_ocsp.key
```

Accept the root OCSP Certificate Signing Request (CSR) to generate the root OCSP certificate using the root-ca configuration file:

```
openssl ca -config root-ca.conf -extensions ocsp -in ocsp/root_ocsp.csr -out ocsp/root_ocsp.crt
```

Check root OCSP certificate:

```
openssl x509 -noout -text -in ocsp/root_ocsp.crt
```

Check root-ca index file:

```
cat index
```

Change to the sub-ca directory and create OCSP directory:

```
cd ../sub-ca
mkdir ocsp
```

Generate sub OCSP Certificate Signing Request (CSR) and sub OCSP key file using the sub-ca configuration file:

```
openssl req -config sub-ca.conf -extensions ocsp -new -nodes -out ocsp/sub_ocsp.csr -keyout ocsp/sub_ocsp.key
```

Accept the sub OCSP Certificate Signing Request (CSR) to generate the sub OCSP certificate using the sub-ca configuration file:

```
openssl ca -config sub-ca.conf -extensions ocsp -in ocsp/sub_ocsp.csr -out ocsp/sub_ocsp.crt
```

Check sub OCSP certificate:

```
openssl x509 -noout -text -in ocsp/sub_ocsp.crt
```

Check sub-ca index file:

```
cat index
```

Revoke server certificate:

Change to the sub-ca directory:

```
cd ca/sub-ca
```

Revoke server certificate using the sub-ca configuration file:

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

Check sub-ca index file:

```
cat /root/ca/sub-ca/index
```

Generate a Certificate Revocation List (CRL) for sub-ca:

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

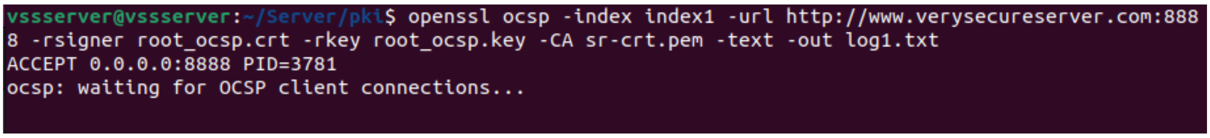
Check sub CRL:

```
openssl crl -in /root/ca/sub-ca/crl/sub-ca.crl -text
```

Start OCSP Responder:

Copy the root OCSP certificate, root OCSP key, and root-ca index file into the same dictionary. Then rename the root-ca index file to index1 and start OCSP Responder:

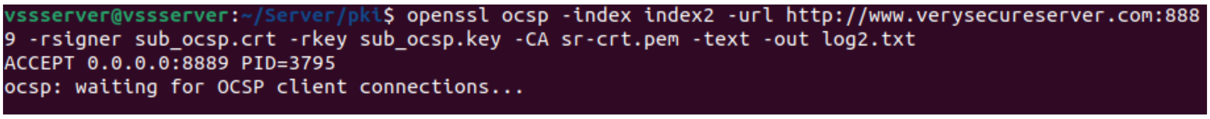
```
openssl ocsp -index index1 -url http://www.verysecureserver.com:8888 -rsigner root_ocsp.crt -rkey
root_ocsp.key -CA sr-crt.pem -text -out log1.txt
```



```
vssserver@vssserver:~/Server/pki$ openssl ocsp -index index1 -url http://www.verysecureserver.com:888
8 -rsigner root_ocsp.crt -rkey root_ocsp.key -CA sr-crt.pem -text -out log1.txt
ACCEPT 0.0.0.0:8888 PID=3781
ocsp: waiting for OCSP client connections...
```

Copy the sub OCSP certificate, sub OCSP key, and sub-ca index file into the same dictionary. Then rename the sub-ca index file to index2 and start OCSP Responder:

```
openssl ocsp -index index2 -url http://www.verysecureserver.com:8889 -rsigner sub_ocsp.crt -rkey
sub_ocsp.key -CA sr-crt.pem -text -out log2.txt
```



```
vssserver@vssserver:~/Server/pki$ openssl ocsp -index index2 -url http://www.verysecureserver.com:888
9 -rsigner sub_ocsp.crt -rkey sub_ocsp.key -CA sr-crt.pem -text -out log2.txt
ACCEPT 0.0.0.0:8889 PID=3795
ocsp: waiting for OCSP client connections...
```

Configuring IDS using Snort:

Find IPv4 in CIDR Notation:

```
ip address

192.168.43.0/24
```

Install Snort:

```
sudo apt install snort
```

Configuring Snort:

```
sudo nano /etc/snort/snort.conf
```

Modify the file:

```
ipvar HOME_NET 192.168.43.0/24
```

Apply Configuration of Snort:

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

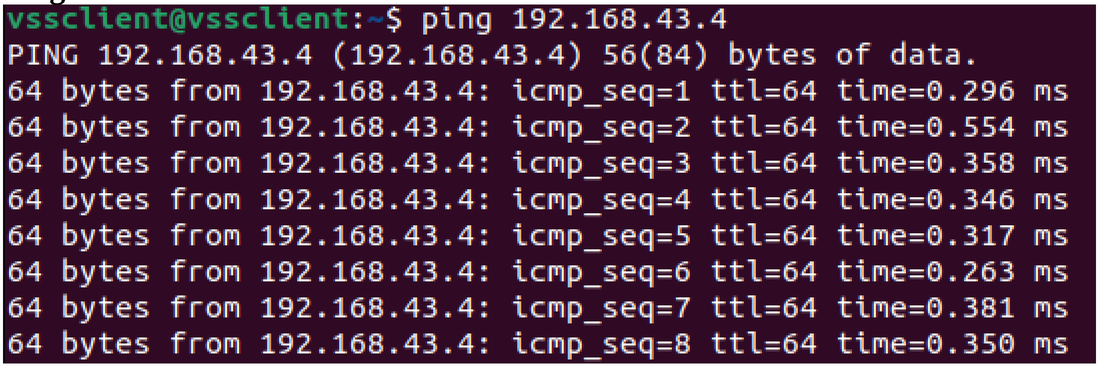
Set rules for Snort:

```
sudo nano /etc/snort/rules/local.rules
```

Modify the file:
For Pinging Alert:

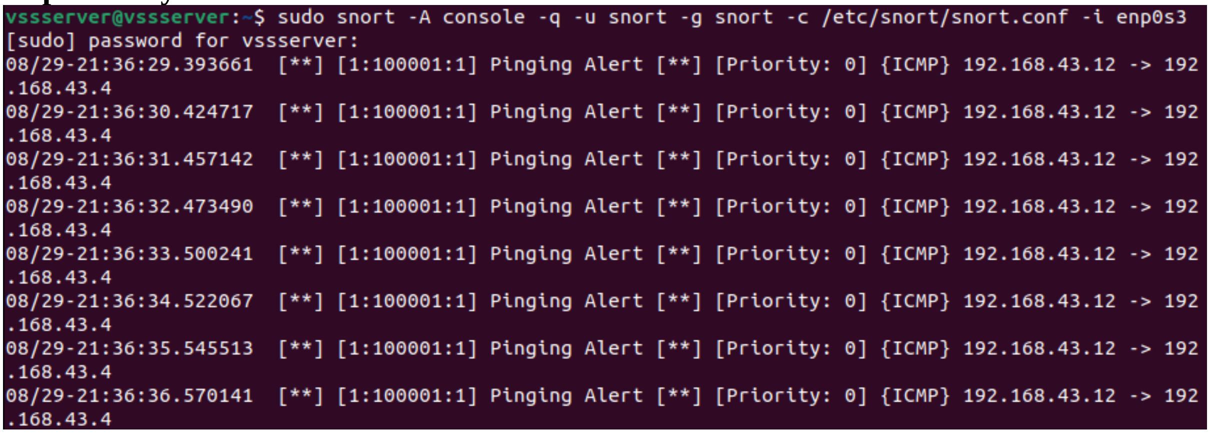
```
alert icmp any any -> 192.168.43.4 any (msg:"Pinging Alert"; sid:100001; rev:1;)
```

Ping from client:



```
vssclient@vssclient:~$ ping 192.168.43.4
PING 192.168.43.4 (192.168.43.4) 56(84) bytes of data.
64 bytes from 192.168.43.4: icmp_seq=1 ttl=64 time=0.296 ms
64 bytes from 192.168.43.4: icmp_seq=2 ttl=64 time=0.554 ms
64 bytes from 192.168.43.4: icmp_seq=3 ttl=64 time=0.358 ms
64 bytes from 192.168.43.4: icmp_seq=4 ttl=64 time=0.346 ms
64 bytes from 192.168.43.4: icmp_seq=5 ttl=64 time=0.317 ms
64 bytes from 192.168.43.4: icmp_seq=6 ttl=64 time=0.263 ms
64 bytes from 192.168.43.4: icmp_seq=7 ttl=64 time=0.381 ms
64 bytes from 192.168.43.4: icmp_seq=8 ttl=64 time=0.350 ms
```

Captured by Snort:

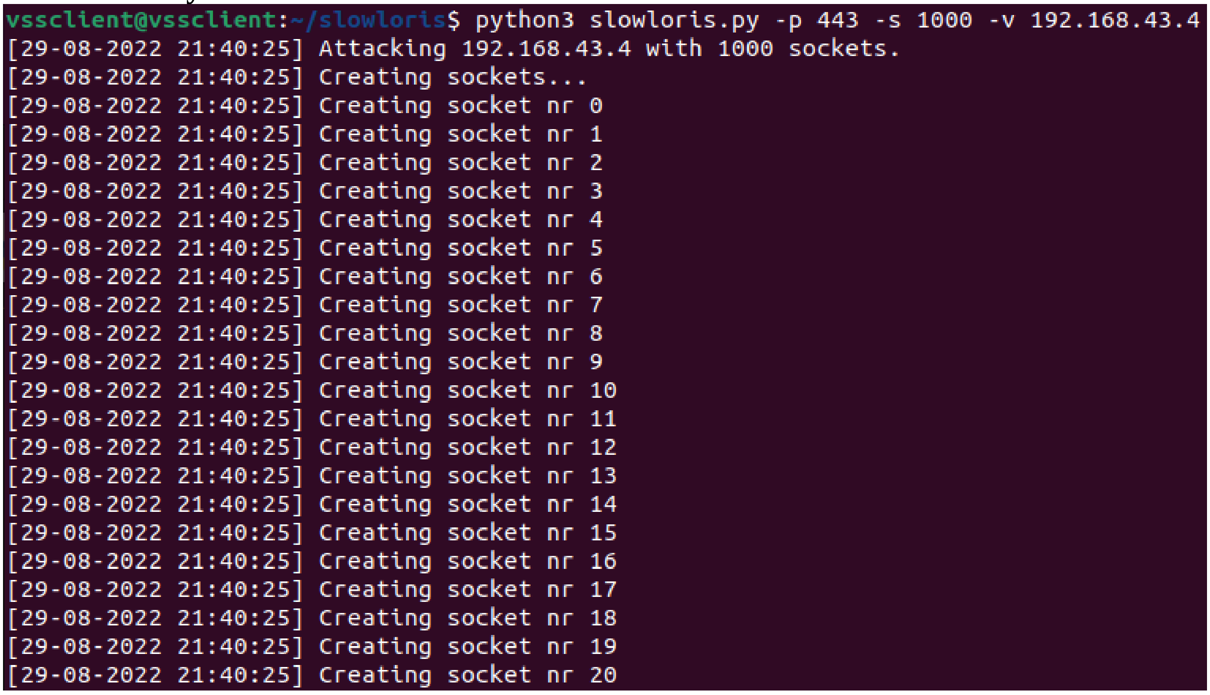


```
vssserver@vssserver:~$ sudo snort -A console -q -u snort -g snort -c /etc/snort/snort.conf -i enp0s3
[sudo] password for vssserver:
08/29-21:36:29.393661  [**] [1:100001:1] Pinging Alert [**] [Priority: 0] {ICMP} 192.168.43.12 -> 192
.168.43.4
08/29-21:36:30.424717  [**] [1:100001:1] Pinging Alert [**] [Priority: 0] {ICMP} 192.168.43.12 -> 192
.168.43.4
08/29-21:36:31.457142  [**] [1:100001:1] Pinging Alert [**] [Priority: 0] {ICMP} 192.168.43.12 -> 192
.168.43.4
08/29-21:36:32.473490  [**] [1:100001:1] Pinging Alert [**] [Priority: 0] {ICMP} 192.168.43.12 -> 192
.168.43.4
08/29-21:36:33.500241  [**] [1:100001:1] Pinging Alert [**] [Priority: 0] {ICMP} 192.168.43.12 -> 192
.168.43.4
08/29-21:36:34.522067  [**] [1:100001:1] Pinging Alert [**] [Priority: 0] {ICMP} 192.168.43.12 -> 192
.168.43.4
08/29-21:36:35.545513  [**] [1:100001:1] Pinging Alert [**] [Priority: 0] {ICMP} 192.168.43.12 -> 192
.168.43.4
08/29-21:36:36.570141  [**] [1:100001:1] Pinging Alert [**] [Priority: 0] {ICMP} 192.168.43.12 -> 192
.168.43.4
```

For DoS Attack by Slowloris Tool:

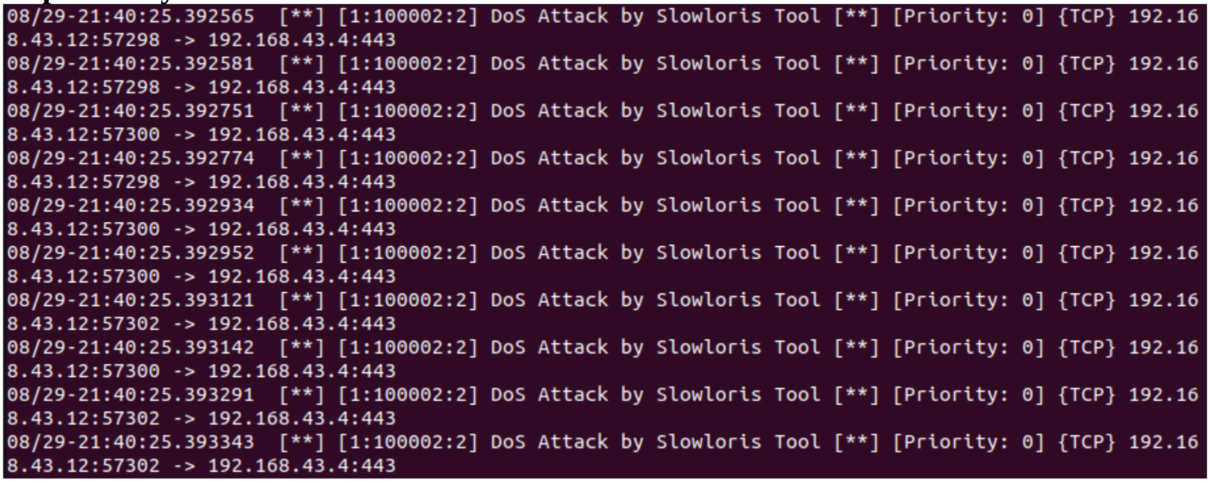
```
alert tcp any any -> 192.168.43.4 443 (msg:"DoS Attack by Slowloris Tool"; sid:100002; rev:2;)
```

DoS Attack by Slowloris Tool from client:



```
vssclient@vssclient:~/slowloris$ python3 slowloris.py -p 443 -s 1000 -v 192.168.43.4
[29-08-2022 21:40:25] Attacking 192.168.43.4 with 1000 sockets.
[29-08-2022 21:40:25] Creating sockets...
[29-08-2022 21:40:25] Creating socket nr 0
[29-08-2022 21:40:25] Creating socket nr 1
[29-08-2022 21:40:25] Creating socket nr 2
[29-08-2022 21:40:25] Creating socket nr 3
[29-08-2022 21:40:25] Creating socket nr 4
[29-08-2022 21:40:25] Creating socket nr 5
[29-08-2022 21:40:25] Creating socket nr 6
[29-08-2022 21:40:25] Creating socket nr 7
[29-08-2022 21:40:25] Creating socket nr 8
[29-08-2022 21:40:25] Creating socket nr 9
[29-08-2022 21:40:25] Creating socket nr 10
[29-08-2022 21:40:25] Creating socket nr 11
[29-08-2022 21:40:25] Creating socket nr 12
[29-08-2022 21:40:25] Creating socket nr 13
[29-08-2022 21:40:25] Creating socket nr 14
[29-08-2022 21:40:25] Creating socket nr 15
[29-08-2022 21:40:25] Creating socket nr 16
[29-08-2022 21:40:25] Creating socket nr 17
[29-08-2022 21:40:25] Creating socket nr 18
[29-08-2022 21:40:25] Creating socket nr 19
[29-08-2022 21:40:25] Creating socket nr 20
```

Captured by Snort:



```
08/29-21:40:25.392565  [**] [1:100002:2] DoS Attack by Slowloris Tool [**] [Priority: 0] {TCP} 192.16
8.43.12:57298 -> 192.168.43.4:443
08/29-21:40:25.392581  [**] [1:100002:2] DoS Attack by Slowloris Tool [**] [Priority: 0] {TCP} 192.16
8.43.12:57298 -> 192.168.43.4:443
08/29-21:40:25.392751  [**] [1:100002:2] DoS Attack by Slowloris Tool [**] [Priority: 0] {TCP} 192.16
8.43.12:57300 -> 192.168.43.4:443
08/29-21:40:25.392774  [**] [1:100002:2] DoS Attack by Slowloris Tool [**] [Priority: 0] {TCP} 192.16
8.43.12:57298 -> 192.168.43.4:443
08/29-21:40:25.392934  [**] [1:100002:2] DoS Attack by Slowloris Tool [**] [Priority: 0] {TCP} 192.16
8.43.12:57300 -> 192.168.43.4:443
08/29-21:40:25.392952  [**] [1:100002:2] DoS Attack by Slowloris Tool [**] [Priority: 0] {TCP} 192.16
8.43.12:57300 -> 192.168.43.4:443
08/29-21:40:25.393121  [**] [1:100002:2] DoS Attack by Slowloris Tool [**] [Priority: 0] {TCP} 192.16
8.43.12:57302 -> 192.168.43.4:443
08/29-21:40:25.393142  [**] [1:100002:2] DoS Attack by Slowloris Tool [**] [Priority: 0] {TCP} 192.16
8.43.12:57300 -> 192.168.43.4:443
08/29-21:40:25.393291  [**] [1:100002:2] DoS Attack by Slowloris Tool [**] [Priority: 0] {TCP} 192.16
8.43.12:57302 -> 192.168.43.4:443
08/29-21:40:25.393343  [**] [1:100002:2] DoS Attack by Slowloris Tool [**] [Priority: 0] {TCP} 192.16
8.43.12:57302 -> 192.168.43.4:443
```

Start Snort on the console:

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

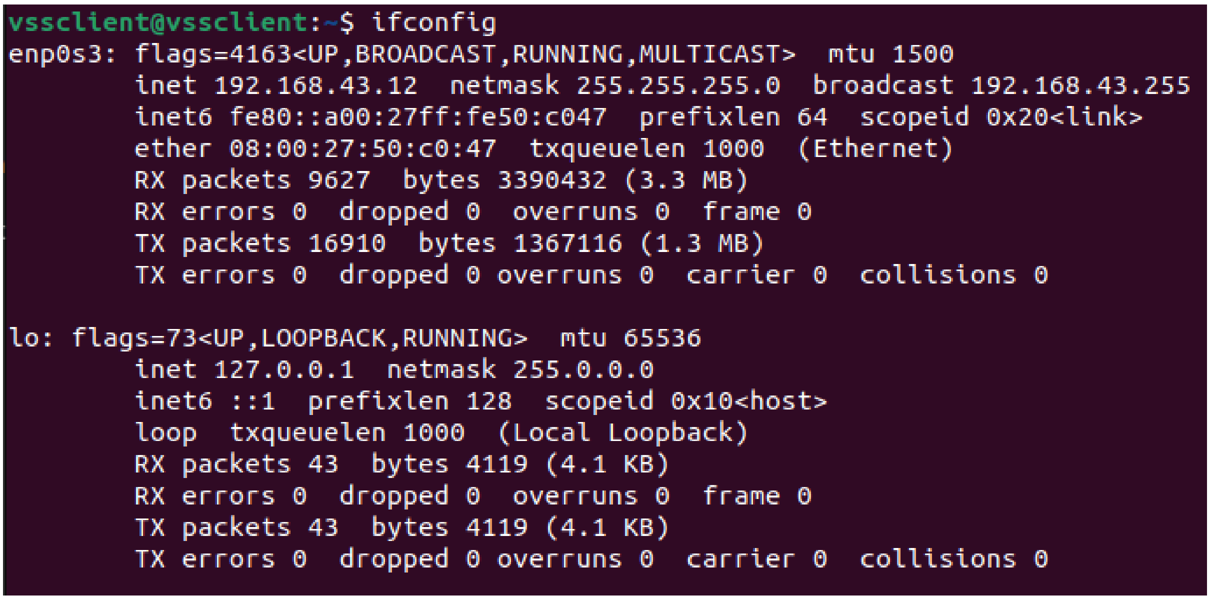
Configuring DNS Clients:

Client Ubuntu Machine Username:

```
vssclient
```

First, find the IPv4 for Client Ubuntu machine using the following command:

```
ifconfig
```



```
vssclient@vssclient:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.43.12  netmask 255.255.255.0  broadcast 192.168.43.255
        inet6 fe80::a00:27ff:fe50:c047  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:50:c0:47  txqueuelen 1000  (Ethernet)
        RX packets 9627  bytes 3390432 (3.3 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 16910  bytes 1367116 (1.3 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 43  bytes 4119 (4.1 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 43  bytes 4119 (4.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Here, the IPv4 is **192.168.43.12**, which will be the Client IP Address.

Find IPv4 in CIDR Notation:

```
ip address
```

Configuring Netplan:

```
sudo nano /etc/netplan/01-network-manager-all.yaml
```

Modify the file:

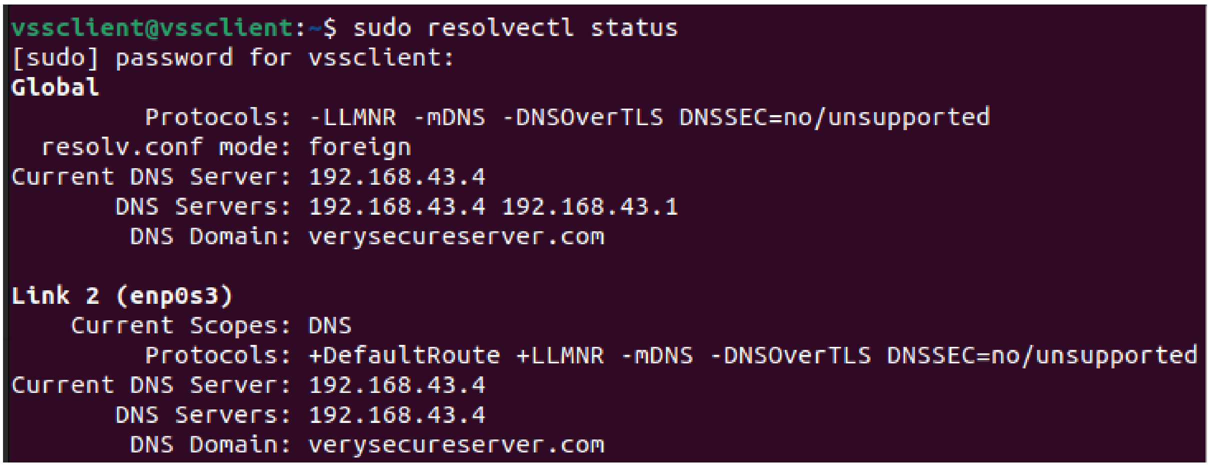
```
network:
  version: 2
  renderer: NetworkManager
  ethernets:
    enp0s3:
      dhcp4: no
      addresses: [ 192.168.43.12/24 ]
      routes:
        - to: default
          via: 192.168.43.1
      nameservers:
          addresses: [ 192.168.43.4 ]
          search: [ verysecureserver.com ]
```

Apply Configuration of Netplan:

```
sudo netplan try
sudo netplan apply
```

Check DNS Status:

```
sudo resolvectl status
```

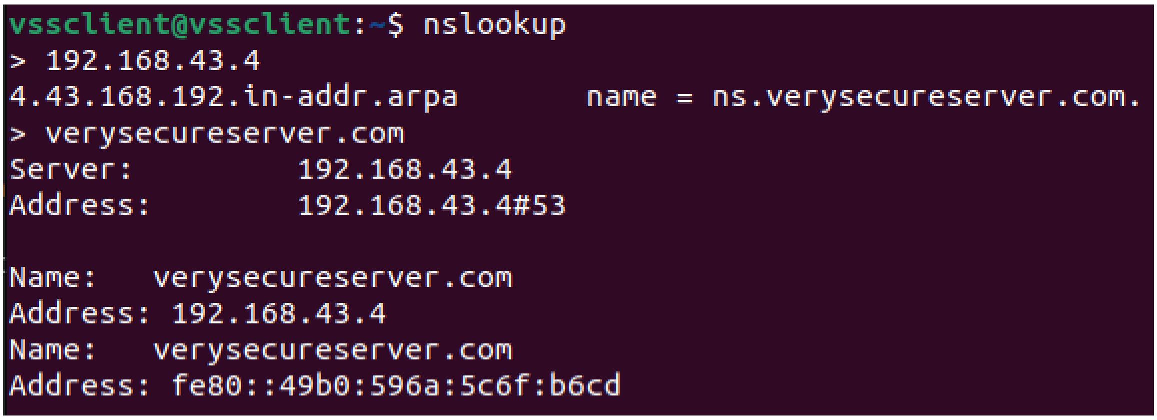


```
vssclient@vssclient:~$ sudo resolvectl status
[sudo] password for vssclient:
Global
         Protocols: -LLMNR -mDNS -DNSOverTLS DNSSEC=no/unsupported
  resolv.conf mode: foreign
Current DNS Server: 192.168.43.4
       DNS Servers: 192.168.43.4 192.168.43.1
        DNS Domain: verysecureserver.com

Link 2 (enp0s3)
    Current Scopes: DNS
         Protocols: +DefaultRoute +LLMNR -mDNS -DNSOverTLS DNSSEC=no/unsupported
Current DNS Server: 192.168.43.4
       DNS Servers: 192.168.43.4
        DNS Domain: verysecureserver.com
```

Search DNS Records using NsLookup:

```
nslookup
192.168.43.4
verysecureserver.com
```



```
vssclient@vssclient:~$ nslookup
> 192.168.43.4
4.43.168.192.in-addr.arpa       name = ns.verysecureserver.com.
> verysecureserver.com
Server:         192.168.43.4
Address:        192.168.43.4#53

Name:   verysecureserver.com
Address: 192.168.43.4
Name:   verysecureserver.com
Address: fe80::49b0:596a:5c6f:b6cd
```

Wireshark Tool Configuration for packet sniffing and analysis:

Install Wireshark:

```
sudo apt install wireshark
```

Modify User for Wireshark:

```
sudo usermod -aG wireshark vssclient
```

Start Wireshark:

```
sudo wireshark
```

Install Slowloris Tool for DoS Attack:

Install Git and Python3:

```
sudo apt install git
sudo apt install python3
```

Clone Slowloris Tool from Github:

```
git clone https://github.com/gkbrk/slowloris.git
```

Start Slowloris Tool for DoS Attack:

```
python3 slowloris.py -p 443 -s 1000 -v 192.168.43.4
```