School of Computing

B.Sc Computer Science (Infrastructure).

Programme Code: DT211C/4

2013– 2017

Systems Integration Assignment 2

Student Name: *Micheál Slattery*

Student Number: *C12383326*

Assignment: *CA2*

Lecturer: *Brian Keegan*

Date for Return: *5/12/2016*

Overview

This report will show how to configure a client and server type relationship between two nodes using VirtualBox.
There will be two Ubuntu 14.04 machines in use, one called node1 and another called node2. Node1 will act as the server and node2 as the client.
Node 1 will act as a router, connecting through eth0 to the internet and eth1 as a static ip address on which ip addresses will be allocated in the range of 192.168.1.150 to 192.168.1.200.
This report will show how to install, setup and configure DHCP using isc-dhcp-server, DNS using bind9, SSH using openssh, NFS using nfs-kernel-server and FTP using vsftpd, which is a FTP daemon for ubuntu.

Initial setup

Ensure you have two virtual machines setup and you have and know the sudo password as there are configuration files that must be manually changed. Without the sudo password, changes will not be able to be saved and thus the setup hindered.
For this report, I will be using ubuntu 14.04 and VirtualBox with two nodes.
Clone the base machine of your virtual machine and call it node1, this will act as the server.
Clone the base machine of your VM and call it node2, this will act as the client.

DHCP

Introduction:

Dynamic Host Configuration Protocol is a network service that enables client computers to be automatically assigned settings from a server computer.
The server configures each of the settings and the client receives these settings without knowing. It is important that DHCP is configured correctly in order to allow clients who request internet access, to receive internet access.
As previously stated, we will be using NODE1 as the server and NODE2 as the client.
We will be using the isc-dhcp-server package for this report.

step1)

Login to Node1

Step2)

On node1 (Acting as server) install DHCP server using the command

sudo apt-get install isc-dhcp-server

Step3)

Configure the dhcp server by doing the following:
'ifconfig eth0'
-Type the following command after taking note of inet address on eth0

sudo nano /etc/dhcp/dhcpd.conf

-Inside this file, you should find the following commented out section, fill it out by copying as here, but changing the 192.168.".0 to whatever is on your eth0.

 # A slightly different configuration for an internal subnet.

subnet 192.168.1.0 netmask 255.255.255.0 {
range 192.168.1.150 192.168.1.200;
option domain-name-servers 192.168.1.1, 8.8.4.4;
 # option domain-name "internal.example.org";
option routers 192.168.1.1;
option broadcast-address 192.168.1.255;
default-lease-time 600;
max-lease-time 7200;
}

-Also, uncomment '#authoritative'

This allows for the dhcp server to act as the official DHCP server for the local network.
These settings will result in the DHCP server giving node2 ip address from the range 192.168.1.150-192.168.1.200.

Step4)

Type in the following command:

Sudo nano /etc/init.d/isc-dhcp-server restart

-Followed by:

Sudo nano /etc/network/interfaces

The first command resets the server, allowing changes to be committed.
The second command opens the network interfaces, this is where configuration occurs.

-Inside /etc/network/interfaces, fill out the following:

 # The loopback network interface
auto lo
iface lo inet loopback

 # The primary network interface
auto eth0
iface eth0 inet dhcp

auto eth1
iface eth1 inet static
address 192.168.1.11
netmask 255.255.255.0
network 192.168.1.0
broadcast 192.168.1.255
gateway 192.168.1.1
#post-up route add -net 192.168.0.0 netmask 255.255.0.0 gw 192.168.1.1 dev eth1
#pre-down route del -net 192.168.0.0 netmask 255.255.0.0 gw 192.168.1.1 dev eth1

This allows eth0 to act as the connection to the internet and a static address on eth1 which node2 will connect to and receive addresses from.

Step5)
Now, go restart the dhcp server by typing in the command

Sudo /etc/init.d/networking restart

This restarts the network settings and allows changes to be committed.

Step6 NODE2)
Log into node2 and configure the network settings by typing the command

Sudo nano /etc/network/interfaces

From there, fill out the following as below:

auto lo
iface lo inet loopback

#auto eth0
#iface eth0 inet dhcp

#Gets ip address off node1 via eth1

auto eth1
iface eth1 inet dhcp

This gets an ip address off of node1 via dhcp which will should give it 192.168.1.150 (As it's the only one connecting on eth1)

Step7)

Restart services by using the command

Sudo /etc/init.d/networking restart

Step8)

Verify that it is working by running:

Sudo reboot

This reboots the machine and and starts fresh.
Log back in and run the command:

Ifconfig

Output should be as follows:

```
network@node2:~$ ifconfig
eth1  Link encap:Ethernet  HWaddr 08:00:27:3e:e0:30
inet addr:192.168.1.150  Bcast:192.168.1.255  Mask:255.255.255.0
inet6 addr: fe80::a00:27ff:fe3e:e030/64 Scope:Link
UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
RX packets:143 errors:0 dropped:0 overruns:0 frame:0
TX packets:99 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:14479 (14.4 KB)  TX bytes:15856 (15.8 KB)

lo        Link encap:Local Loopback
inet addr:127.0.0.1  Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING  MTU:65536  Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

If this is what you see, then you have configured it correctly.

Step9)

Verification can be done by using the following command from node2:

Ping -c 5 192.168.1.11

-This will ping the server 5 times.

If you see a return like below, then your client and server are set-up correctly. For added testing, ping various websites (rte.ie, facebook.com, india.gov.in(response time should be longer!))

PING 192.168.1.11 (192.168.1.11) 56(84) bytes of data.
64 bytes from 192.168.1.11: icmp_seq=1 ttl=64 time=0.356 ms
64 bytes from 192.168.1.11: icmp_seq=2 ttl=64 time=0.452 ms
64 bytes from 192.168.1.11: icmp_seq=3 ttl=64 time=0.448 ms
64 bytes from 192.168.1.11: icmp_seq=4 ttl=64 time=0.386 ms
64 bytes from 192.168.1.11: icmp_seq=5 ttl=64 time=0.767 ms

--- 192.168.1.11 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 3999ms

DHCP Completed.

NFS

Introduction:

NFS (Network File System) allows you to 'share' a directory located on one networked computer with other computers/devices on that network. Where the directory is located is called the server and the devices that connect to it are called clients. This allows files to be shared, viewed and altered by computers on a shared network.
We will be using the nfs-kernel-server package in this report.

SERVER SIDE
(NODE1)

Step1

On the server (node 1), run the following command:

apt-get install nfs-kernel-server

Step2

On the server (node1), Create the export file system.

mkdir -p /export/users

Step3

Export and /export/users must have 777 permissions. To do this, run the following:

Sudo chmod 777 /exports
Sudo chmod 777 /exports/users

Step4

Now mount the real users directory with

mount --bind /home/users /export/users

Step5

To allow this to stay after reboot, add the following to /etc/fstab (sudo nano /etc/fstab)

/home/users /export/users none bind 0 0

Step6

Navigate to /etc/idmapd.conf file and check the contents, should be as follows:

[Mapping]

Nobody-User = nobody
Nobody-Group = nogroup

Step7

To export the directory to the desired lan, the following lines must be used in /etc/exports. Run the following command:

sudo nano /etc/exports

-Copy this information into the file.

/export 192.168.1.0/24(rw,fsid=0,insecure,no_subtree_check,async)
/export/users 192.168.1.0/24(rw,nohide,insecure,no_subtree_check,async)

-You should replace '.1' with your own network variable.
What this does is allows all clients within the .1.0 to 254 range to have access.

Step8

These steps are for added security, it is known as portmap lockdown. They will lock out all client requests unless they are added manually to the hosts.allow file.
To do this, run the following in the /etc/hosts.deny (sudo nano /etc/hosts.deny)

rpcbind mountd nfsd statd lockd rquotad : ALL

Step9

This step is the allocation of ip addresses which will be allowed to access the server.
Edit the etc/hosts.allow file by running the following:

sudo nano /etc/hosts.allow

Put the following inside the file:

rpcbind mountd nfsd statd lockd rquotad : 127.0.0.1 192.168.1.150

What this does is allows both the server itself (127.0.0.1) and the client (192.168.1.150) to access it (change the ip address to your own client's ip address of node2).

Step10

Now restart the service in order to allow the changes to be committed.

Sudo /etc/init.d/nfs-kernel-server restart

CLIENT SIDE
(NODE2)

Step11)

Install the required packages for the client side to function:

sudo apt-get install nfs-common

Step12)

This step is where the export tree is mounted, it is all done in the one line command:

mount -t nfs -o proto=tcp,port=2049 192.168.1.11:/ /mnt

-Again, replace the ip address with the ip of your server. /mnt is the location of where it will be located.

Step13)

I mounted an exported subtree by using the following

mount -t nfs -o proto=tcp,port=2049 192.168.1.11:/export/shared /home/users

-This allowed the export/users file to be mounted, provisioning viewing and editing on both the client and server nodes.

Step14)

To prevent repetition and to allow it to occur on each boot, edit /etc/fstab

sudo nano /etc/fstab

-Add this to the file:

192.168.1.11:/ /mnt nfs auto 0 0

-Again, change the ip to your servers.

NFS Completed.

DNS

Introduction:

DNS, domain name service, is service that maps IP addresses and domain names to one another. This takes away the need to remember long ip addresses and instead allows you to use domain names.

For this report, we will be using BIND (Berkley Internet Naming Daemon). BIND as stated, an internet naming daemon.

ON SERVER SIDE,
NODE1.
(PRIMARY MASTER)

Step1

Login to node1 and enter the following:

sudo apt install bind9

Step2

At default, bind is setup as a caching server. To change this, edit the /etc/bind/named.conf.options.

Sudo nano /etc/bind/named.conf.options

Inside this file, uncomment #Forwarders, and enter the following:

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

-These are google's name servers.

Step3

Restart the server in order to allow configuration to be committed by running the following:

Sudo /etc/init.d/bind9 restart

Step4

In order to add a DNS zone to BIND9, turning BIND9 into a Primary Master server, configuration of the /etc/bind/named.conf.local file must happen.

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

Enter the following:

```
zone "example.lan" {
        type master;
```

file "/etc/bind/db.example.lan";

};

Step5)

Now we must create the db.example.lan file.

sudo cp /etc/bind/db.local /etc/bind/db.example.lan

Step6)

Edit /etc/bind/db.example.lan as follows:

Sudo nano /etc/bind/db.example.lan

Fill in as follows:

```
; BIND  data file for example.lan
;
; DO NOT EDIT THIS FILE - it is used for multiple zones.
; Instead, copy it, edit named.conf, and use that copy.
                                        ;
$TTL  604800
@       IN      SOA   example.lan root.example.lan (
                                      4         ; Serial
                                 604800         ; Refresh
                                  86400         ; Retry
                                 2419200        ; Expire
                                 604800 )  ; Negative Cache TTL
IN      A       192.168.1.11
                                        ;
@       IN      NS      ns.example.lan.
@       IN      A       192.168.1.11
@       IN      AAAA ::1
ns      IN      A       192.168.1.11
```

Step7

Restart the server in order to allow configuration to be committed by running the following:

Sudo /etc/init.d/bind9 restart

Step8

Edit /etc/bind/named.conf.local and add the following:

zone "1.168.192.in-addr.arpa" {
type master;
file "/etc/bind/db.192";
};

Step9

Create the db.192 file now by doing the following:

sudo cp /etc/bind/db.127 /etc/bind/db.192

Step10

Configure the file as follows:

```
;
; BIND reverse data file for 192.168.1. interface
;
$TTL  604800
@      IN      SOA   ns.example.lan root.example.lan (
                                                  2        ; Serial
                                             604800        ; Refresh
                                              86400        ; Retry
                                            2419200        ; Expire
                                             604800 )      ; Negative Cache TTL
;
@      IN      NS     ns.
10     IN      PTR    ns.example.lan
```

Step11

Again, restart services to commit changes:

Sudo /etc/init.d/bind9 restart

Step12

On the Primary Master server, the zone transfer needs to be allowed.

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

-Fill in as follows:

```
//
// Do any local configuration here
//
```

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

zone "example.lan" {
type master;
file "/etc/bind/db.example.lan";
allow-transfer { 192.168.1.150; };
};

zone "43.168.192.in-addr.arpa" {
type master;
file "/etc/bind/db.192";
allow-transfer { 192.168.1.150; };
};

Restart services again. *(Sudo /etc/init.d/bind9 restart).*

STEP 13*:*
Create symbolic links. This is done due to write permissions on the /etc/bind folder
-First off, navigate to the /var/cache/bind folder.

cd /var/cache/bind/
-Next, make a symbolic link by running the following (-s means symbolic):

sudo ln -s /etc/bind/db.example.lan
-Repeat for db.192
ln -s /etc/bind/db.192.

CLIENT SIDE,
ON NODE 2.
(SECONDARY MASTER)

Step14

Login to node2, the client, and install bind as before:

sudo apt install bind9

Step15

Edit /etc/bind/named.conf.local

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

Fill in file as follows:

```
//
// 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 "example.lan" {
        type slave;
        file "db.example.lan";
        masters { 192.168.1.11; };
        };

zone "43.168.192.in-addr.arpa" {
        type slave;
        file "db.192";
        masters { 192.168.1.11; };
        };
```

Restart services to commit changes. *(Sudo /etc/init.d/bind9 restart).*

DNS Completed.

FTP

Introduction:

FTP is a file transfer protocol which allows for downloading and uploading files between computers. FTP works on a client/server model. The server component is called an FTP daemon (vsftpd). FTP works by continuously listening for FTP requests from remote clients. When a request is received by the server, it manages the login and sets up the connection.
The daemon we will be using is VSFTPD, The first two letters of vsftpd stand for "very secure" and the program was built to have strongest protection against possible FTP vulnerabilities.

Step1

On your server (Node1), run the following command:

sudo apt-get install vsftpd

Step2

After downloading the packages, the config file must be changed. Run the following:

sudo nano /etc/vsftpd.conf

Step3

The following must be changed in order for FTP to work. Uncomment and have the following exactly as follows:

anonymous_enable=NO

local_enable=YES

write_enable=YES

chroot_local_user=YES

-This allows only identified users to access.

-The next three comments ensure all the local users will be constrained within their chroot and will be denied access to any other part of the server.
-Save and exit (ctrl + x).

Step4

Create a new directory within the user's home directory:

mkdir /home/$USERNAME/FTP

-$USERNAME replace with your username.

Step5

Change the ownership of that file to root:

chown root:root /home/$USERNAME

-$USERNAME replace with your username.

Step6

Restart the daemon:

sudo service vsftpd restart

Step7

Download filezilla from a trusted source, then once downloaded and installed, enter the host (192.168.1.11), username (network), password(*) then connect.

-From here then you can read write execute files within the home directory of the server.

FTP Completed.

SSH

Introduction:

SSH ("Secure Shell") is a protocol for securely accessing one computer from another. With SSH, you must install an SSH client on the computer you connect from, and an SSH server on the computer you connect to. (Client on node2) (Server on Node1).

Step1

sudo apt install openssh-client

Step2

sudo apt install openssh-server

Step3

Copy the /etc/ssh/sshd_config file and protect it from writing with the following:

sudo cp /etc/ssh/sshd_config /etc/ssh/sshd_config.original
sudo chmod a-w /etc/ssh/sshd_config.original

Step4

CONFIG THIS FILE:

/etc/ssh/sshd_config

Step5

Change port to more desirable port such as 2222
(when ssh'ing, use ssh - p 2222)

To have sshd allow public key-based login credentials uncomment + add the following:

PubkeyAuthentication yes

To make your OpenSSH server display the contents of the /etc/issue.net file as a pre-login banner, simply add or modify the line:

Banner /etc/issue.net

Step6
Restart the server

sudo systemctl restart sshd.service

Step7
Generate the keys:

ssh-keygen -t rsa

Step8
Copy the id_rsa.pub file to the remote host and append it to ~/.ssh/authorized_keys:

ssh-copy-id username@remotehost

Step9
Change the file permissions:

chmod 600 .ssh/authorized_keys

Step10
SSH to the host without the need for a password.

SSH Completed.

Declaration

I hereby certify that the material, which is submitted in this assignment, is entirely my own work and has not been submitted for any academic assessment other than as part fulfilment of the assessment procedures for the program of Bachelor of Science in Computer Science (infrastructure) (Bsc(Hons)) (DT211C/4).

Signed: .....

Date: .....