CONTENTS:

APACHE SERVER CONFIGURATION

INTRODUCTION

- The Apache HTTP server is a software (or program) that runs in the background under an appropriate operating system, which supports multi-tasking, and provides services to other applications that connect to it, such as client web browsers. I
- t was first developed to work with Linux/Unix operating systems, but was later adapted to work under other systems, including Windows and Mac.
- The Apache binary running under UNIX is called *HTTPd* (short for HTTP daemon), and under win32 is called *Apache.exe*.

INSTALLATION

- To install the APACHE Server,type the following command in the terminal:
    - **sudo apt-get install apache2**
    - All the components of the APACHE server will be installed on running the above command.
    - On installation,the APACHE Server goes in running state.So if we open the web browser and type localhost or 127.0.0.1 in the address bar then it must display a predefined file named index.html
    - To start/stop/restart the APACHE server we can type following commands:

sudo /etc/init.d/apache2 start

sudo /etc/init.d/apache2 stop

sudo /etc/init.d/apache2 restart

- To see the status of APACHE Server type the following command:

s**udo /etc/init.d/apache2 status**

CONFIGURATION

1.To check the status of the ports

Type the following command in the terminal: **netstat -a | more**

- It will display both the listening and non-listening sockets.
- When the APACHE server is running,it must Listen to the HTTP port or port 80.
- To see the numeric address ,type **netstat -an | more**

2. To change the default home page of the local host

Type the following command in the terminal: **sudo gedit /var/www/index.html** and press Enter key.

- This is used to edit the index.html file in the www folder of var folder.Here we can edit this file which gets displayed when we type localhost or 127.0.0.1 in the address bar of web browser.

- Now add the following lines of code(code depends on how the user want to display the page):

```
<html>
<head>
<title>RITESH HOMEPAGE </title>
</head>
<body><h1>Welcome to my webpage!</h1>
<p>This web page is under construction.</p>
</body></html>
```

Save the file and close the gedit window.

- Now if we type localhost or 127.0.0.1 in the address bar of web browser(when APACHE server is in running state) then the page is displayed with title as ABCD HOMEPAGE which contains a heading *Welcome to my webpage!* and a single line paragraph consisting of This web page is under construction.

3. To change the ports assigned to the web services

Type the following command in the terminal to open the port configuration file :

s**udo gedit /etc/apache2/ports.conf** and press Enter key.

- This file specifies the information about the ports associated with the web services.

- Here you can change the port to which APACHE Server is listening to. By default NameVirtualHost *:80

```
Listen 80
```

- Save the file and close the gedit window.

4. To create new hosts

Now open the Hosts file in the etc folder by typing the following command in the terminal:**sudo gedit /etc/hosts**

- Now add the following line of code:

- 127.0.1.2 apacheserver
- Save the file and close the gedit window.

5. To give a server name to the new host

Type the following command in the terminal: s**udo gedit /etc/apache2/sites-available/default**

and press Enter key.

- Enter the following code: ServerName apacheserver**.**
- Save the file and close the gedit window.

6. To check for the connectivity to the new host

Now type the following command to check if it is able to ping the domain-name

ping apacheserver

DHCP Server Configuration

INTRODUCTION

DHCP stands for Dynamic Host Configuration Protocol. A DHCP server is that server in the network that allocates IP addresses to all the clients. This allocation of IP addresses can be done in two ways – dynamic or random allocation, or fixed allocation using hardware addresses of a particular host.

When dynamic allocation is used, a range of IP addresses is specified and the server assigns any random address from this pool to a client which requests for an IP address. On the other hand, suppose we wish to assign a fixed IP address to a particular client, say a network printer, then we can do this by assigning a static IP address based on the Hardware (MAC) address of the device. In this case, we need to explicitly mention the MAC address of the device so that every time it requests the DHCP server for an address, the server will assign the same address.

INSTALLATION

For installing the DHCP Server on a Ubuntu host machine, open the terminal and type the following command

```
sudo apt-get install isc-dhcp-server
```

Press y and let the installation continue.

CONFIGURATION

Checking the interfaces

First of all, we need to check the interfaces available on our system and chose the interface we wish to use for the DHCP server to listen. The available interfaces can be seen by typing the command 'ifconfig' in the terminal.

Assign a static address to the interface

After choosing the interface on which the server is to be configured, we assign a static IP address to that particular interface. This can be done as follows –

```
sudo vim /etc/network/interfaces
```

This command will open the interfaces file. We need to add the following lines to the file. The chosen interface in this case is the Ethernet port i.e eth0

```
auto eth0
iface eth0 inet static
address 192.168.1.1
netmask 255.255.255.0
gateway 192.168.1.254
```

Here, the netmask is the mask used by the network that the server is configured for. Gateway is the address of the default gateway of the network through which it is connected to an external network like the internet.

Save this file and use 'ifconfig' to verify that the required address is assigned.

Direct DHCP server to listen to eth0

After assigning a static IP address to the interface of choice, the DHCP server should be configured to listen to that interface (eth0 in this case). For this, the configuration file named "isc-dhcp-server" should be edited. The steps are as follows –

```
sudo vim /etc/default/isc-dhcp-server
```

This will open a file with a pre written description. The last line of this file says INTERFACES=""

We need to specify the interface name in the quotes. The server can also be configured to listen to multiple interfaces, in that case, we specify the interface names in the quotes separated by a space. Now, the last line will be : INTERFACES="eth0"

Save this file.

·Configure the config file dhcpd.conf

The next step is to enter the configuration for the required network settings in the dhcpd.config file. This file can be accessed by

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

The file is a sample configuration file and also explains the various settings through inline comments. Comments start with #. The configuration used in this case are as follows –

```
#custom configuration start#
option domain-name "dineshd.net";
option domain-name-servers 192.168.1.3;
option subnet-mask 255.255.255.0;
default-lease-time 600;
max-lease-time 7200;
authoritative;
subnet 192.168.1.0 netmask 255.255.255.0 {
range 192.168.1.10 192.168.1.200;
option routers 192.168.1.100 192.168.1.254;
option subnet-mask 255.255.255.0;
option broadcast-address 192.168.1.255;
option domain-name-servers 192.168.1.3;
}
#custom configuration end#
```

After editing the configuration file, save it and exit.

·Start the DHCP server

The DHCP server can be run by typing the following command in the terminal

```
sudo service isc-dhcp-server start
```

In case, the server is already running, it must be restarted by replacing “start” with “restart”.

DNS SERVER USING BIND

INTRODUCTION

DNS Stands for Domain Name Service.On the Internet, the Domain Name Service (DNS) stores and associates many types of information with domain names; most importantly, it translates domain names (computer hostnames) to IP addresses. It also lists mail exchange servers accepting e-mail for each domain.

Step 1: Install Ubuntu or use your WORKING installation.

Step 2: Install bind 9:

```
sudo apt-get install bind9
```

Step 3: Configure the main Bind files.

Usually, if you install Bind from the source code, you will have to edit the file *named.conf*. However, Ubuntu provides you with a pre-configured Bind, so we will edit another file:

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

This is where we will insert our zones. By the way, a zone is a domain name that is referenced in the DNS server.
Insert this in the *named.conf.local* file:

```
# This is the zone definition. replace example.com with your domain name
zone "example.com" {
        type master;
        file "/etc/bind/zones/example.com.db";
        };

# This is the zone definition for reverse DNS. replace 0.168.192 with your network address in
reverse notation - e.g my network address is 192.168.0
zone "0.168.192.in-addr.arpa" {
     type master;
     file "/etc/bind/zones/rev.0.168.192.in-addr.arpa";
};
```

Ok, now, let's edit the options file:

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

We need to modify the forwarder. This is the DNS server to which your own DNS will forward the requests he cannot process.

```
Forwarders {
      # Replace the address below with the address of your provider's DNS server
      123.123.123.123;
};
```

Now, let's add the zone definition files (replace example.com with your domain name:

```
sudo mkdir /etc/bind/zones
sudo vi /etc/bind/zones/example.com.db
```

The zone definition file is where we will put all the addresses / machine names that our DNS server will know. You can take the following example:

```
example.com.        IN      SOA     ns1.example.com.  admin.example.com.  (
                                                2006081401
                                                28800
                                                3600
                                                604800
                                                38400
 )

example.com.        IN      NS              ns1.example.com.
example.com.        IN      MX      10      mta.example.com.

www             IN      A       192.168.0.2
mta             IN      A       192.168.0.3
ns1             IN      A       192.168.0.1
```

Now, let's create the reverse DNS zone file:

```
sudo vi /etc/bind/zones/rev.0.168.192.in-addr.arpa
```

Copy and paste the following text, modify as needed:

```
@ IN SOA ns1.example.com.  admin.example.com.  (
                        2006081401;
                        28800;
                        604800;
                        604800;
                        86400
)

                    IN    NS      ns1.example.com.
1                   IN    PTR     example.com
```

Ok, now you just need to restart bind:

```
sudo /etc/init.d/bind9 restart
```

We can now test the new DNS server.

Step 4: Modify the file *resolv.conf* with the following settings:

```
sudo vi /etc/resolv.conf
```

Enter the following:

```
search example.com
```

```
nameserver 192.168.0.1
```

Now, test your DNS:

```
dig example.com
```

SQUID PROXY SERVER

INTRODUCTION

Squid is a full-featured web proxy cache server application which provides proxy and cache services for Hyper Text Transport Protocol (HTTP), File Transfer Protocol (FTP), and other popular network protocols. Squid can implement caching and

proxying of Secure Sockets Layer (SSL) requests and caching of Domain Name Server (DNS) lookups, and perform transparent caching. Squid also supports a wide variety of caching protocols, such as Internet Cache Protocol, (ICP) the Hyper Text Caching Protocol, (HTCP) the Cache Array Routing Protocol (CARP), and the Web Cache Coordination Protocol. (WCCP)

The Squid proxy cache server is an excellent solution to a variety of proxy and caching server needs, and scales from the branch office to enterprise level networks while providing extensive, granular access control mechanisms and monitoring of critical parameters via the Simple Network Management Protocol (SNMP). When selecting a computer system for use as a dedicated Squid proxy, or caching servers, ensure your system is configured with a large amount of physical memory, as Squid maintains an in-memory cache for increased performance.

INSTALLATION

- To install the SQUID PROXY Server,type the following command in the terminal:
    - **sudo apt-get install squid squid-common**
    - All the components of the SQUID PROXY server will be installed on running the above command.
    - Now Start the SQUID PROXY server by typing the following command:

sudo service squid3 start

- we can also use this command to start SQUID PROXY Server: **sudo /usr/sbin/squid3**
- We can restart the SQUID PROXY server by typing the following command:

sudo service squid3 restart

- We can see the status of the SQUID PROXYserver by typing the following command:

sudo service squid3 status

- To kill the Squid proxy process,type the following command:

    sudo pkill -9 squid

CONFIGURATION STEPS

1.We should make the copy of the original configure file.To do this,Type the following command in the terminal: **sudo cp /etc/squid3/squid.config /etc/squid3/squid.conf.bak**

- It will create a copy of file squid.config as squid.conf.bak

2. Type the following command in the terminal: **sudo gedit /etc/squid3/squid.conf &** and press Enter key.

- This is used to open the SQUID Configuration file in the squid3 folder which is in the etc folder .
- Close the gedit window.

3. Open the web browser and go to EDIT->Preferences->Advanced tab->Network->Settings

- Under manual proxy configuration, enter HTTP proxy as 10.84.1.124(your IP address) and a default port number 3128(for Squid).Click on Ok button.
- Now if we open any site(www.yahoo.com) then it will block the access to that site with an error message "The requested URL could not be retreived".

4.Open the SQUID configuration file as given in step2.

- Now move the cursor to following line: TAG: visible_hostname and enter any hostname of your choice(arnavproxyserver).Save the file.Restart the squid proxy server.

- Now if we open any site(www.yahoo.com) then it will block the access to that site with the hostname given in the configuration file.

5.Open the SQUID configuration file as given in step2.

- Now uncomment the following line(line no 700):**acl localnet src 10.0.0.0/8**
- Now uncomment the following line(line no 845):**http_access allow localnet**

This will enable access to the local network and save the fileand then restart the squid proxy server.

- Now if we open any site(www.yahoo.com) then it will grant the access to that site.

6.Open the SQUID configuration file as given in step2.

- Now move the cursor above the following line: TAG: http_access and add following code:

```
acl  block_websites dstdomain .msn.com .yahoo.com

http_access deny block_websites
```

- Now the first line mentions the websites which will be blocked by the squid proxy server by defining an access list(named block_websites).
- The next line implements the deny access on the access list defined above.
- Save the file and then restart the squid proxy server.
- Now if we open any site from the above access list then it will block the access to that site.

OPEN SSH

INTRODUCTION

OpenSSH is a freely available version of the Secure Shell (SSH) protocol family of tools for remotely controlling a computer or transferring files between computers.OpenSSH provides a server daemon and client tools to facilitate secure, encrypted remote control and file transfer operations, effectively replacing the legacy tools.

The OpenSSH server component, **sshd**, listens continuously for client connections from any of the client tools. When a connection request occurs, **sshd** sets up the correct connection depending on the type of client tool connecting. For example, if the remote computer is connecting with the **ssh** client application, the OpenSSH server sets up a remote control session after authentication. If a remote user connects to an OpenSSH server with **scp**, the OpenSSH server daemon initiates a secure copy of files between the server and client after authentication. OpenSSH can use many authentication methods, including plain password, public key, and **Kerberos** tickets.

INSTALLATION

Installation of the OpenSSH client and server applications is simple. To install the OpenSSH client applications on your Ubuntu system, use this command at a terminal prompt:

```
sudo apt-get install openssh-client
```

To install the OpenSSH server application, and related support files, use this command at a terminal prompt:

```
sudo apt-get install openssh-server
```

The **openssh-server** package can also be selected to install during the Server Edition installation process.

CONFIGURATION

You may configure the default behavior of the OpenSSH server application, **sshd**, by editing the file `/etc/ssh/sshd_config`. For information about the configuration directives used in this file, you may view the appropriate manual page with the following command, issued at a terminal prompt:

```
man sshd_config
```

There are many directives in the **sshd** configuration file controlling such things as communication

settings and authentication modes. The following are examples of configuration directives that can be changed by editing the /etc/ssh/sshd_config file.

Prior to editing the configuration file, you should make a copy of the original file and protect it from writing so you will have the original settings as a reference and to reuse as necessary.

Copy the /etc/ssh/sshd_config file and protect it from writing with the following commands, issued at a terminal prompt:

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

The following are examples of configuration directives you may change:

To set your OpenSSH to listen on TCP port 2222 instead of the default TCP port 22, change the Port directive as such:

Port 2222

To have **sshd** allow public key-based login credentials, simply add or modify the line:

PubkeyAuthentication yes

In the /etc/ssh/sshd_config file, or if already present, ensure the line is not commented out.

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

In the /etc/ssh/sshd_config file.

After making changes to the /etc/ssh/sshd_config file, save the file, and restart the **sshd** server application to effect the changes using the following command at a terminal prompt:

```
sudo /etc/init.d/ssh restart
```

Many other configuration directives for **sshd** are available for changing the server application's behavior to fit your needs. Be advised, however, if your only method of access to a server is **ssh**, and you make a mistake in configuring **sshd** via the /etc/ssh/sshd_config file, you may find you are locked out of the server upon restarting it, or that the **sshd** server refuses to start due to an incorrect configuration directive, so be extra careful when editing this file on a remote server.

SSH KEYS

SSH *keys* allow authentication between two hosts without the need of a password. SSH key authentication uses two keys a *private* key and a *public* key.

To generate the keys, from a terminal prompt enter:

```
ssh-keygen -t dsa
```

This will generate the keys using a *DSA* authentication identity of the user. During the process you will be prompted for a password. Simply hit *Enter* when prompted to create the key.

By default the *public* key is saved in the file `~/.ssh/id_dsa.pub`, while `~/.ssh/id_dsa` is the *private* key. Now copy the `id_dsa.pub` file to the remote host and append it to `~/.ssh/authorized_keys` by entering:

```
ssh-copy-id username@remotehost
```

Finally, double check the permissions on the `authorized_keys` file, only the authenticated user should have read and write permissions. If the permissions are not correct change them by:

```
chmod 600 .ssh/authorized_keys
```

You should now be able to SSH to the host without being prompted for a password.