Objective:

To utilize the virtualization concept of VMware and create a LAN network with DNS, DHCP and Web Servers on individual Linux virtual machines and implement networking protocols such as NFS, IPSec VPN tunnelling and ufw firewall.

Protocols implemented and their behavior:

Domain Name Server (DNS)

It is highly impractical to remember IP addresses of different hosts connected over networks. Domain name server acts a decentralized directory of internet or a private network, which contains the IP addresses of each domain names. When a user request to access a client host or server using domain name, it is forwarded to DNS server of domain which maps the IP address corresponding to domain name. DNS uses UDP packets to initiate or receive requests.

Dynamic Host Configuration Protocol (DHCP)

Dynamic Host Configuration Protocol is a standard client/server protocol which helps to assign dynamic IP addresses to host machines connected in a network. Since a unicast is required to access all the resources in a network, DHCP helps to achieve this by assigning unanimous gateways & address pool for each network segments. DHCP also reduces considerable amount of work in network administration by assigning dynamic IP addresses, subnets & gateways for a machine without any manual work.

Process of DHCP:

1. The user broadcasts a request for a DHCP server. This is the “discover” process.

2. An “offer” message is sent to the user by the DHCP server if the IP address is available.

3. The user then “requests” to lease the IP address from the DHCP server.

4. DHCP server sends an “acknowledged” message to the user and assigns IP address along with the

corresponding network parameters like subnet mask and default gateway.

Web Server:

A Web Server is a server software or a dedicated hardware that stores, process & deliver web pages to clients on the network. The communication between server & client happens with HTTP. In this project, we are using apache v2 as a web server for serving web content. The basic unit representing an individual website is the virtual host. When the domain is accessed by the user,

the user is directed by the apache server to a specific directory where the domain is stored and maintaining the secrecy of the fact that it has other websites stored in its repository. In this way each domain can be customized and configured individually and autonomously.

Firewall

Firewall is generally used to filter or restrict access to incoming or outgoing traffic based on certain rules. It creates a segregation between trusted local network & untrusted external network like network. In our project we have implemented firewall by using UFW (Uncomplicated Firewall), which is used to build firewalls based on IPv4 protocol.

Backup Server

Backup server generally acts as a redundant machine of main server. When there is any occurrence of catastrophe in main server, backup server takes charge. In this project backup server is configured along with automatic backup functionality that happens periodically

DHCP Implementation:

A static IP address should be set for the DHCP server:

```
sudo nano /etc/network/interfaces
```

Edit the /etc/network/interfaces file:

```
auto lo
iface lo inet loopback
auto ens33
iface ens33 inet static
address 192.168.73.2
netmask 255.255.255.0
network 192.168.73.0
broadcast 192.168.73.255
gateway 192.168.73.3
dns-search server.project3.com
dns-nameservers 192.168.73.3
```

To install an DHCP server to allocate IP addresses execute the following commands:

```
sudo apt-get install isc-dhcp-server
sudo nano /etc/default/isc-dhcp-server
```

Edit the following line with the interface name of your network adapter:

```
INTERFACESv4=”ens33”
```

Open the following file and edit:

```
sudo nano /etc/dhcp/dhcpd.conf
```

Uncomment the authoritative command:

```
# Since this is the official DHCP server for our local network, the “authoritative” command
#should be commented out.
authoritative;
```

The following configuration is added for our local subnetwork:

```
 subnet 192.168.73.0 netmask 255.255.255.0 {
 range 192.168.73.100 192.168.73.200;
 option domain-name-servers 192.168.73.3;
 option domain-name "project3.com";
 option subnet-mask 255.255.255.0;
 option routers 192.168.73.3;
 option broadcast-address 192.168.73.255;
 default-lease-time 600;
 max-lease-time 7200;
}
```

- The subnet address should be changed according to our requirements.
- The default gateway should be specified to the option routers.

Certain hosts like DNS servers and Web Servers require fixed IP addresses so that every time when these machines request an IP addresses from the DHCP server, the same IP address is assigned to them i.e. their IP addresses do not change. In order to do this, we specify the MAC addresses of these servers and assign them fixed IP addresses.

```
host dns-server1 {
 hardware ethernet 00:0c:29:f3:54:b0;
 fixed-address 192.168.73.3;
}
host dns-server2 {
 hardware ethernet 00:0c:29:3c:68:aa;
 fixed-address 192.168.73.4;
}

host web-server1 {
 hardware ethernet 00:0c:29:6a:6c:f3;
 fixed-address 192.168.73.5;
}
host web-server2 {
```

```
  hardware ethernet 00:0c:29:36:31:da;
  fixed-address 192.168.73.6;
}
```

Restart the network interface and the DHCP service and restart the system if needed:

```
sudo systemctl restart networking
sudo systemctl restart NetworkManager
sudo systemctl restart isc-dhcp-server
init 6
```

DNS Implementation:

Open the following file:

```
sudo nano /etc/hosts
```

Add the IP address and its corresponding domain name:

```
192.168.73.3 server.project3.com server
```

To change the hostname of the system, open the following file:

```
sudo nano /etc/hostname
```

Change it to the desired hostname:

```
DnsServer1
```

Open the following file:

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

Make the following changes in the file by providing the DNS server IP address and the corresponding domain name:

```
nameserver 192.168.73.3 #IP address of the master DNS server
search project3.com
```

Install bind9:

```
sudo apt-get install bind9 bind9utils bind9-doc
```

Add the forward and reverse zones to the named.conf.local file in the /etc/bind directory:

```
#forward zone
zone "project3.com" {
        type master;
        file "/etc/bind/forward.project3.com";
        allow-transfer { 192.168.73.4; }; #slave DNS IP
```

```
        also-notify { 192.168.73.4; };
};

#reverse zone
zone "73.168.192.in-addr.arpa" {
        type master;
        file "/etc/bind/reverse.project3.com";
        allow-transfer { 192.168.73.4; };
        also-notify { 192.168.73.4; };
};
```

Copy the db.local file to forward and reverse files for the DNS database service:

```
sudo cp db.local forward.proj3.com
```

Make the following changes to the forward file:

```
@       IN      SOA     server.project3.com. root.server.project3.com. (
                        2               ; Serial
                        604800          ; Refresh
                        86400       ; Retry
                         2419200        ; Expire
                         604800 )       ; Negative Cache TTL
;
        IN      NS      server.project3.com.
        IN      A       192.168.73.3
server  IN      A       192.168.73.3
host    IN      A       192.168.73.3
@       IN      NS      dns2.project3.com.
dns2    IN      A       192.168.73.4
www     IN      A       192.168.73.5
```

To create the reverse file:

```
sudo cp /etc/bind/forward.proj3.com /etc/bind/reverse.proj3.com
```

Make the following changes to the reverse file:

```
@       IN      SOA     server.project3.com. root.server.project3.com. (
                          2       ; Serial
                        604800          ; Refresh
                       86400            ; Retry
                        2419200         ; Expire
                        604800 )        ; Negative Cache TTL
```

```
;
        IN      NS      server.project3.com.
@       IN      PTR     project3.com.
server  IN      A       192.168.73.3
host    IN      A       192.168.73.3
        IN      NS      dns2.proj3.com.
www     IN      A       192.168.73.5
3       IN      PTR     server.proj3.com.
4       IN      PTR     dns2.proj3.com.
5       IN      PTR     www.proj3.com.
```

Restart the bind9 service and the system:

```
sudo systemctl restart bind9
init 6
```

To configure the Dns slave server on another machine, repeat the above steps.

Open the following file on the slave machine:

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

Make the following changes in the file:

```
nameserver 192.168.73.4 #IP address of the slave DNS server
search project3.com
```

To configure the forward and reverse zones for the slave DNS server:

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

Make the following changes to the file:

```
#forward zone
zone "project3.com" {
        type slave;
        file "/etc/bind/forward.project3.com";
        masters { 192.168.73.3; };
};
#reverse zone
zone "73.168.192.in-addr.arpa" {
        type slave;
        file "/etc/bind/reverse.proj3.com";
        masters { 192.168.73.3; };
};
```

Restart the bind9 service and the system:
```
sudo systemctl restart bind9
init 6
```

Web Server:
Install the Apache web server:
```
sudo apt-get install apache2
```

Make changes to the default webpage by changing the following file as required:
```
sudo nano /var/www/html/index.html
```

Web Server backup:
To schedule a full backup of the web server to a remote machine i.e. web server 2, we should first create a local copy of the backup.
```
mkdir -p backup/html
```

Now we use tar to create a backup of the /var/www/html directory:
```
sudo tar -czf backup/html/backup_file.tar.gz /var/www/html
```

Additionally, we can add the date to each of our backups using the following command:
```
sudo tar -czf backup/html/backup_file-`date '+%m%d%y'`.tar.gz /var/www/html
```

In order to secure the backup to a remote server we need to use SSH. First generate the SSH key on the current machine using the following command.
```
ssh-keygen
```

In order to view the generated public key:
```
cat .ssh/id_rsa.pub
```

If it doesn't already exist, we need to create the following files on the remote server.
```
mkdir -p /home/santhoshvijay/.ssh
touch /home/santhoshvijay/.ssh/authorized_keys
mkdir -p /home/santhoshvijay/backup/backups
```
To copy the public key from the web server to remote server we use scp:
```
scp .ssh/id_rsa.pub santhoshvijay@192.168.73.6:/home/santhoshvijay/backup_key.pub
```

On the remote server, we should copy the received public key to the authorized_keys file to enable ssh without requiring a password:
```
cat /home/santhoshvijay/backup_key.pub >> /home/santhoshvijay/.ssh/authorized_keys
```

Now, we can use this to schedule backups using crontab. Open crontab using the following command:
EDITOR=nano crontab –e

Enter the following line to schedule a backup every day at 20:00:
00 20 * * * /bin/tar -czf /home/santhoshvijay/backup/html/backup_file-`date +\%m\%d\%y`.tar.gz /var/www/html;/usr/bin/scp -i /home/santhoshvijay/.ssh/id_rsa /home/santhoshvijay/backup/html/backup_file-`date +\%m\%d\%y`.tar.gz santhoshvijay@192.168.73.6:/home/santhoshvijay/backup/backups

Firewall:
To setup a firewall for the webserver, we need to install ufw.
sudo apt-get install ufw

Now we need to set the default policies for the firewall:
sudo ufw default deny incoming
sudo ufw default allow outgoing

In order to allow SSH, HTTP, FTP and HTTPS connections, set the following policies:
sudo ufw allow from 192.168.73.0/24 to any port 443
sudo ufw allow from 192.168.73.0/24 to any port 80
sudo ufw allow from 192.168.73.0/24 to any port 21
sudo ufw allow from 192.168.73.0/24 to any port 22

To disable pinging to the web server:
sudo nano /etc/ufw/before.rules

Comment this line:
#-A ufw-before-input -p icmp --icmp-type echo-request -j ACCEPT

To enable the firewall, we run the following:
sudo ufw enable

ADD-ONS:
NFS:
To install the NFS server:
sudo apt-get install nfs-kernel-server

Open /etc/exports:
```
sudo nano /etc/exports
```

We are going to share the /home with the two clients from the Web server. Add the following lines to the file:
```
/home      192.168.73.103(rw,sync,no_root_squash,no_subtree_check)
/home      192.168.73.104(rw,sync,no_root_squash,no_subtree_check)
```

Start the NFS service:
```
sudo systemctl start nfs-kernel-server
```

Install the NFS service on both the clients:
```
sudo apt-get install nfs-common
```

To mount the exported folders from the NFS server on the clients:
```
sudo mount 192.168.73.6:/home /nfs/home
```

Execute the following command on the client machines to verify the mounted files:
```
df –h
```

Alternatively, check if the files have been mounted by changing to the /nfs/home directory:
```
cd /nfs/home
```

IPSec VPN tunnel:
Install IPSec VPN using strongswan:
```
sudo apt-get install ipsec-tools strongswan-starter
```

Open the /etc/ipsec.conf file and edit:
```
sudo nano /etc/ipsec.conf
```

Add the connection here:
```
conn ubuntu-to-ubuntu
        authby=secret
        auto=route
        keyexchange=ike
        left=192.168.73.102
        right=192.168.73.103
        type=tunnel
```

```
        esp=aes128gcm16!
```

Add the RSA private keys or the password in the /etc/ipsec.secrets file:
`sudo nano /etc/ipsec.secrets`
Here, we add the password:
`192.168.73.102 192.168.73.103 : PSK "secret"`

Restart the IPSec service:
`sudo systemctl restart ipsec`

Repeat the above steps on the second machine by swapping the details of the two machines accordingly.
To verify, continuously ping from one server to the other and run the following command:
`watch ipsec statusall`

Testing:

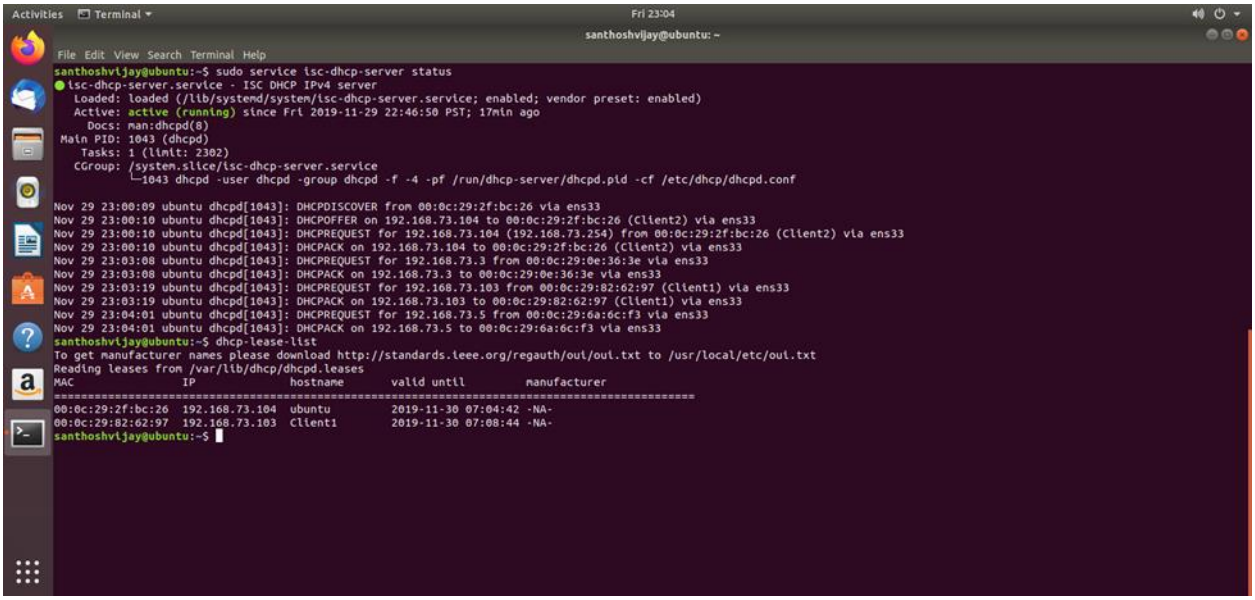
1.) DHCP Testing:

If the clients are able to get the IP addresses from the DHCP server within the range of IP addresses defined in the server pool, then the DHCP is working properly.

We can also see the status of the DHCP servers by using the commands:

`systemctl status isc-dhcp-server`

`dhcp-lease-list`



2.) DNS Testing:

For testing the functioning and effectiveness of DNS, the following commands will be useful:

- Dig:

  Dig is used to perform DNS lookups and returns the responses from the name servers. Execute the following command:

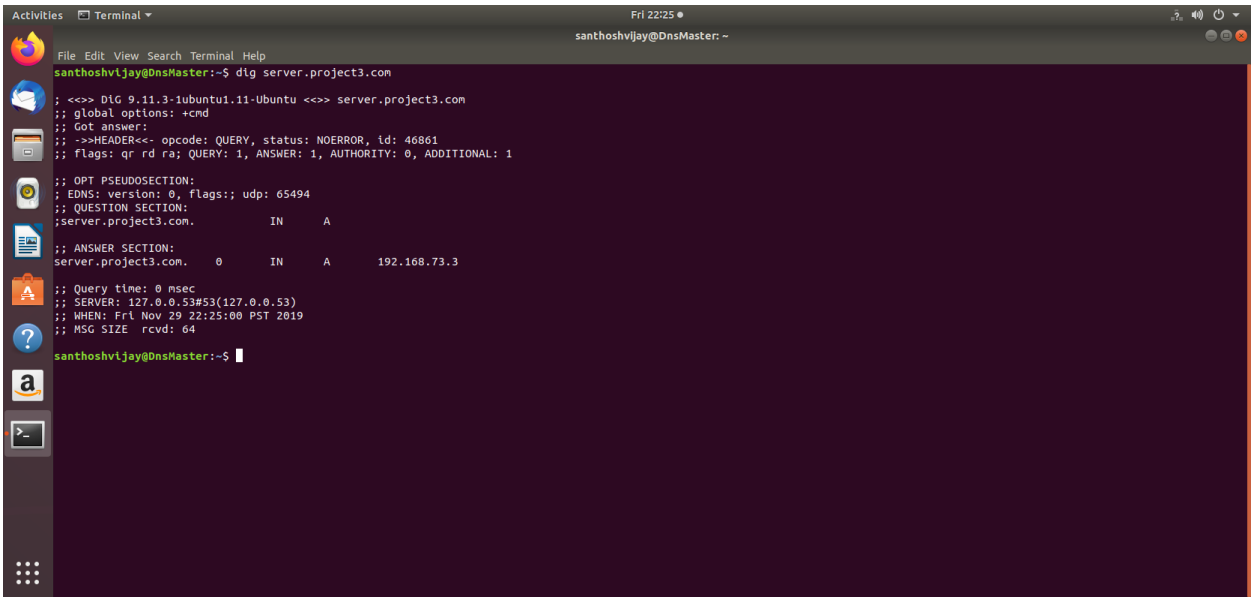
  dig server.project3.com

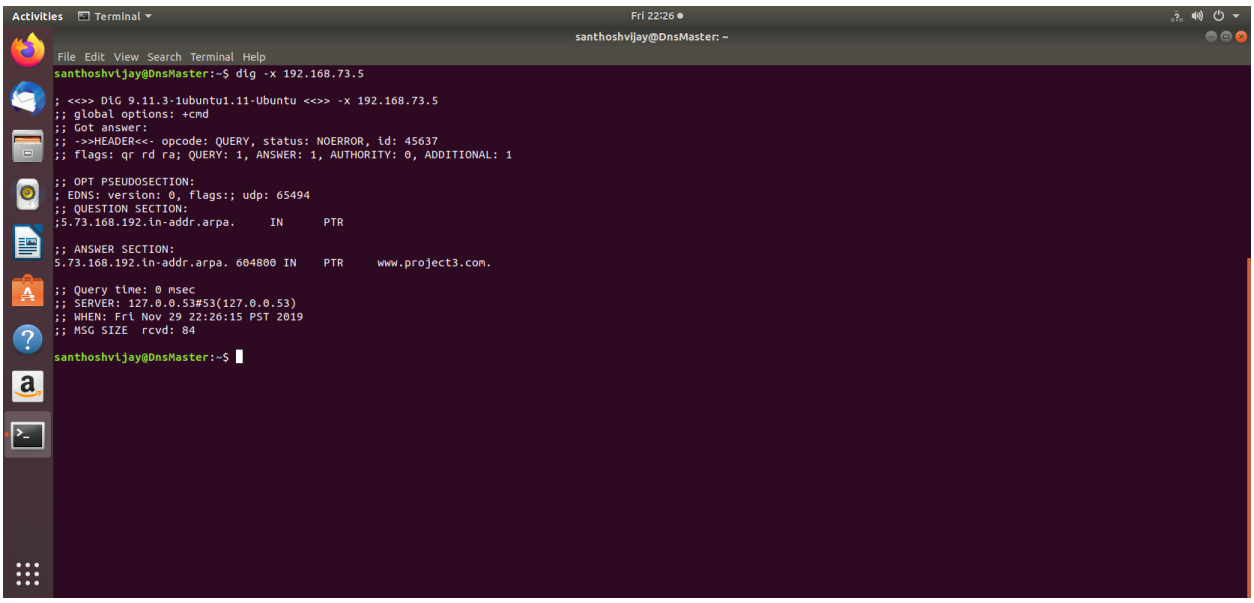
  dig -x 192.168.73.5

  This will return with the records from the master server.

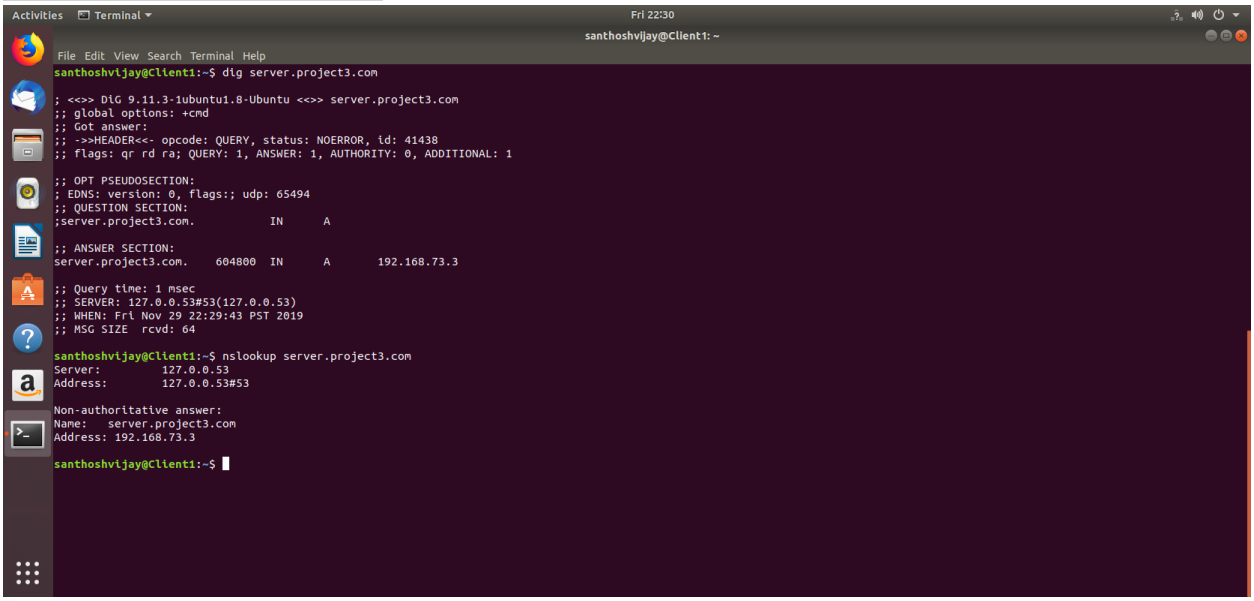
- Nslookup:

  Nslookup is a command used to query DNS servers. It can be checked by executing the following command:



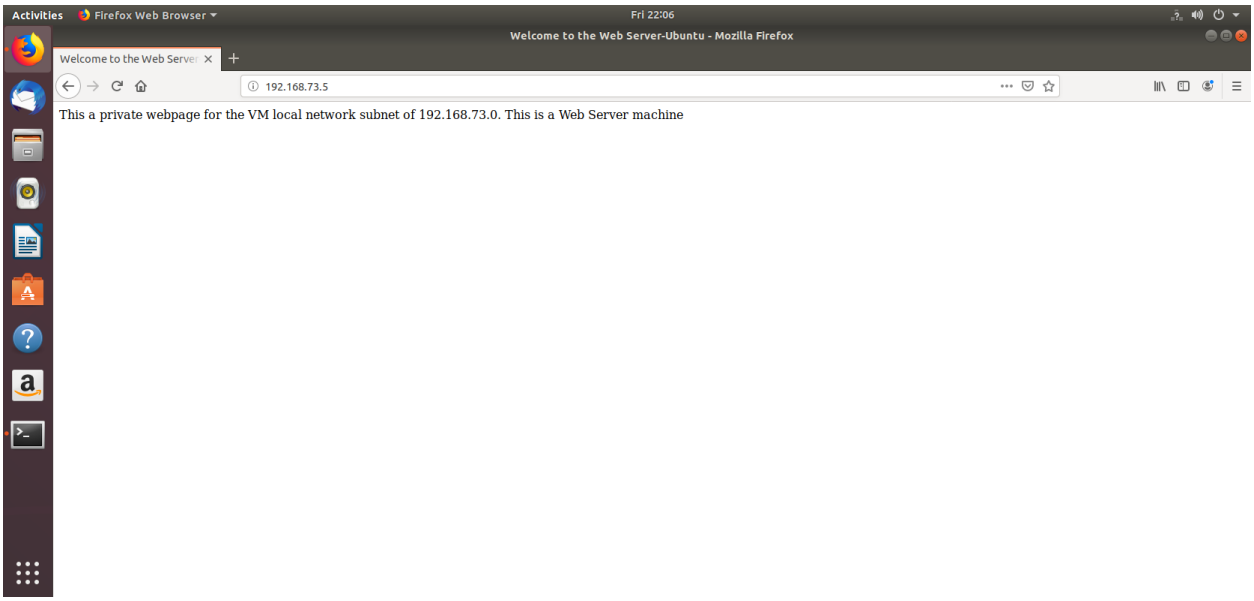


nslookup server.project3.com



3. Web server Test

Open the web browser in Ubuntu and enter the host name or the local IP address. If the html page hosted locally is shown, then the web server is up and running.



4. Firewall Test Ubuntu

When the firewall is activated on web servers, then those servers cannot be pinged as per our firewall policies. Hence if we try to ping the web servers or machines, we will get “request timed out “as the response.

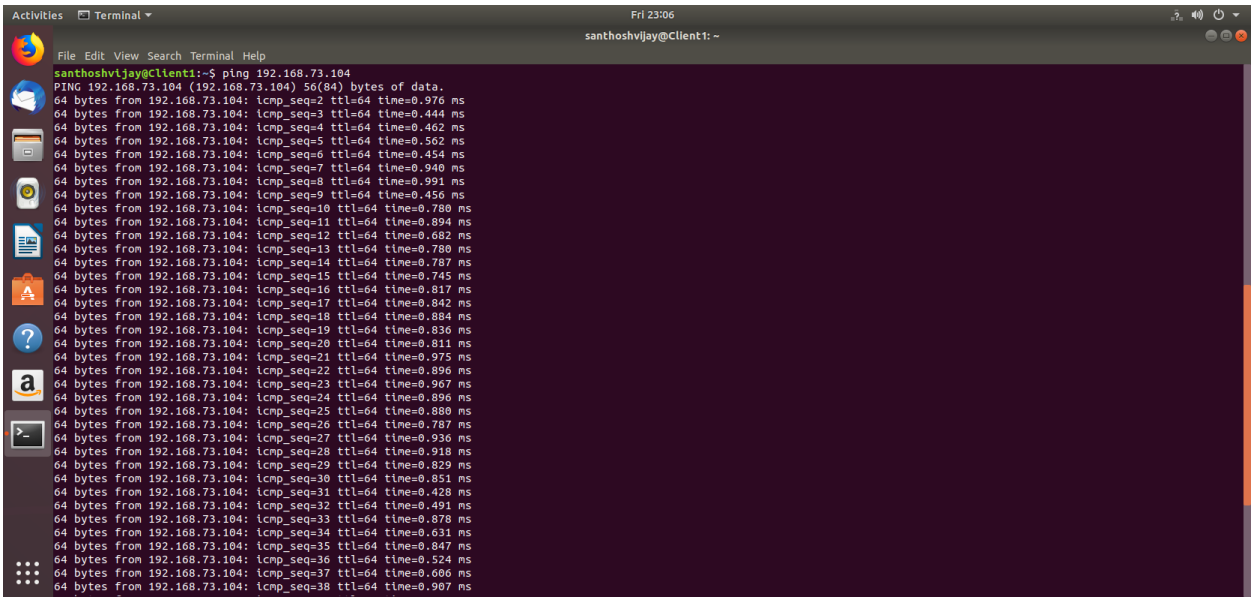
5. IPsec VPN Test

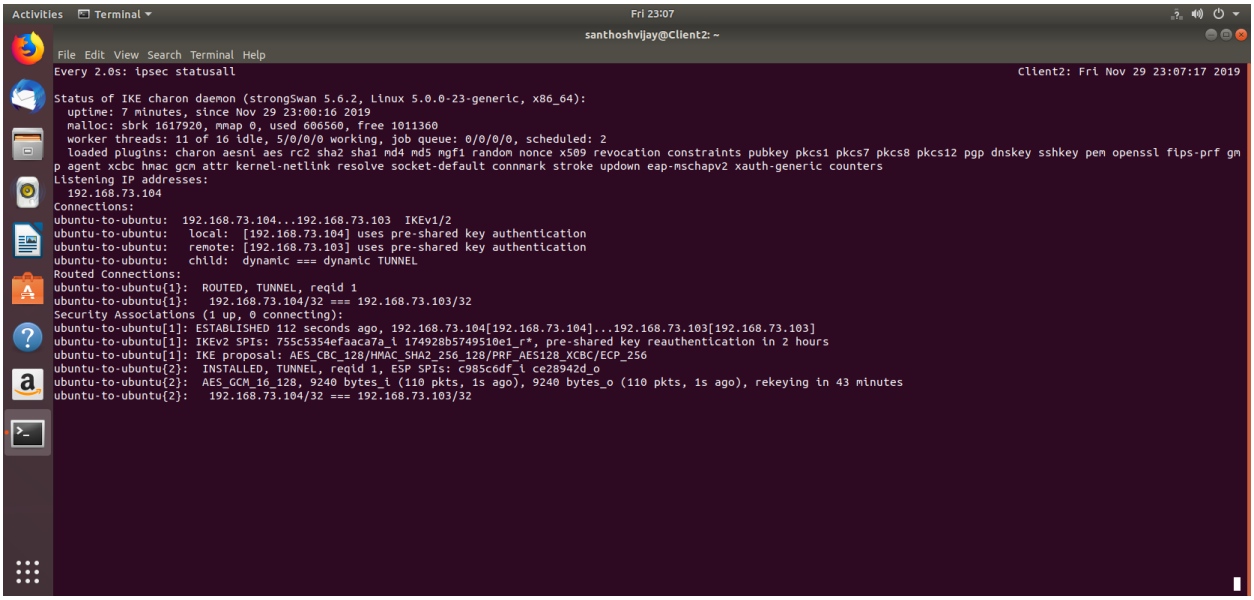
To test IPSEC VPN, run a continuous ping from one client to the other. Go to the other client and then run the following commands:

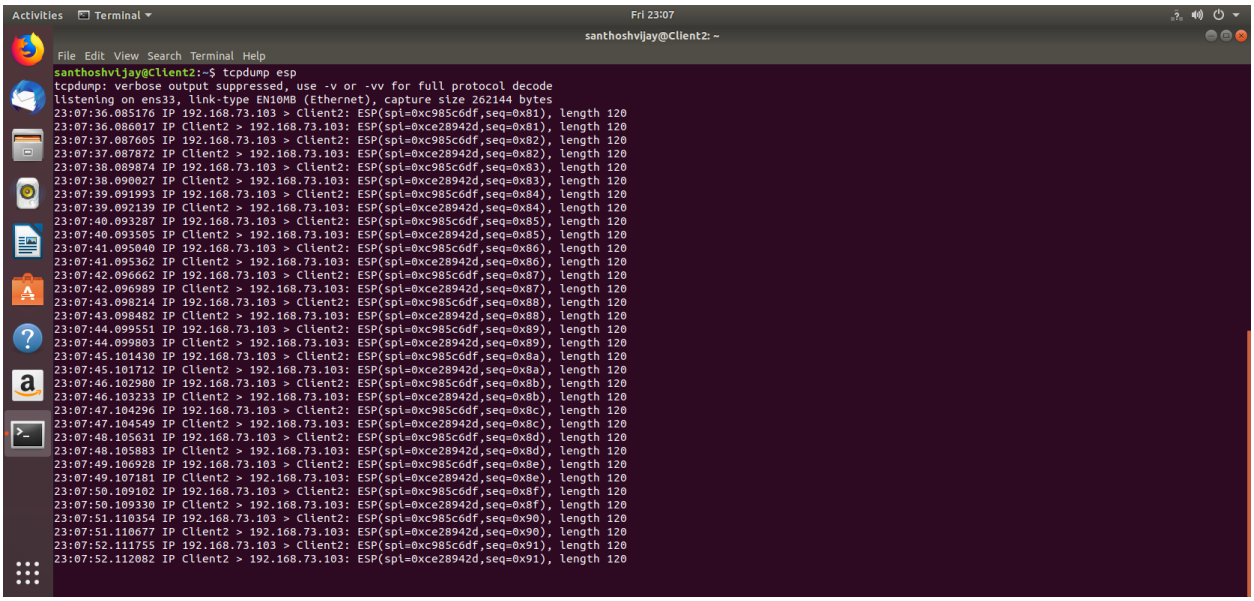
```
watch ipsec statusall
```

or

```
tcpdump esp
```


Activities
Terminal
Fri 23:06
santhoshvijay@Client1: ~
File Edit View Search Terminal Help
santhoshvijay@Client1:~$ ping 192.168.73.104
PING 192.168.73.104 (192.168.73.104) 56(84) bytes of data.
64 bytes from 192.168.73.104: icmp_seq=2 ttl=64 time=0.976 ms
64 bytes from 192.168.73.104: icmp_seq=3 ttl=64 time=0.444 ms
64 bytes from 192.168.73.104: icmp_seq=4 ttl=64 time=0.462 ms
64 bytes from 192.168.73.104: icmp_seq=5 ttl=64 time=0.562 ms
64 bytes from 192.168.73.104: icmp_seq=6 ttl=64 time=0.454 ms
64 bytes from 192.168.73.104: icmp_seq=7 ttl=64 time=0.940 ms
64 bytes from 192.168.73.104: icmp_seq=8 ttl=64 time=0.991 ms
64 bytes from 192.168.73.104: icmp_seq=9 ttl=64 time=0.456 ms
64 bytes from 192.168.73.104: icmp_seq=10 ttl=64 time=0.780 ms
64 bytes from 192.168.73.104: icmp_seq=11 ttl=64 time=0.894 ms
64 bytes from 192.168.73.104: icmp_seq=12 ttl=64 time=0.682 ms
64 bytes from 192.168.73.104: icmp_seq=13 ttl=64 time=0.780 ms
64 bytes from 192.168.73.104: icmp_seq=14 ttl=64 time=0.787 ms
64 bytes from 192.168.73.104: icmp_seq=15 ttl=64 time=0.745 ms
64 bytes from 192.168.73.104: icmp_seq=16 ttl=64 time=0.817 ms
64 bytes from 192.168.73.104: icmp_seq=17 ttl=64 time=0.842 ms
64 bytes from 192.168.73.104: icmp_seq=18 ttl=64 time=0.884 ms
64 bytes from 192.168.73.104: icmp_seq=19 ttl=64 time=0.836 ms
64 bytes from 192.168.73.104: icmp_seq=20 ttl=64 time=0.811 ms
64 bytes from 192.168.73.104: icmp_seq=21 ttl=64 time=0.975 ms
64 bytes from 192.168.73.104: icmp_seq=22 ttl=64 time=0.896 ms
64 bytes from 192.168.73.104: icmp_seq=23 ttl=64 time=0.967 ms
64 bytes from 192.168.73.104: icmp_seq=24 ttl=64 time=0.896 ms
64 bytes from 192.168.73.104: icmp_seq=25 ttl=64 time=0.880 ms
64 bytes from 192.168.73.104: icmp_seq=26 ttl=64 time=0.787 ms
64 bytes from 192.168.73.104: icmp_seq=27 ttl=64 time=0.936 ms
64 bytes from 192.168.73.104: icmp_seq=28 ttl=64 time=0.918 ms
64 bytes from 192.168.73.104: icmp_seq=29 ttl=64 time=0.829 ms
64 bytes from 192.168.73.104: icmp_seq=30 ttl=64 time=0.851 ms
64 bytes from 192.168.73.104: icmp_seq=31 ttl=64 time=0.428 ms
64 bytes from 192.168.73.104: icmp_seq=32 ttl=64 time=0.491 ms
64 bytes from 192.168.73.104: icmp_seq=33 ttl=64 time=0.878 ms
64 bytes from 192.168.73.104: icmp_seq=34 ttl=64 time=0.631 ms
64 bytes from 192.168.73.104: icmp_seq=35 ttl=64 time=0.847 ms
64 bytes from 192.168.73.104: icmp_seq=36 ttl=64 time=0.524 ms
64 bytes from 192.168.73.104: icmp_seq=37 ttl=64 time=0.606 ms
64 bytes from 192.168.73.104: icmp_seq=38 ttl=64 time=0.907 ms


Activities
Terminal
Fri 23:07
santhoshvijay@Client2: ~
File Edit View Search Terminal Help
Every 2.0s: ipsec statusall
Client2: Fri Nov 29 23:07:17 2019
Status of IKE charon daemon (strongSwan 5.6.2, Linux 5.0.0-23-generic, x86_64):
uptime: 7 minutes, since Nov 29 23:00:16 2019
malloc: sbrk 1617920, mmap 0, used 606560, free 1011360
worker threads: 11 of 16 idle, 5/0/0/0 working, job queue: 0/0/0/0, scheduled: 2
loaded plugins: charon aesni aes rc2 sha2 sha1 md4 md5 mgf1 random nonce x509 revocation constraints pubkey pkcs1 pkcs7 pkcs8 pkcs12 pgp dnskey sshkey pem openssl fips-prf gmp agent xcbc hmac gcm attr kernel-netlink resolve socket-default connmark stroke updown eap-mschapv2 xauth-generic counters
Listening IP addresses:
192.168.73.104
Connections:
ubuntu-to-ubuntu: 192.168.73.104...192.168.73.103 IKEv1/2
ubuntu-to-ubuntu: local: [192.168.73.104] uses pre-shared key authentication
ubuntu-to-ubuntu: remote: [192.168.73.103] uses pre-shared key authentication
ubuntu-to-ubuntu: child: dynamic === dynamic TUNNEL
Routed Connections:
ubuntu-to-ubuntu{1}: ROUTED, TUNNEL, reqid 1
ubuntu-to-ubuntu{1}: 192.168.73.104/32 === 192.168.73.103/32
Security Associations (1 up, 0 connecting):
ubuntu-to-ubuntu[1]: ESTABLISHED 112 seconds ago, 192.168.73.104[192.168.73.104]...192.168.73.103[192.168.73.103]
ubuntu-to-ubuntu[1]: IKEv2 SPIs: 755c5354efaaca7a_i 174928b5749510e1_r*, pre-shared key reauthentication in 2 hours
ubuntu-to-ubuntu[1]: IKE proposal: AES_CBC_128/HMAC_SHA2_256_128/PRF_AES128_XCBC/ECP_256
ubuntu-to-ubuntu[2]: INSTALLED, TUNNEL, reqid 1, ESP SPIs: c985c6df_i ce28942d_o
ubuntu-to-ubuntu[2]: AES_GCM_16_128, 9240 bytes_i (110 pkts, 1s ago), 9240 bytes_o (110 pkts, 1s ago), rekeying in 43 minutes
ubuntu-to-ubuntu[2]: 192.168.73.104/32 === 192.168.73.103/32


Activities
Terminal
Fri 23:07
santhoshvijay@Client2: ~
File Edit View Search Terminal Help
santhoshvijay@Client2:~$ tcpdump esp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens33, link-type EN10MB (Ethernet), capture size 262144 bytes
23:07:36.085176 IP 192.168.73.103 > Client2: ESP(spi=0xc985c6df,seq=0x81), length 120
23:07:36.086017 IP Client2 > 192.168.73.103: ESP(spi=0xce28942d,seq=0x81), length 120
23:07:37.087605 IP 192.168.73.103 > Client2: ESP(spi=0xc985c6df,seq=0x82), length 120
23:07:37.087872 IP Client2 > 192.168.73.103: ESP(spi=0xce28942d,seq=0x82), length 120
23:07:38.089874 IP 192.168.73.103 > Client2: ESP(spi=0xc985c6df,seq=0x83), length 120
23:07:38.090027 IP Client2 > 192.168.73.103: ESP(spi=0xce28942d,seq=0x83), length 120
23:07:39.091993 IP 192.168.73.103 > Client2: ESP(spi=0xc985c6df,seq=0x84), length 120
23:07:39.092139 IP Client2 > 192.168.73.103: ESP(spi=0xce28942d,seq=0x84), length 120
23:07:40.093287 IP 192.168.73.103 > Client2: ESP(spi=0xc985c6df,seq=0x85), length 120
23:07:40.093505 IP Client2 > 192.168.73.103: ESP(spi=0xce28942d,seq=0x85), length 120
23:07:41.095040 IP 192.168.73.103 > Client2: ESP(spi=0xc985c6df,seq=0x86), length 120
23:07:41.095362 IP Client2 > 192.168.73.103: ESP(spi=0xce28942d,seq=0x86), length 120
23:07:42.096662 IP 192.168.73.103 > Client2: ESP(spi=0xc985c6df,seq=0x87), length 120
23:07:42.096989 IP Client2 > 192.168.73.103: ESP(spi=0xce28942d,seq=0x87), length 120
23:07:43.098214 IP 192.168.73.103 > Client2: ESP(spi=0xc985c6df,seq=0x88), length 120
23:07:43.098482 IP Client2 > 192.168.73.103: ESP(spi=0xce28942d,seq=0x88), length 120
23:07:44.099551 IP 192.168.73.103 > Client2: ESP(spi=0xc985c6df,seq=0x89), length 120
23:07:44.099803 IP Client2 > 192.168.73.103: ESP(spi=0xce28942d,seq=0x89), length 120
23:07:45.101430 IP 192.168.73.103 > Client2: ESP(spi=0xc985c6df,seq=0x8a), length 120
23:07:45.101712 IP Client2 > 192.168.73.103: ESP(spi=0xce28942d,seq=0x8a), length 120
23:07:46.102980 IP 192.168.73.103 > Client2: ESP(spi=0xc985c6df,seq=0x8b), length 120
23:07:46.103233 IP Client2 > 192.168.73.103: ESP(spi=0xce28942d,seq=0x8b), length 120
23:07:47.104296 IP 192.168.73.103 > Client2: ESP(spi=0xc985c6df,seq=0x8c), length 120
23:07:47.104549 IP Client2 > 192.168.73.103: ESP(spi=0xce28942d,seq=0x8c), length 120
23:07:48.105631 IP 192.168.73.103 > Client2: ESP(spi=0xc985c6df,seq=0x8d), length 120
23:07:48.105883 IP Client2 > 192.168.73.103: ESP(spi=0xce28942d,seq=0x8d), length 120
23:07:49.106928 IP 192.168.73.103 > Client2: ESP(spi=0xc985c6df,seq=0x8e), length 120
23:07:49.107181 IP Client2 > 192.168.73.103: ESP(spi=0xce28942d,seq=0x8e), length 120
23:07:50.109102 IP 192.168.73.103 > Client2: ESP(spi=0xc985c6df,seq=0x8f), length 120
23:07:50.109330 IP Client2 > 192.168.73.103: ESP(spi=0xce28942d,seq=0x8f), length 120
23:07:51.110354 IP 192.168.73.103 > Client2: ESP(spi=0xc985c6df,seq=0x90), length 120
23:07:51.110677 IP Client2 > 192.168.73.103: ESP(spi=0xce28942d,seq=0x90), length 120
23:07:52.111755 IP 192.168.73.103 > Client2: ESP(spi=0xc985c6df,seq=0x91), length 120
23:07:52.112082 IP Client2 > 192.168.73.103: ESP(spi=0xce28942d,seq=0x91), length 120