UNIT 13: NETWORKING IN LINUX

UNIT STRUCTURE

13.1 Learning Objectives

13.2 Introduction

13.3 Installation and Configuration of a Simple LAN

13.4 Installation and Configuration of Proxy Server (Squid)

13.5 Installation and Configuration of Webserver Apache

13.6 Installation and Configuration of a SSH Server and Client

13.7 Installation and Configuration of File Server (Samba)

13.8 Rules Governing IP address Classes and Netmasks

13.9 The Ping Command

13.10 The netstat Command

13.11 The Traceroute Command

13.12 The telnet Command

13.13 The /etc/services File

13.14 Port

13.15 Let Us Sum Up

13.16 Further Readings

13.17 Answers to Check Your Progress

13.18 Model Questions

13.1 LEARNING OBJECTIVES

After going through this unit, you will be able to:

- learn to install and configure various Linux services
- understand Network services rules and IP address
- use network commands to configure and trouble shoot
- understand the concept of port

13.2 INTRODUCTION

In the previous unit, we have learnt the login process of Linux. We discussed how to create, modify and remove users and groups. The system monitoring tools and how to navigate X windows, GNOME and KDE are

also discussed. In this unit, we will discuss the installation and configuration of various Linux services like proxy server, Apache web server etc.

13.3 INSTALLATION AND CONFIGURATION OF A SIMPLE LAN

Linux has very powerful and robust networking capabilities. Linux's networking setup is open to review and completely configurable. In Linux everything is transparent to user and no parameter is forced upon. The basic networking principles are same both in Windows and Linux. This chapter starts with an overview of networking, and then looks in more detail at Linux networking on a Local Area Network (LAN).

By default Linux distro install network capabilities so we don't need to install anything for simple LAN .

To configure a IPv4 network we need three fundamental pieces of numerical, for IPv6 we need some more information's but here we will only discuss about IPv4. Those three fundamental numerical pieces are:

IP Address: It is an unique number defining the access point to the network interface. It has the form: xxx.xxx.xxx.xxx, where "xxx" are three, or fewer, numbers between 0 and 255.This address can be manually assigned or can be automatically assigned by DHCP.

Gateway: It is the unique number assigned to the network interface through which it communicate with other networks. It has the general xxx.xxx.xxx.xxx format and takes a form that also works with the other two numbers.

Netmask: The non-unique number that defines the network itself. This number can be automatically generated but is sometimes requested by the method use to configure the interface. It also has the format xxx.xxx.xxx.xxx.

At installation time

Some of the installer ask users to provide network configuration during installation time itself , we can skip this part if not , then installer prompts with "Network & Host Name" , The installer has automatically

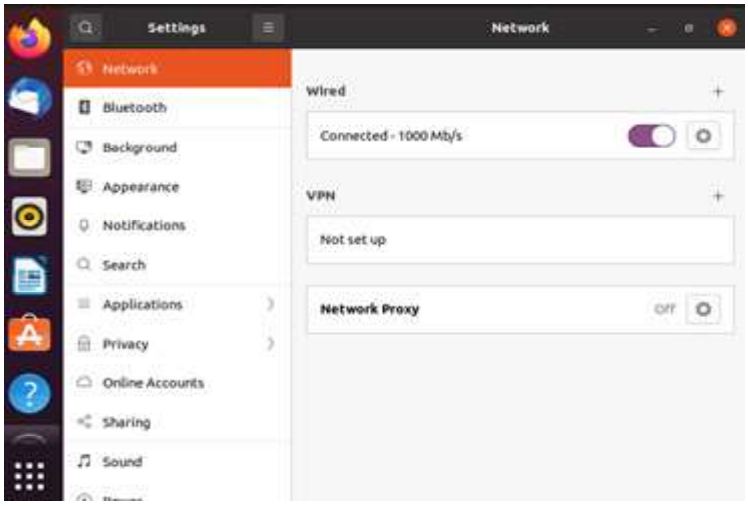
identified the one ethernet interface available and selected it for configuration. The text next to the selected interface is the system name of the interface, connection state but hardware address and speed are found from the system.

From GUI of the System

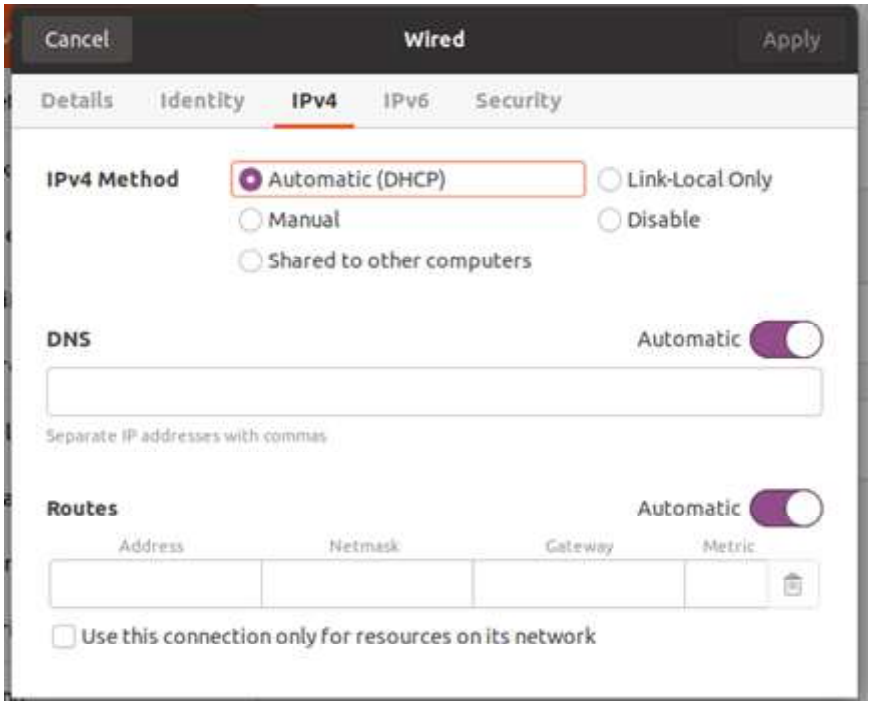
We can open the settings and go to network settings.



If we click on the setting it will open network settings it will show you Wired connection. There's the ON/OFF button and in this case if it's in the ON position. Choosing the gear button displays the network numbers that have been assigned (still DHCP) to the network interface.



If we click on IPv4, you can see familiar numbers we already discussed were given to your network interface by DHCP.





If we want to choose our network addresses manually, we can apply manual settings instead of DHCP

By Command Line

Suppose we are at totally new environment, and we need to setup an ethernet. While dealing with Live CD distros we can type these

commands for network setup in command line.

Setting up the hostname and DNS stuff

```
# Setup hostname
hostname mycomputerhostname
#  DNS setting
echo "nameserver 1.2.3.4" > /etc/resolv.conf
# Adding 1.2.3.4 as nameserver
```

When dealing with a live CD distro we don't care about /etc/ resolv.conf file but if we are running from our own HD then taking a backup will be a good idea

Now let's flush the eth card and setup the new ip address (let's assume we've a /dev/eth0 card to setup). Here a little bit about naming convention for network interfaces. It uses ethX. The "eth " for ethernet and "X" a single digit. No some new convention uses "enoX"

```
# Flush network address for the device /dev/eth0
ip addr flush dev eth0
# Add a new IP address to /dev/eth0 (use the form you want, that's just mine)
ip addr add 192.168.0.2/24 dev eth0
# adding 192.168.0.2, subnet mask (255.255.255.0) to my eth card
# Let's it up and running
ip link set eth0 up
# Now add a default route, 192.168.0.1
ip route add default via 192.168.0.1
```

The Ifconfig command

ifconfig (interface configurator) command can be used to initialize an interface, assign IP Address to interface and enable or disable interface. With this command, we can view IP Address and Hardware / MAC address assign to interface and also MTU (Maximum transmission unit) size.

```
saurabh@saurabh-VirtualBox:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::ff62:6fe1:63bf:606e  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:a8:d5:fc  txqueuelen 1000  (Ethernet)
        RX packets 4991  bytes 6769792 (6.7 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2177  bytes 168681 (168.6 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 17481  bytes 1250413 (1.2 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 17481  bytes 1250413 (1.2 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

To check details of a specific interface we can use ifconfig with interface (enp0S3) command. with -a option will display all available interface details if it is disabled also.

```
saurabh@saurabh-VirtualBox:~$ ifconfig enp0s3
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::ff62:6fe1:63bf:606e  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:a8:d5:fc  txqueuelen 1000  (Ethernet)
        RX packets 5061  bytes 6775838 (6.7 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2264  bytes 175659 (175.6 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

To set IP Address and Gateway in Linux we can use ifconfig command but this setting will be removed in case of a system reboot.

```
saurabh@saurabh-VirtualBox:~$ ifconfig enp0s3 192.168.20.10 netmask 255.255.255
.0
```

To enable or disable a specific Interface we can use this command like this .

Enable enp0s3 : ifup enp0s3

Disable enp0s3: ifdown enp0s3

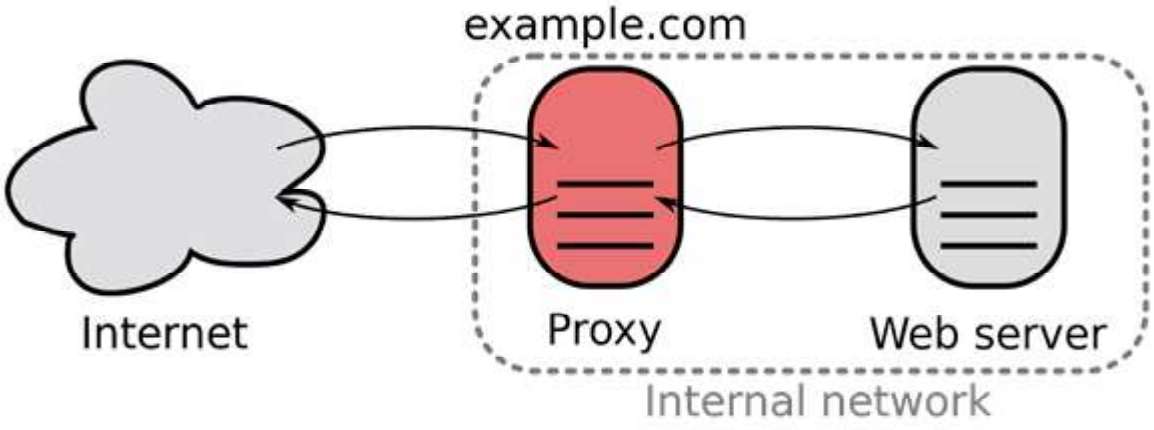
Setting MTU Size

By default MTU size is 1500. I we required size less than this we can replace XXXX with size.

ifconfig enp0s3 mtu XXX

13.4 INSTALLATION AND CONFIGURATION OF PROXY SERVER (SQUID)

Squid is a caching proxy for the Web supporting HTTP, HTTPS, FTP, and more. It reduces bandwidth and improves response times by caching and reusing frequently visited web pages. Squid has access controls and a great server accelerator. It runs on most available operating systems, including Windows and is licensed under the GNU GPL.



Squids reverse proxy is a service that sits between the Internet and the webserver (usually within a private network) that redirects inbound client requests to a server where data is stored for easier retrieval. If the caching server (proxy) does not have the cached data, it then forwards the request on to the web server where the data is stored. This type of caching allows for the collection of data and reproducing the original data values stored in a different location to provide for easier access.

A reverse proxy typically provides an additional layer of control to smooth the flow of inbound network traffic between your clients and the webserver.

Installation and configuration of: Proxy server (Squid) has been discussed in Unit 14 in details, here we have provided a basic idea about it.

Installation and configuration of: DNS server (BIND)

A DNS is a computer server that stores a IP addresses and their associated domain names. It serves to translate a domain name to its IP address, so that the computer knows which IP address to connect to for

the requested contents. For a human mind it is easier to remember a domain name as myexample.com, rather than a string of numbers, 104.20.73.209 (IP address). A DNS plays an important role in helping us to conveniently use the Internet as its translate domain name to particular IP address.

A DNS starts working as soon as user types a domain name in the address bar of a browser. It will search the Internet to find the correct IP address that is associated with the domain name. After successfully identifying the IP address, it then guides the user's browser to connect to it, which will then serve the requested website contents.

At first a DNS send a DNS query to several other DNS servers. A DNS is not just a single server responding to over billions of domain name requests, but instead it is distributed globally across a network of DNS. All the DNS servers work together to attend to the billions of domain names requests worldwide. It reduces the response for their requests.

When a website address is entered by a user in an Internet browser, a DNS query is initiated and a DNS server sends the query to several other DNS servers, each tasked with translating a different part of the domain name the user entered. There are mainly four servers which work together to translate the website address into a computer readable IP address, namely the DNS Resolver server, the root server, the Top-level Domain (TLD) server and the Authoritative Name Server. The DNS Resolver is the server who does most of the process in translating a domain name to an IP address. It receives the DNS query and in turn acts like a client to query the three other DNS servers mentioned above to translate the domain name.

It first queries the root server, and the root server responds to the query by returning the IP address of a TLD server (like .com, .net, .org etc.). A TLD server stores the information for its domains and will return the IP address of the Authoritative Name Server to the DNS Resolver. This is where the requested website is located. It then returns the actual IP address of the requested website to the DNS Resolver, which in turn response to the initial DNS query the actual IP address. Check out the illustration below to have a clear picture of the step-by-step processes of how a DNS work.

Installation and configuration of Mail server

If anyone wants to send or receive an email from its Linux system, it must have a mail sever. A mail server works according to SMTP (Simple Mail Transfer Protocol) works as well as other mail-related protocols, like Post Office Protocol (POP) and Internet Message Access Protocol (IMAP).

SMTP protocol defines how to send mail from one host to another; it is also system independent, which means the sender and receiver may have different operating systems. SMTP server can send ASCII text to another server by its default port 25.

Most Linux distribution today by default has two of the most common implementations of SMTP : sendmail and Postfix. Sendmail is a famous and free mail server, but it has a little complex design and less secure. The Postfix took mail server implementation is easy and has security . So here we will try Postfix server.

The mail service on any mail server has three components:

Mail user agent (MUA): this component that the user sees and interacts with like Thunderbird and Microsoft Outlook, these user agents are responsible for reading mail and allowing you to compose mail.

Mail transport agent (MTA): this component is responsible for getting the mail from one site to another like Sendmail and Postfix.

Mail delivery agent (MDA): this component is responsible for distributing received messages on the local machine to the appropriate user mailbox like postfix-maildrop and Procmail.

Setup Email server

Postfix is a default mail server in most of the Linux distribution. But to be sure first check is it present or not in your system. We can check that with following command

$ rpm -qa | grep postfix

```
saurabh@saurabh-VirtualBox:~$ rpm -qa | grep postfix
saurabh@saurabh-VirtualBox:~$
```

The rpm package may not be installed at your system so you may have to install rpm package before using it . If the output of this command returns nothing that means Postfix is not installed at your system.

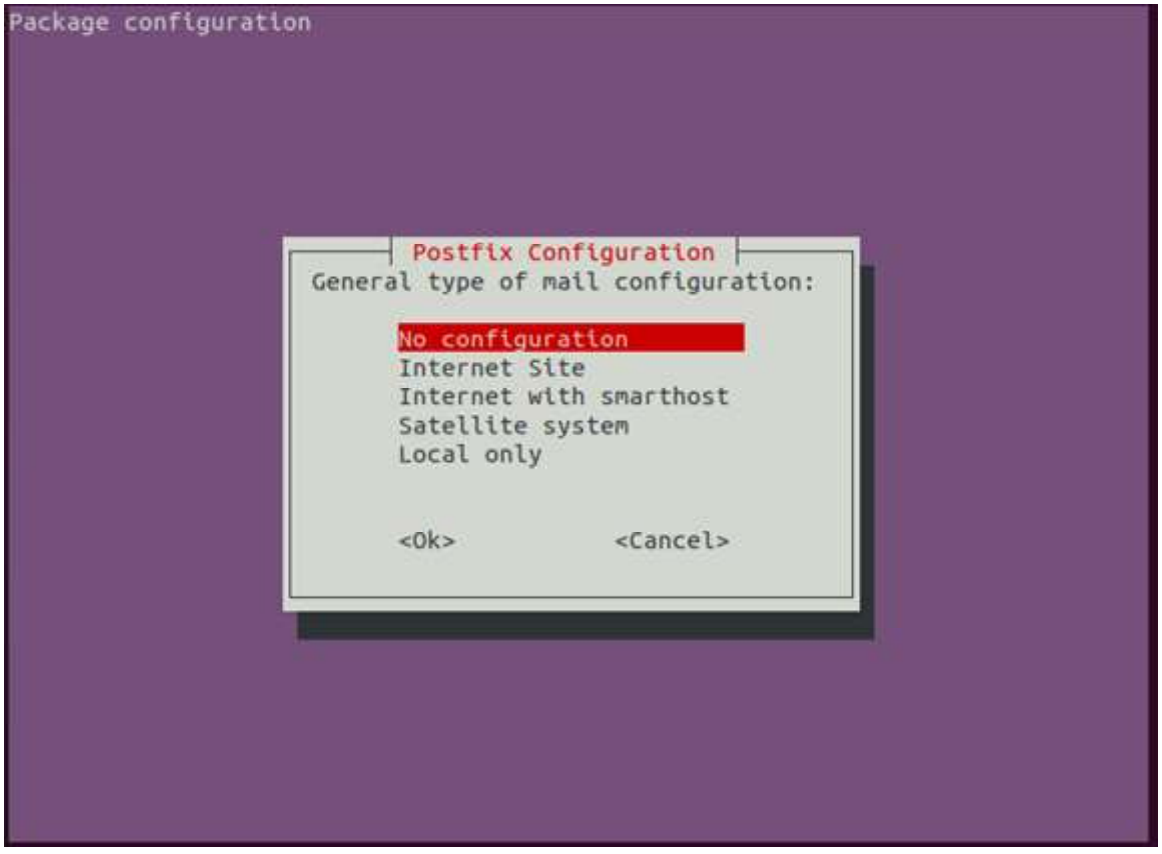
On Debian based distros like Ubuntu, we can install it with following command

$ sudo apt-get -y install postfix

```
saurabh@saurabh-VirtualBox:~$ sudo apt-get -y install postfix
[sudo] password for saurabh:
Reading package lists... Done
Building dependency tree
Reading state information... Done
```

It will ask you to select the type of configuration of the Postfix mail server during the installation process.

There are four choices No configuration, Internet site, Internet with smarthost, Satellite system, and Local only, we will choose No configuration option as we are going to configure it our own



Now start the postfix service and enable it on system start-up:

```
saurabh@saurabh-VirtualBox:/$ systemctl start postfix
saurabh@saurabh-VirtualBox:/$ systemctl enable postfix
```

After installation of the Postfix mail server, we need to configure it; we can find most of its configuration files under the /etc/postfix/ directory.

You can find the main configuration for Postfix mail server in /etc/postfix/main.cf file. It is always good practice to take a backup before doing any change.

This file contains a lot of options like:

myhostname

You can use this option for specifying the hostname of the mail server. This is the Internet hostname, which Postfix will receive emails on it. The hostnames could be like mail.example.com, smtp.example.com.

You incorporate the hostname this way:

myhostname = mail.example.com

```
89 # The myhostname parameter specifies the internet hostname of this
90 # mail system. The default is to use the fully-qualified domain name
91 # from gethostname(). $myhostname is used as a default value for many
92 # other configuration parameters.
93 #
94 #myhostname = host.domain.tld
95 #myhostname = virtual.domain.tld
96 myhostname = mail.example.com
```

mydomain

This option is the mail domain that pwe will be servicing, like example.com The syntax is like this: mydomain = example.com

```
 99 # The mydomain parameter specifies the local internet domain name.
100 # The default is to use $myhostname minus the first component.
101 # $mydomain is used as a default value for many other configuration
102 # parameters.
103 #
104 #mydomain = domain.tld
105 mydomain = example.com
106
```

myorigin

All emails sent from this mail server will look as though it came from this option. We can set this to $mydomain value.

myorigin = $mydomain

we can use any option value,but we neet to put a $ like $mydomain.

```
# SENDING MAIL
#
# The myorigin parameter specifies the domain that locally-posted
# mail appears to come from. The default is to append $myhostname,
# which is fine for small sites.  If you run a domain with multiple
# machines, you should (1) change this to $mydomain and (2) set up
# a domain-wide alias database that aliases each user to
# user@that.users.mailhost.
#
# For the sake of consistency between sender and recipient addresses,
# myorigin also specifies the default domain name that is appended
# to recipient addresses that have no @domain part.
#
# Debian GNU/Linux specific:  Specifying a file name will cause the
# first line of that file to be used as the name.  The Debian default
# is /etc/mailname.
#
#myorigin = /etc/mailname
#myorigin = $myhostname
myorigin = $mydomain
```

mydestination

This option lists the domains that the Postfix server uses for incoming emails.

It can take values like this:

mydestination = $myhostname, localhost.$mydomain, $mydomain, mail.$mydomain, www.$mydomain

```
# The local machine is always the final destination for mail addressed
# to user@[the.net.work.address] of an interface that the mail system
# receives mail on (see the inet_interfaces parameter).
#
# Specify a list of host or domain names, /file/name or type:table
# patterns, separated by commas and/or whitespace. A /file/name
# pattern is replaced by its contents; a type:table is matched when
# a name matches a lookup key (the right-hand side is ignored).
# Continue long lines by starting the next line with whitespace.
#
# See also below, section "REJECTING MAIL FOR UNKNOWN LOCAL USERS".
#
#mydestination = $myhostname, localhost.$mydomain, localhost
#mydestination = $myhostname, localhost.$mydomain, localhost, $mydomain
mydestination = $myhostname, localhost.$mydomain, localhost, $mydomain,
        mail.$mydomain, www.$mydomain, ftp.$mydomain
```

mail_spool_directory

There are two modes of delivery that Postfix mail server can use:

- Directly to a user's mailbox.
- To a central spool directory, this way, the mail will be in /var/spool/mail with a file for each user.

mail_spool_directory = /var/spool/mail

```
445 # The mail_spool_directory parameter specifies the directory where
446 # UNIX-style mailboxes are kept. The default setting depends on the
447 # system type.
448 #
449 #mail_spool_directory = /var/mail
450 mail_spool_directory = /var/spool/mail
451
```

mynetworks

This will let you arrange which servers can relay through your Postfix server.It should only take local addresses like local mail scripts on your server.If this isn't the case, then spammers can piggyback on your Linux mail server. That means your lovely shiny server will be doing the heavy lifting for some bad guys and it will also end up getting banned.

Here's the syntax for this option:

mynetworks = 127.0.0.0/8, 192.168.1.0/24

smtpd_banner

This one determines what message is sent after the client connects successfully. Consider changing the banner so it doesn't give away any potentially compromising information about your server.

inet_protocols

This option designates which IP protocol version is used for server connections.

inet_protocols = ipv4

When we change any of files used to configure Linux mail server for Postfix, you must reload the service, with this directive:

$ systemctl reload postfix

Of course, we all get distracted and typing things in can often result in mistakes, but you can track down any misspellings that might compromise your Linux mail server using this command:

$ postfix check

Checking the Mail Queue

Things like network failure (and many other reasons) can mean that the mail queue on your Linux email server can end up getting full, but you can check the Postfix mail queue with this command:

$ mailq

If that reveals that its full then you can flush the queue using this

command:

$ postfix flush

Look at it again and you should see that your Linux email server queue is clear.

Test Linux Mail Server

Once your configuration is done you need to test your Linux mail server.

The first thing to do is use a local mail user agent such as mailx or mail which is a symlink to mailx.

Send your first test to someone on the Linux mail server and if that works then send the next one to somewhere external.

$ echo "This is the body of the message" | mailx -s "Here we have a Subject" -r "for instance <small example@mydomain.com>" -a /path/to/attachment someone@mydomain.com

Then check if your Linux email server can pick up external mail.

If you run into any snags, have a peek at the logs. The Red Hat log file can be found in /var/log/maillog and for Debian versions in /var/log/mail.log, or wherever else the rsyslogd configuration specifies.

I would suggest you review the Linux syslog server for an in-depth clarification on logs and how to set up rsyslogd.

13.5 INSTALLATION AND CONFIGURATION OF WEBSERVER APACHE

A web server is a system which serves the requests via the HTTP protocol, if you request a file from the server, and it responds with the requested file, that's why web server are not only for web. Web servers also serves devices such as printers, routers, when you open your router configuration page, there is a web server behind it.

When we type something in a browser it sends a request to the server. The server takes the requested file or page from browser and maps it to the corresponding file from the server. The server sends the file back to the browser with some information such as the MIME type, the length of

the content, and other useful information. Sometimes the requested file is a static page like HTML pages or dynamic pages like PHP, Java, Perl, or any other server-side language.

There are many Linux web server implementations available for you to use:

- Apache server
- Nginx
- Lighttpd
- Apache Tomcat

But we will use Apache server here as it is stable, flexible and secure.

We can install apache using following command :

$ apt-get -y install apache2

The Apache web server service is called apache 2 in ubuntu. Apache works on http protocol which uses port 80. so we need to open port 80. In linux firewall like iptables, we can add a rule to open port 80.

$ iptables -I INPUT 1 -m state --state NEW -m tcp -p tcp --dport 80 -j ACCEPT

To start your service and enable it on boot:

$ systemctl start httpd

$ systemctl enable httpd

To check if your service is running or not, use the following command

$ systemctl status httpd

Configuring Apache web server

You can add files to Apache in the

/var/www/html

directory for top-level pages. Just remember to make sure that any files or directories placed in that directory are world-readable. The default index page is index.html.

The Apache configuration files are present in

/etc/apache2/apache2.conf

To discuss every options for conf file is beyond scope of this article, but we will discuss the most important options.

ServerRoot option

This option specifies the configuration folder for the Apache web server.. On Debian distros the ServerRoot option is /etc/apache2/.

ServerRoot /etc/apache2

Listen option

The Listen option specifies the port that the Apache web server will use to wait for incoming connections. The default value for this option is 80 for nonsecure connections and 443 for secured connections. If you have multiple IP addresses on your server, you can assign which IP should listen for connection using the Listen option.You can specify a different port other than 80, just make sure that it's not in use.You can run many HTTP servers on the same hardware every one on a unique port.When a server runs on a non-standard port such as port 8080, it will require the port number to be explicitly stated like this:

www.example.com:8080

Listen 80

ServerName option

This option specifies the hostname of the web server that appears to the visitors.

ServerName FQDN

DocumentRoot option

The DocumentRoot option defines the path that will contain your files that will be served.

The default path is /var/www/html.

DocumentRoot /var/www/html

MaxRequestWorkers option

This option sets the maximum number of concurrent connections that the server will receive.

LoadModule option

We use this option to load modules into the Apache web server.

There are a lot of Apache modules like these:

mod_cgid: This module runs CGI scripts using the Apache web server.

mod_ssl: Provides secure connections via SSL and TLS protocols.

mod_userdir: This module allows you to serve content from users' specific directories.

If you want to disable loading a specific module, you can use these commands:

$ a2enmod modulename

The a2enmod enables the module.

$ a2dismod modulename

This command to disable the module.

All these commands do is create a symlink under

/etc/apache2/mods-enabled

And if you use a2dismod, the symlink will be removed.If you enable or disable a module, you have to reload or restart the Apache web server.

LoadModule mod_cgid.so

Include option

This option allows you to include other configuration files.You can store all the configuration for different virtual domains, and Apache will include them at runtime.

Include filePath

UserDir option

This option specifies the directory that will contain the files that will be accessible via the web server. This directory is usually public_html and its location in the user's home directory.For example, if you have a user adam who wants to make his web content available via the Apache web server.

First, we make a public_html folder under his home directory.

Then set the permission for the public_html folder:

$ chmod 644 public_html

Now if we put an index.html file, it will be accessible via the browser like this:

http://YOURHOSTNAME/~adam

UserDir public_html

Alias option

This option specifies the location of the files that are outside the DocumentRoot location and need to be served by the Apache web server.Like you have files outside DocumentRoot, and you want them to be available to the visitors.

Alias URL_Path Actual_Path

ErrorLog option

This option specifies the error log file for the Apache web server.

ErrorLog /var/log/httpd/error_log

VirtualHost option

This option allows you to host multiple websites on the same server.

The idea is that the content is served based on the requested hostname.

To setup a virtual host for the host www.example.com. First, create a VirtualHost option in /etc/httpd/conf/httpd.conf file.

And specify the DocumentRoot and ServerName like this:

<VirtualHost www.example.com>
ServerAdmin admin@example.com
DocumentRoot /home/adam/public_html
ServerName www.example.com
ErrorLog /var/log/users/adam/error_log
</VirtualHost>

Keep in mind that the ServerName option must be resolvable via DNS.

These are the most used Apache options.

Virtual Host types

There are two types of virtual hosts that you can define in Apache web server:

- Name-based virtual hosts
- IP-based virtual hosts

The NameVirtualHost directive defines which addresses can be virtual hosts; the asterisk (*) means any name or address on this server.

You can write them like this:

```
NameVirtualHost *
<VirtualHost *>
 ServerName www.example.com
 DocumentRoot "/home/user1/public_html/"
</VirtualHost>
<VirtualHost *>
 ServerName www.example2.com
 DocumentRoot "/ home/user2/public_html/"
</VirtualHost>
```

If you have more than one IP address and you want to use an SSL certificate, the website must be on a dedicated IP address. You can write IP-based virtual hosts like this:

```
<VirtualHost 192.168.1.2>
 ServerName www.example.com
 DocumentRoot "/home/user1/public_html/"
</VirtualHost>
<VirtualHost 192.168.1.3>
 ServerName www.example2.com
 DocumentRoot "/ home/user2/public_html/"
</VirtualHost>
```

Apache process ownership

We know from the Linux process management that each process inherits its permissions from its parent process.This fact is true for all processes except for applications with the SETUID bit set; they inherit permissions from the owner, not the parent process. A good example is the /bin/su.If a normal user runs /bin/su program, it does not inherit the permission from adam, but it acts as a root user running it.Since Apache web server needs to bind port 80, and this needs root privileges.

After binding to port 80, Apache can run as a normal user and only read files that have permissions to read them.

User option

The User option specifies the user ID which the web server will use

to answer requests.

User www-data

Group option

The Group option specifies the group that the Apache web server will use to read files.

Group www-data

Security is very important for sites that use executable scripts such as CGI or PHP scripts.

The user you will use will have the permission to read and write the content of all sites on the server. But we want to ensure that only the members of a particular site can read their site only.

This is very important because if an attacker compromises your website, he will be able to read all files since the Apache user has permission to do that.

So how to solve this problem?

suEXEC support

A popular method is to use suEXEC. suEXEC is a program that runs with root permissions and makes CGI programs run as the user and group IDs of a specific user, not the Apache server user.

You can specify the user on each virtual host like this:

<VirtualHost www.example.com>

SuExecUserGroup adam adamGroup

</VirtualHost>

Just that simple.

Apache authentication

You may want to restrict some parts to specific visitors. It's like a password-protected directory.

In Apache, you can store the authentication information file called .htpasswd file.

You can use the htpasswd command to do that.

First, create the .htpasswd file using the htpasswd command:

$ htpasswd -c /home/adam/.htpassswd myuser

You need the -c option the first time you run htpasswd, but when you need to add more users, you shouldn't use -c because it will overwrite the file.

Then create a .htaccess file in the public_html folder and write the following:

```
<Location /vip>
 AuthName "test"
 AuthType Basic
 AuthUserFile /home/adam/.htpasswd
 Order deny,allow
 require valid-user
</Location>
```

AuthName is required, you can use any string you want.

AuthType Basic says that you're using an htpasswd style user file.

AuthUserFile points to the file that contains the generated password from the htpasswd command.

The Order line indicates that Apache must deny access by default, and only allow access for users specified in the htpasswd file.The require directive means any user in the .htpasswd file is allowed.

13.6 INSTALLATION AND CONFIGURATION OF A SSH SERVER AND CLIENT

SSH (Secure Shell framework) is a framework which is powerful and efficient system for accessing. It helps to establish connection and communication between two computing devices or routers remotely. It is a secured protocol; it encrypts all connections to ensure that all data or information does not land in the wrong hands. Thus, attackers are unable to eavesdrop or hijack details in the connection. It also offers an array of authentication modes and configuration options.

Setting Up Linux SSH on Ubuntu Linux

sshd or OpenSSH Daemon or server is an open source (free) ssh server which functions as the utility program for the ssh client. Ubuntu

distro do not come with sshd set up and running. So we need to install it .

First update your system

```
# sudo apt update

# sudo apt upgrade
```

Then install it

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

```
saurabh@saurabh-VirtualBox:/$ sudo apt-get install openssh-server
[sudo] password for saurabh:
Reading package lists... Done
Building dependency tree
Reading state information... Done
```

First, we will install the server than we will try with client. Once installed, verify the status of your ssh service using the below command.

```
# sudo systemctl status ssh
```

The above command should give you a result similar to the one in the image below. Note that the "Active (running)" is highlighted to indicate that SSH installation is successful and the protocol is already running in your systems.

```
saurabh@saurabh-VirtualBox:/$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset:
     Active: active (running) since Thu 2022-08-18 12:30:27 IST; 2min 25s ago
       Docs: man:sshd(8)
             man:sshd_config(5)
   Main PID: 6812 (sshd)
      Tasks: 1 (limit: 1082)
     Memory: 1.0M
     CGroup: /system.slice/ssh.service
             └─6812 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups

Aug 18 12:30:27 saurabh-VirtualBox systemd[1]: Starting OpenBSD Secure Shell s
Aug 18 12:30:27 saurabh-VirtualBox sshd[6812]: Server listening on 0.0.0.0 por
Aug 18 12:30:27 saurabh-VirtualBox sshd[6812]: Server listening on :: port 22.
Aug 18 12:30:27 saurabh-VirtualBox systemd[1]: Started OpenBSD Secure Shell se
```

If SSH does not return an active result, enable the SSH protocol in your systems by using following commands

```
# sudo systemctl enable ssh

# sudo systemctl start ssh
```

Ubuntu comes with the Uncomplicated Firewall (UFW), which manages network rules by default. An active UFW will potentially prevent

the connection of SSH to your server. You can change UFW settings to allow connection to the SSH server. You should also use this command t open port 22. These commands will help you configure UFW to allow connection;

```
# sudo ufw allow ssh
# sudo ufw enable
# sudo ufw status
```

To check ssh is working or not use following command : ssh user@localhost

You can get the user name by using command: whoami

```
saurabh@saurabh-VirtualBox:/$ ssh saurabh@localhost
The authenticity of host 'localhost (127.0.0.1)' can't be established.
ECDSA key fingerprint is SHA256:spE1egz0U5a/EbGmqmyg40biDFddVDSE+rRvAXCOZ9o.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
saurabh@localhost's password:
Welcome to Ubuntu 20.04.3 LTS (GNU/Linux 5.15.0-41-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

366 updates can be applied immediately.
259 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Your Hardware Enablement Stack (HWE) is supported until April 2025.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
```

Setting Up SSH Client on Linux

Now we can configure the SSH client. Notably, this allows connection to the remote server from your local machine. The procedure for setting up an SSH client is as follows;

The local machine should have the latest SSH version installed. But in cases where no SSH client is already installed, you can use the below command to install the client on Ubuntu;

```
# sudo apt install openssh-client
```

Connect to the Remote System

You can establish a connection to your remote system from the local computer. To achieve this, you should know the IP address of your local command and use this command.

```
# ssh username@IP_address
```

After installing and setting up SSH on both your server and client machines, you should be able to access your server remotely once you establish a connection.

13.7 INSTALLATION AND CONFIGURATION OF FILE SERVER (SAMBA)

Samba is a free and open-source software that allows files to be shared across Windows and Linux systems. It works on the (SMB) Server Message Block Protocol which is a client-server communication protocol that is used for sharing access to files, printers, serial ports, and other resources on a network. The (CIFS) Common Internet File System Protocol is a dialect of the SMB protocol. A collection of message bundles that describes a distinct variant of a protocol is called a dialect.While using Samba, we can share files and printing services, use authentication and authorization, name resolution, and service announcements between Linux/ Unix servers and Windows clients.

The basic tasks that can be performed using Samba are

- share a Linux drive with Windows machines.
- access an SMB share with Linux machines.
- share a Linux printer with Windows machines.
- share a Windows printer with Linux machines.
- set up a domain controller on a Unix/Linux server and integrate Windows clients to the Domain controller.

Installation and configuration part of Sama has been put as a Case Study for the learners.

13.8 RULES GOVERNING IP ADDRESS CLASSES AND NETMASKS

An IP (Internet Protocol) address is a numerical label assigned to the devices connected to a computer network that uses the IP for communication. IP address act as an identifier for a specific machine on a particular network. It also helps you to develop a virtual connection between a destination and a source. The IP address is also called IP number or internet address. It helps you to specify the technical format of the addressing and packets scheme. Most networks combine TCP with IP. An IP address consists of four numbers, each number contains one to three digits, with a single dot (.) separates each number or set of digits.

IP Address is divided into two parts:

- Prefix: The prefix part of IP address identifies the physical network to which the computer is attached. Prefix is also known as a network address.
- Suffix: The suffix part identifies the individual computer on the network. The suffix is also called the host address.

For example IP address 192.168.30.20 first 3 octave that is 192.168.30 is network address and last 20 part is host address

IP address works in an IP network like a postal address. For example, a postal address combines two addresses, address, or your area your house address.The address or your area is a group address of all houses that belong to a specific area. The house address is the unique address of your homes in that area. Here, your area is represented by a PIN code number.In this example, the network address comprises all hosts which belong to a specific network. The host address is the unique address of a particular host in that network.

Classful Addressing

Classful addressing is a network addressing the Internet's architecture from 1981 till Classless Inter-Domain Routing was introduced in 1993.This addressing method divides the IP address into five separate classes based on four address bits. Here, classes A, B, C offers addresses

for networks of three distinct network sizes. Class D is only used for multicast, and class E reserved exclusively for experimental purposes.

Let's see each of the network classes in detail:

Class A Network

This IP address class is used when there are a large number of hosts. In a Class A type of network, the first 8 bits (also called the first octet) identify the network, and the remaining have 24 bits for the host into that network.

An example of a Class A address is 102.168.212.226. Here, "102" helps you identify the network and 168.212.226 identify the host.

Class A addresses 127.0.0.0 to 127.255.255.255 cannot be used and is reserved for loopback and diagnostic functions.

Class B Network

In a B class IP address, the binary addresses start with 10. In this IP address, the class decimal number that can be between 128 to 191. The number 127 is reserved for loopback, which is used for internal testing on the local machine. The first 16 bits (known as two octets) help you identify the network. The other remaining 16 bits indicate the host within the network.

An example of Class B IP address is 168.212.226.204, where *168 212* identifies the network and *226.204* helps you identify the Hut network host.

Class C Network

Class C is a type of IP address that is used for the small network. In this class, three octets are used to indent the network. This IP ranges between 192 to 223.

In this type of network addressing method, the first two bits are set to be 1, and the third bit is set to 0, which makes the first 24 bits of the address them and the remaining bit as the host address. Mostly local area network used Class C IP address to connect with the network.

Example for a Class C IP address:

192.168.178.1

Class D Network

Class D addresses are only used for multicasting applications. Class

D is never used for regular networking operations. This class addresses the first three bits set to "1" and their fourth bit set to use for "0". Class D addresses are 32-bit network addresses. All the values within the range are used to identify multicast groups uniquely.

Therefore, there is no requirement to extract the host address from the IP address, so Class D does not have any subnet mask.

Example for a Class D IP address:

227.21.6.173

Class E Network

Class E IP address is defined by including the starting four network address bits as 1, which allows you two to incorporate addresses from 240.0.0.0 to 255.255.255.255. However, E class is reserved, and its usage is never defined. Therefore, many network implementations discard these addresses as undefined or illegal.

Example for a Class E IP address: 243.164.89.28

Network Mask

Mask - It is a 32-bit binary number that gives the network address in the address block when AND operation is bitwise applied on the mask and any IP address of the block.

The default mask in different classes are :

Class A - 255.0.0.0

Class B - 255.255.0.0

Class C - 255.255.255.0

Example : Given IP address 132.6.17.85 and default class B mask, find the beginning address (network address).

Solution : The default mask is 255.255.0.0, which means that the only the first 2 bytes are preserved and the other 2 bytes are set to 0. Therefore, the network address is 132.6.0.0.

Subnetting: Dividing a large block of addresses into several contiguous sub-blocks and assigning these sub-blocks to different smaller networks is called subnetting. It is a practice that is widely used when classless addressing is done.

Classless Addressing

To reduce the wastage of IP addresses in a block, we use sub-netting. What we do is that we use host id bits as net id bits of a classful IP address. We give the IP address and define the number of bits for mask along with it (usually followed by a '/' symbol), like, 192.168.1.1/28. Here, subnet mask is found by putting the given number of bits out of 32 as 1, like, in the given address, we need to put 28 out of 32 bits as 1 and the rest as 0, and so, the subnet mask would be 255.255.255.240.

Some values calculated in subnetting:

- Number of subnets : Given bits for mask - No. of bits in default mask
- Subnet address : AND result of subnet mask and the given IP address
- Broadcast address : By putting the host bits as 1 and retaining the network bits as in the IP address
- Number of hosts per subnet : 2(32 - Given bits for mask) - 2
- First Host ID : Subnet address + 1 (adding one to the binary representation of the subnet address)
- Last Host ID : Subnet address + Number of Hosts

Example : Given IP Address - 172.16.0.0/25, find the number of subnets and the number of hosts per subnet. Also, for the first subnet block, find the subnet address, first host ID, last host ID and broadcast address.

Solution : This is a class B address. So, no. of subnets = 2(25-16) = 29 = 512.

No. of hosts per subnet = 2(32-25) - 2 = 27 - 2 = 128 - 2 = 126

For the first subnet block, we have subnet address = 0.0, first host id = 0.1, last host id = 0.126 and broadcast address = 0.127

Rules for assigning Network ID:

The network ID will be assigned based on the below-given rules:

The network ID cannot start with 127 because 127 belongs to class A address and is reserved for internal loopback functions.

All bits of network ID set to 1 are reserved for use as an IP broadcast address and cannot be used.

All bits of network ID are set to 0. They are used to denote a particular host on the local network and should not be routed.

13.9 THE PING COMMAND

Ping is short for Packet Internet Groper used for checking the network connectivity among host/server and host. The ping command can take a URL or IP address as input and transfers the data packet to a specified address along with a "PING" message. Then, it will get a reply from the host/server. This time is known as "latency".

The ping command permits us to:

- Test our Internet connection.
- Check if the remote machine is active.
- Analyse when there are network problems such as high latency or dropped packages.

When we use "ping" command to a remote host, our machine starts transferring ICMP echo requests and wait for a response. An echo response for all requests has been transferred when the connection is established. The result includes the times it takes for all packets for reaching its target and return. Besides, the terminal prints the reply until interrupted which will be followed by some ping statistics

The basic syntax of the ping command contains ping followed by the hostname, a website name, or exact IP address.

Syntax: ping [options] hostname or IP address

```
saurabh@saurabh-VirtualBox:~$ ping google.com
PING google.com (142.250.183.110) 56(84) bytes of data.
64 bytes from bom12s13-in-f14.1e100.net (142.250.183.110): icmp_seq=1 ttl=116 t
ime=58.7 ms
64 bytes from bom12s13-in-f14.1e100.net (142.250.183.110): icmp_seq=4 ttl=116 t
ime=60.3 ms
64 bytes from bom12s13-in-f14.1e100.net (142.250.183.110): icmp_seq=7 ttl=116 t
ime=57.7 ms
```

This reply will goes until we manually stop it by pressing ctrl+c. The detail of the reply as follows

- **from:** The destination and its IP address. This IP address may be different for a website depending on geographical location.

- **icmp_seq=1:** The sequence number of each ICMP packet. Increases by one for every subsequent echo request.
- **ttl=52:** The Time to Live value from 1 to 255. It represents the number of networks hops a packet can take before a router discards it.
- **time=58.7 ms:** The time it took a packet to reach the destination and come back to the source. Expressed in milliseconds.

Ping has many other utilities also which can be explored with its different options like :

Option	Description
a	Use this option for a beep sound when the peer is reachable
b	Use this option to allow pinging a broadcast address
B	Use this option if you do not want to allow the ping to change the source address of the probe
c (count)	Use this option to set the number of times to send the ping request
d	Use this option to set the SO-DEBUG option on the socket being used
f	Use this option to flood the network by sending hundred or more packets per second
i	(interval) Use this option to specify an interval between successive packet transmissions. The default value of interval is 1 second
I	(interface address) Use this option to set source address to the specified interface address. This option is required when pinging IPv6 link local address. Its argument can be an IP address or name of the device.
l	(preload) Use this option to set the number of packets to send without waiting for a reply. For selecting a value more than 3, you need to be a super user.
n	Use this option to display network addresses as numbers rather than as hostnames
q	Use this option to display a quiet output. It means that only the summary is displayed at startup and finish time

T (ttl) Use this option to set the Time To Live

v Use this option for verbose output

V Use this option to display the version and exit

w (deadline) Use this option to specify a timeout, in seconds, before ping exits, regardless of how many packets have been sent or received.

W (timeout) Use this option to set the time(seconds) to wait for a response

Generally ping will not stop until we stop it forcefully. To make the ping command automatically stop after it sends a certain number of packets we can use -c options and a number. This sets the desired amount of ping requests, for example if we want to stop ping after two replies

ping -c 2 google.com

```
saurabh@saurabh-VirtualBox:~$ ping -c 2 google.com
PING google.com (142.250.199.142) 56(84) bytes of data.
64 bytes from bom07s36-in-f14.1e100.net (142.250.199.142): icmp_seq=1 ttl=57 ti
me=62.2 ms
64 bytes from bom07s36-in-f14.1e100.net (142.250.199.142): icmp_seq=2 ttl=57 ti
me=61.6 ms

--- google.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 61.603/61.914/62.226/0.311 ms
```

Suppose we do not want to fill our screen with information for every packet, then we can use the -q option (Quiet output) to display the summary only. For example if we want the summary of 10 packets

ping -c 10 -q google.com

```
saurabh@saurabh-VirtualBox:~$ ping -c 10 -q  google.com
PING google.com (142.250.199.174) 56(84) bytes of data.

--- google.com ping statistics ---
10 packets transmitted, 9 received, 10% packet loss, time 9043ms
rtt min/avg/max/mdev = 67.443/70.100/73.764/2.181 ms
```

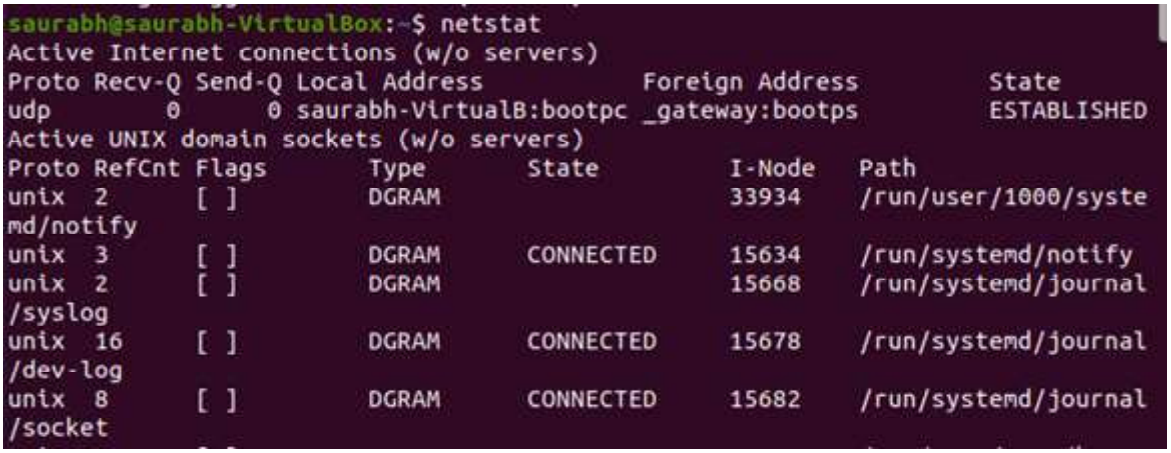
13.10 THE NETSTAT COMMAND

The netstat command abbreviated for network statistics used to detailed information about the networking information about your machine such as network connections, routing tables, interface statistics, masquerade connections, multicast memberships etc. By default Netstat

is not installed in Linux distro , we can install it using

sudo apt install net-tools

A simple execution of the netstat command alone to show a relatively simple list of all active TCP connections which, for each one, will show the local IP address (your computer), the foreign IP address (the other computer or network device), along with their respective port numbers, as well as the TCP state. There are different options available also with netstat



Option	Explanation
-a	This switch displays active TCP connections, TCP connections with the listening state, as well as UDP ports that are being listened to.
-b	This netstat switch is very similar to the **-o** switch listed below, but instead of displaying the PID, will display the process's actual file name. Using **-b** over **-o** might seem like it's saving you a step or two but using it can sometimes greatly extend the time it takes netstat to fully execute.
-e	Use this switch with the netstat command to show statistics about your network connection. This data includes bytes, unicast packets, non-unicast packets, discards, errors, and unknown protocols received and sent since the connection was established.
-f	The **-f** switch will force the netstat command to display the Fully Qualified Domain Name (FQDN) for each foreign IP addresses when possible.
-n	Use the **-n** switch to prevent netstat from attempting to determine host names for foreign IP addresses. Depending on your current network connections, using this switch could considerably reduce the time it takes for netstat to fully execute.

-o	A handy option for many troubleshooting tasks, the **-o** switch displays the process identifier (PID) associated with each displayed connection. See the example below for more about using **netstat -o**.
-p	Use the **-p** switch to show connections or statistics only for a particular *protocol*. You can not define more than one *protocol* at once, nor can you execute netstat with **-p** without defining a *protocol*.
protocol	When specifying a *protocol* with the **-p** option, you can use **tcp**, **udp**, **tcpv6**, or **udpv6**. If you use **-s** with **-p** to view statistics by protocol, you can use **icmp**, **ip**, **icmpv6**, or **ipv6** in addition to the first four I mentioned.
-r	Execute netstat with **-r** to show the IP routing table. This is the same as using the route command to execute **route print**.
-s	The **-s** option can be used with the netstat command to show detailed statistics by protocol. You can limit the statistics shown to a particular protocol by using the **-s** option and specifying that *protocol* but be sure to use **-s** before **-p** *protocol* when using the switches together.
-t	Use the **-t** switch to show the current TCP chimney offload state in place of the typically displayed TCP state.
-x	Use the **-x** option to show all NetworkDirect listeners, connections, and shared endpoints.
-y	The **-y** switch can be used to show the TCP connection template for all connection. You cannot use **-y** with any other netstat option.
time_interval	This is the time, in seconds, that you'd like the netstat command to re-execute automatically, stopping only when you use Ctrl-C to end the loop.

For example

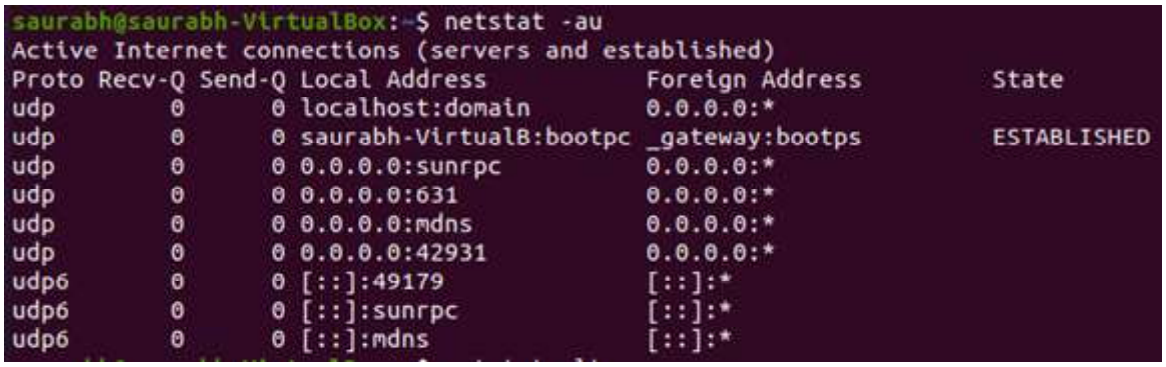
To List all tcp ports.

 # netstat -at

```
saurabh@saurabh-VirtualBox:~$ netstat -at
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 localhost:ipp           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:sunrpc          0.0.0.0:*               LISTEN
tcp        0      0 localhost:domain        0.0.0.0:*               LISTEN
tcp6       0      0 [::]:sunrpc             [::]:*                  LISTEN
tcp6       0      0 ip6-localhost:ipp       [::]:*                  LISTEN
```

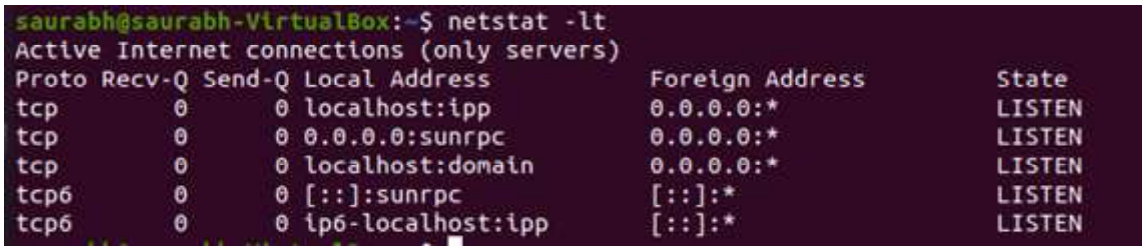
To List all udp ports.

 # netstat -au



List only listening TCP ports.

 # netstat -lt



13.11 THE TRACEROUTE COMMAND

Linux traceroute command is used as network troubleshooting utility that helps in determining the number of hops and packets traveling path required to reach a destination. The traceroute is not a default utility of the Linux system so we need to install it manually. To install it, execute the following commands:

sudo apt install traceroute

The basic syntax of traceroute is : traceroute [OPTION...] HOST . For example, to trace the route of the server 'google.com,' execute the below command:

traceroute google.com

This command will display hops, packets, and much other information on the given route. Its output has been shown bellow

```
saurabh@saurabh-VirtualBox:~$ traceroute google.com
traceroute to google.com (142.250.192.238), 30 hops max, 60 byte packets
 1  _gateway (10.0.2.2)  0.547 ms  0.515 ms  0.500 ms
 2  * * *
 3  * * *
 4  * * *
 5  * * *
 6  * * *
 7  * * *
 8  * * *
 9  * * *
10  * * *
11  * * *
12  * * *
13  * * *
14  * * *
```

- The first line is displaying the hostname and IP address (which has not been resolved, so send it to gateway address) that is to be reached, hops that are going to be attempted by the traceroute command, and the size of the packets to be sent.
- By default, it sends three packets for each host, so three response times are listed.
- The '*' symbol represents the packet loss.

The following command-line options are supported by the traceroute command:

-f, --first-hop=NUM: It is used to set the initial hop distance.

-g, --gateways=GATES: It is used to display a list of gateways for loose source routing.

-I, --icmp: It is specified to use ICMP ECHO as a probe.

-m, --max-hop=NUM: It is used to set maximal hop count, the default is 64.

-M, --type=METHOD: It specifies the METHOD (icmp or udp) for traceroute operations, the default method is udp.

-p, --port=PORT: It is defined to use destination PORT port, the default PORT is 33434.

-q, --tries=NUM: It is used to forward NUM probe packets per hop, the default is 3.

-- resolve-hostnames: It is used to resolve the hostnames.

-t, --tos=NUM: It is used to set the type of service (TOS) to NUM.

-w, --wait=NUM: It is used to wait in seconds for a response, the default is 3.

For example, if we want to trace the Route Using IPv6 than the '6' option is used to trace the route to a host network using the Ipv6 protocol. Consider the below command:

traceroute6 google.com

To set the Number of Queries per hop we use the '-q' option. Consider the below command:

traceroute -q 2 google.com

13.12 THE TELNET COMMAND

In Linux, the telnet command is used to establish the connections between different machines. This command allows us to manage the remote devices using the CLI (command-line interface). It uses TCP port 23 which is assigned to the telnet protocol. By default, telnet utility is not present in Linux. In Ubuntu and Debian based distributions it can be installed using the command:

$ sudo apt install telnetd -y

After the installation is completed, check the status of telnet protocol using the systemctl command:

$ sudo systemctl status inetd

The telnet protocol is installed and is in running status. We know that it uses port 23, so we will enable port 23 using the ufw command:

$ sudo ufw allow 23/tcp

Reload the ufw using the command:

$ sudo ufw reload

To connect to the devices we need the IP address of the particular machine and also make sure the telnet protocol is installed in that machine as well and port 23 is enabled. In our case, the IP address of the machine to which we want to connect is 192.168.18.135, we will run the command:

$ telnet host_ipr_adress

To connect to the devices we need the IP address of the particular machine and also make sure the telnet protocol is installed in that machine as well and port 23 is enabled. In our case, the IP address of the machine

to which we want to connect is 192.168.18.135, we will run the command:

$ telnet 192.168.10.10

After connecting it will prompt you to provide user name and password. after successfully login you can use its services.

13.13 THE / ETC/SERVICES FILE

UNIX operating systems store what's called a services file at /etc/ services. It stores information about numerous services that client applications might use on the computer. Within the file are the service name, port number, and protocol it uses, and any applicable aliases. /etc/ services file contains a list of network services and ports mapped to them. inetd or xinetd looks at these details so that it can call a particular program when a packet hits respective port and demand for service.

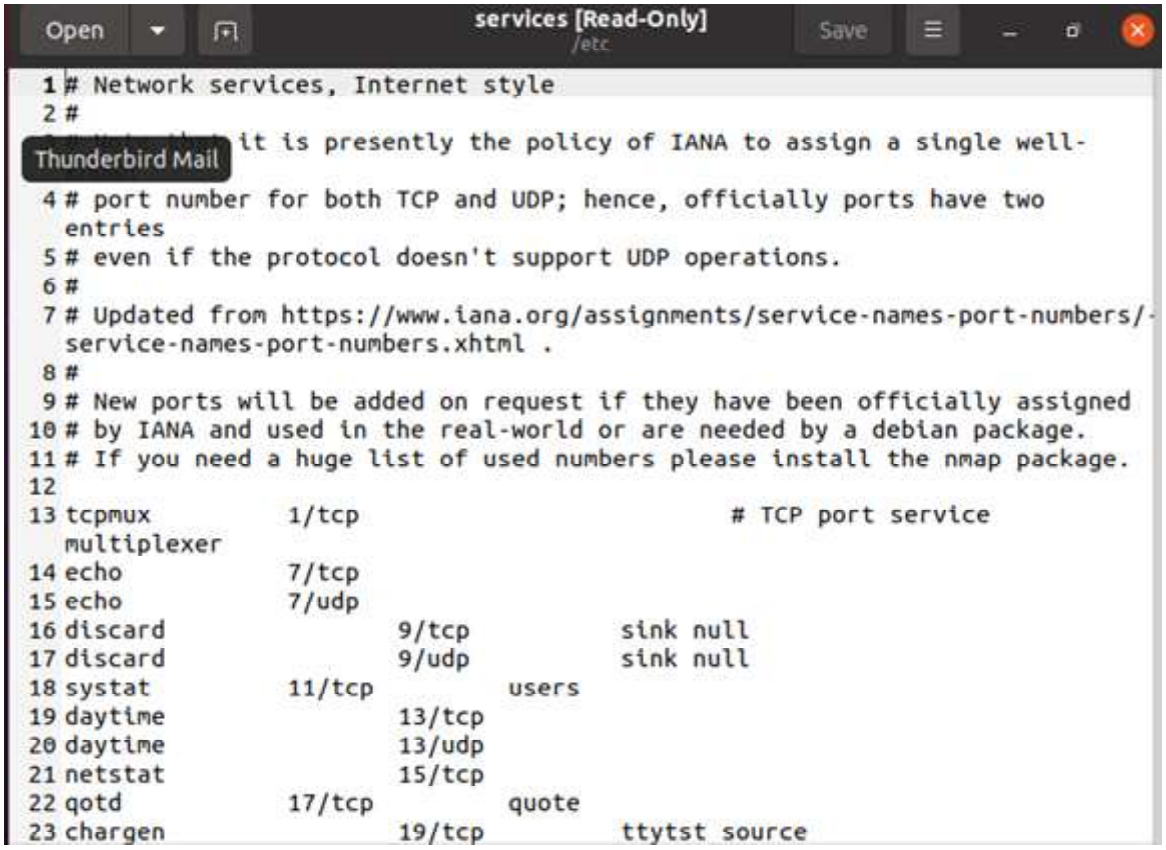
As a normal user, we can view this file since the file is world readable. To edit this file we need to have root privileges.

/etc/services file format

service-name port/protocol [aliases..] [#comment]

Last two fields are optional hence denoted in [] where -

- service-name is the name of the network service. e.g. telnet, FTP, SMTPetc.
- port/protocol is the port being used by that network service (numerical value) and protocol (TCP/UDP) used for communication by service.
- alias is an alternate name for service.
- a comment is a note or description you can add to service. Starts with # mark



13.14PORT

Port number is a 16-bit numerical value that ranges from 0 to 65535. Well-known port (0-1023), registered port (1024-49151), and dynamic port is three types of port number space. (49152-65535). These ports can be opened and used by software applications and operating system services to send and receive data over networks (LAN or WAN) that employ certain protocols (eg TCP, UDP).For example, we use 80 for HTTP-web-based plain-text surfing and 443 for HTTPS-web-based encrypted websites in our daily work. To conclude, a port is a logical form to identify system activities or various network services used to create local or network-based communications.

Port Number	Service name	Transport protocol	Description
7	Echo	TCP, UDP	Echo service

20	FTP-data	TCP, SCTP	File Transfer Protocol data transfer
21	FTP	TCP, UDP, SCTP	File Transfer Protocol (FTP) control connection
22	SSH-SCP	TCP, UDP, SCTP	Secure Shell, secure logins, file transfers (scp, sftp), and port forwarding
23	Telnet	TCP	Telnet protocol—unencrypted text communications
25	SMTP	TCP	Simple Mail Transfer Protocol, used for email routing between mail servers
53	DNS	TCP, UDP	Domain Name System name resolver
69	TFTP	UDP	Trivial File Transfer Protocol
80	HTTP	TCP, UDP, SCTP	Hypertext Transfer Protocol (HTTP) uses TCP in versions 1.x and 2. HTTP/3 uses QUIC, a transport protocol on top of UDP
88	Kerberos	TCP, UDP	Network authentication system
102	Iso-tsap	TCP	ISO Transport Service Access Point (TSAP) Class 0 protocol
110	POP3	TCP	Post Office Protocol, version 3 (POP3)
135	Microsoft EPMAP	TCP, UDP	Microsoft EPMAP (End Point Mapper), also known as DCE/RPC Locator service, used to remotely manage services including DHCP server, DNS server, and WINS. Also used by DCOM
137	NetBIOS-ns	TCP, UDP	NetBIOS Name Service, used for name registration and resolution
139	NetBIOS-ssn	TCP, UDP	NetBIOS Session Service
143	IMAP4	TCP, UDP	Internet Message Access Protocol (IMAP), management of electronic mail messages on a server



CHECK YOUR PROGRESS

Q 1: Fill in the blanks:

(a) _____ is an unique number defining the access point to the network interface.

(b) _____ is the unique number assigned to the network interface through which it communicate with other networks.

(c) The non-unique number that defines the network itself is termed as _____

(d) _____ reduces bandwidth and improves response times by caching and reusing frequently visited web pages.

(e) A _____ is a computer server that stores a IP addresses and their associated domain names.

(f) In a _____ type of network, the first 8 bits (also called the first octet) identify the network, and the remaining have 24 bits for the host into that network.



13.15 LET US SUM UP

- ifconfig (interface configurator) command can be used to initialize an interface, assign IP Address to interface and enable or disable interface.
- Squid is a caching proxy for the Web supporting HTTP, HTTPS, FTP, and more. It reduces bandwidth and improves response times by caching and reusing frequently visited web pages.
- A DNS is a computer server that stores a IP addresses and their associated domain names. It serves to translate a domain name to its IP address, so that the computer knows which IP address to connect to for the requested contents.
- Postfix is a default mail server in most of the Linux distribution.
- Samba is a free and open-source software that allows files to be shared across Windows and Linux systems.
- An IP (Internet Protocol) address is a numerical label assigned to the devices connected to a computer network that uses the IP for

communication. IP address act as an identifier for a specific machine on a particular network.

- Ping is short for Packet Internet Groper used for checking the network connectivity among host/server and host.
- The netstat command abbreviated for network statistics which is used to get the detailed information about the networking information about a machine such as network connections, routing tables, interface statistics, masquerade connections, multicast memberships etc.



13.16 FURTHER READINGS

1) Linux Administration Cookbook By Adam K. Dean
2) Linux Administration By Jason Cannon



13.17 ANSWERS TO CHECK YOUR PROGRESS

Ans to Q No 1: (a) ip address

Ans to Q No 1: (b) gateway

Ans to Q No 1: (c) netmask

Ans to Q No 1: (d) squid

Ans to Q No 1: (e) DNS

Ans to Q No 1: (f) Class A



13.18 MODEL QUESTIONS

Q 1: Configure IP address and netmask of a system using ifconfig .

Q 2: Why we use proxy sever like Squid?

Q 3: Explain how Apache server works.

Q 4: Explain different classes of IP address.

Q 5: Explain the procedure to connect SSH client to its server

Q 6: Explain at least 5 options available with ping command.

Q 7: What are the contents of /etc/services file?

*** ***** ***