**Lab 3a Static Networking**

Diagram

Description automatically generated

10.0.2.2/24

10.0.2.1/24

**IP Address Configuration**

The ifconfig command stands for interface configuration and is used to display network configuration information. The ifconfig command can also be used to modify network settings temporarily.

The ifconfig command can be used without options or arguments to display all interfaces on the network, or with options and an interface name as an argument:

ifconfig [INTERFACE] [OPTIONS]

To assign an IP address to an interface, execute the following command:

**sysadmin@localhost:~#**ifconfig ens37 192.168.1.1 netmask 255.255.255.0

The ifconfig command is becoming obsolete in some Linux distributions (deprecated) and is being replaced with a form of the ip command, it can almost be a one-stop shop for configuration and control of a system’s networking. The format for the ip command is as follows:

ip [OPTIONS] OBJECT COMMAND

**Refer to the cheat sheet for commands example**

**The route Command**

Recall that a router (or gateway) is a machine that allows hosts from one network to communicate with another network. To view a table that describes where network packages are sent, use the route command:

**root@localhost:~#** route

Kernel IP routing table

Destination Gateway Genmask Flags Metric Ref Use Iface

192.168.1.0 \* 255.255.255.0 U 0 0 0 eth0

default 192.168.1.1 0.0.0.0 UG 0 0 0 eth0

The first highlighted line in the preceding example indicates that any network package sent to a machine in the 192.168.1 network is not sent to a gateway machine (the \* indicates *no gateway*). The second highlighted line indicates that all other network packets are sent to the host with the IP address of 192.168.1.1 (the router).

Some users prefer to display this information with numeric data only, by using the -n option to the route command. For example, look at the following and focus on where the output used to display default:

**root@localhost:~#** route -n

Kernel IP routing table

Destination Gateway Genmask Flags Metric Ref Use Ifa

192.168.1.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0

0.0.0.0 192.168.1.1 0.0.0.0 UG 0 0 0 eth0

The 0.0.0.0 refers to *all other machines*, and is the same as *default*.

The fields in the output are as follows:

| **Option** | **Meaning** |
| --- | --- |
| Destination | Destination host or network |
| Gateway | IP address of the gateway  0.0.0.0 means “no route, broadcast on this network” |
| Genmask | Subnet mask for the destination network  Set to 0.0.0.0 for the default route |
| Flags | Values include C (cache entry), G (use gateway) and U (up) |
| Metric | Number of hops to the target, used as a measure of distance |
| Ref | Number of references to this route; this is not used in Linux |
| Use | Count of lookups done for this route |
| Iface | Interface to use for sending packets for this route |

To add this route to the eth0 interface, the administrator could execute the following sudo command (or as root):

**sysadmin@localhost:~$** sudo route add -net 192.56.78.0 netmask 255.255.255.0 gw 192.168.1.1 dev eth0

To add a default gateway, execute the following command:

**sysadmin@localhost:~$** sudo route add default gw 192.168.1.1

To delete a route from the routing table, an administrator can execute a command like the one that added it, except using del instead of add. For example, to delete the route and default gateway that was added earlier, an administrator could execute the following sudo commands (or as root):

**sysadmin@localhost:~$** sudo route del -net 192.56.78.0 netmask 255.255.255.0 gw 192.168.1.1 dev eth0

**sysadmin@localhost:~$** sudo route del default gw 192.168.1.1

The route command is becoming obsolete in some Linux distributions (deprecated) and is being replaced with a form of the ip command, specifically ip route show. Note that the same information highlighted above can also be found using this command:

**root@localhost:~#** ip route show

default via 192.168.1.254 dev eth0 proto static 192.168.1.0/24 dev eth0 proto kernel scope link src 192.1

## The netstat Command

The netstat command is a powerful tool that provides a large amount of network information. It can be used to display information about network connections as well as display the routing table similar to the route command.

To use the netstat command to display routing information, use the -r option:

**root@localhost:~#** netstat -r

Kernel IP routing table

Destination Gateway Genmask Flags MSS Window irtt Iface

192.168.1.0 \* 255.255.255.0 U 0 0 0 eth0

default 192.168.1.1 0.0.0.0 UG 0 0 0 eth0

The netstat command is also commonly used to display open ports. A port is a unique number that is associated with a service provided by a host. If the port is open, then the service is available for other hosts.

For example, you can log into a host from another host using a service called SSH. The SSH service is assigned port #22. So, if port #22 is open, then the service is available to other hosts.

It is important to note that the host also needs to have the services running itself; this means that the service (in this case the ssh daemon) that allows remote users to log in needs to be started (which it typically is, for most Linux distributions).

To see a list of all currently open ports, use the following command:

**root@localhost:~#** netstat -tln

Active Internet connections (only servers) Proto Recv-Q Send-Q Local Address Foreign Address State tcp 0 0 192.168.1.2:53 0.0.0.0:\* LISTEN

tcp 0 0 127.0.0.1:53 0.0.0.0:\* LISTEN

tcp 0 0 0.0.0.0:22 0.0.0.0:\* LISTEN

tcp 0 0 127.0.0.1:953 0.0.0.0:\* LISTEN

tcp6 0 0 :::53 :::\* LISTEN

tcp6 0 0 :::22 :::\* LISTEN

As you can see from the output above, port #22 is listening, which means it is open.

In the previous example, -t stands for TCP (recall this protocol from earlier in this chapter), -l stands for listening (which ports are listening) and -n stands for show numbers, not names.

Sometimes showing the names can be more useful. This can be achieved by dropping the -n option:

**root@localhost:~#** netstat -tl

Active Internet connections (only servers)

Proto Recv-Q Send-Q Local Address Foreign Address State tcp 0 0 cserver.example.:domain \*:\* LISTEN

tcp 0 0 localhost:domain \*:\* LISTEN

tcp 0 0 \*:ssh \*:\* LISTEN

tcp 0 0 localhost:953 \*:\* LISTEN

tcp6 0 0 [::]:domain [::]:\* LISTEN

tcp6 0 0 [::]:ssh [::]:\* LISTEN

tcp6 0 0 localhost:953 [::]:\* LISTEN

**Primary IPv4 Configuration File**

On a CentOS system, the primary configuration file for an IPv4 network interface is the /etc/sysconfig/network-scripts/ifcfg-ensxx file. The following shows what this file looks like when configured for a static IP address. (On a legacy Red Hat-derived system, the /etc/sysconfig/network file contains host and routing details for all configured network interfaces.)

**root@localhost:~#** cat /etc/sysconfig/network-scripts/ifcfg-eth0

DEVICE="ens33"

BOOTPROTO=none

NM\_CONTROLLED="yes"

ONBOOT=yes

TYPE="Ethernet"

UUID="98cf38bf-d91c-49b3-bb1b-f48ae7f2d3b5"

DEFROUTE=yes

IPV4 \_FAILURE\_FATAL=yes

IPV6INOT=no

NAME="ens33"

IPADDR=192.168.1.1

PREFIX=24

#NETMASK=255.255.255.0

GATEWAY=192.168.1.1

DNS1=192.168.1.2

HWADDR=00:50:56:90:18:18

LAST\_CONNECT=1376319928

If the device were configured to be a DHCP client, the BOOTPROTO value would be set to dhcp, and the IPADDR, GATEWAY and DNS1 values would not be set.

The widely accepted method of making changes to a network interface is to take the interface down using a command such as ifdown ens33, make the desired changes to the configuration file, and then bring the interface back up and into service with a command such as ifup ens33.

**Setting the Hostname**

The *hostname* is used to identify the system by applications such as web servers. On Debian-derived and modern Red Hat-derived systems, the /etc/hostname file contains this information, while legacy Red Hat-derived systems store this information in the /etc/sysconfig/network file. This file is read at boot time to set the hostname.

The hostname command is used to set and view the system’s host and domain name. It is the system administrator’s responsibility to assign an appropriate hostname. It cannot be longer than 64 characters and can contain alphanumeric [a-z] [0-9], period . characters, and hyphen - characters only.

To view the currently assigned hostname of the system, execute the hostname or *short name cut at the first dot* hostname -s, or *full domain name* hostname -f command:

**sysadmin@localhost:~$** hostname -s

localhost

The name displayed above is returned by the gethostname() application programming interface (API).

To view the fully-qualified domain name, execute the following command:

**sysadmin@localhost:~$** hostname -f

test.example.com

To set the hostname of the system, the root user can execute the following command:

**root@localhost:~#** hostname example.com

**root@localhost:~#** hostname

example.com

Note that setting the hostname using the hostname command results in a change that is only persistent until the next system boot.

The /etc/hosts file is used for mapping hostnames with IP addresses. It is a flat-file with one record on each line. The format of the file is:

*IP Address Host Name Alias*

A sample /etc/hosts file will look like the following:

127.0.0.1 localhost

192.168.4.8 apps.sample.com apps

192.168.4.12 vm1.sample.com vm1

The Alias field is used for mapping short names or labels to a host.

The functionality of the /etc/hosts file has been relegated by DNS but is still used in the following situations:

* **Bootstrapping**: This file is referred to during system startup since the DNS service is not started at this point.
* **Isolated Nodes**: If a node is not connected to the internet, it is unlikely to use DNS. The /etc/hosts file is useful for such nodes.
* **NIS**: The records in the hosts file are used as input for the NIS (Network Information Services) database.

Systemd systems use an alternative to the hostname command, the hostnamectl command. Similar to the hostname command, the hostnamectl command can also be used to query and set system hostnames, but the hostnamectl command provides additional categories for hostnames; *static*, *pretty*, and *transient* which are described below:

* **Static**: A *static hostname* is limited to [a-z], [0-9], hyphen -, and period . characters (no spaces i.e., localhost or ndg-server). This hostname is stored in the /etc/hostname file. Static hostnames can be set by a user.
* **Pretty**: Hostname can be in a human-readable format using any valid UTF-8 characters and can include special characters (i.e., Sarah’s Laptop or Joe’s Home PC).
* **Transient**: The transient hostname is a dynamic hostname usually set by the kernel to localhost by default. A dynamic hostname can be modified if needed. The transient hostname can be modified by DHCP or mDNS at runtime.

Hostnames can be up to 64 characters, but it is recommended that static and transient hostnames are limited to only 7 bit ASCII lowercase characters with no spaces or dots and conforming to strings acceptable for DNS domain names.

To demonstrate, in order to view the current hostname, simply use the hostnamectl command by itself or with the status subcommand:

**sysadmin@localhost:~$** hostnamectl status

Static hostname: localhost

Icon name: computer-vm

Chassis: vm

Machine ID: 5b91eb5e50594030b48b28f103cf5cd6

Boot ID: c04fc73d939a4fc18b279cd46d08f8d7

Virtualization: kvm

Operating System: Ubuntu 18.04.2 LTS

Kernel: Linux 4.15.0-45-generic

Architecture: x86-64

To change the local machine hostname, use the hostnamectl command with the set-hostname subcommand, this must be done with elevated permissions:

**sysadmin@localhost:~$** sudo hostnamectl set-hostname student

[sudo] password for sysadmin:

sysadmin@localhost:~$ hostnamectl status

Static hostname: student

Icon name: computer-vm

Chassis: vm

Machine ID: 5b91eb5e50594030b48b28f103cf5cd6

Boot ID: c04fc73d939a4fc18b279cd46d08f8d7

Virtualization: kvm

Operating System: Ubuntu 18.04.2 LTS

Kernel: Linux 4.15.0-45-generic

Architecture: x86-64

## Configuring DNS

Name resolution on a Linux host is accomplished by 3 critical files: the /etc/hosts, /etc/resolv.conf and /etc/nsswitch.conf files. Together, they describe the location of name service information, the order in which to check resources, and where to go for that information.

‌⁠​​⁠​

| **Files** | **Explanation** |
| --- | --- |
| /etc/hosts | This file contains a table of hostnames to IP addresses. It can be used to supplement a DNS server.  **sysadmin@localhost:~$** cat /etc/hosts  127.0.0.1 localhost |
| /etc/resolv.conf | This file contains the IP addresses of the name servers the system should consult in any attempt to resolve names to IP addresses. These servers are often DNS servers. It also can contain additional keywords and values that can affect the resolution process.  **sysadmin@localhost:~$** cat /etc/resolv.conf  nameserver 127.0.0.11 |
| /etc/nsswitch.conf | This file can be used to modify where hostname lookups occur. It contains a particular entry that describes in what order name resolution sources are consulted.  **sysadmin@localhost:~$** cat /etc/nsswitch.conf  # /etc/nsswitch.conf  #  Output Omitted...  hosts: files dns  Output Omitted...  The /etc/hosts file is searched first, the DNS server second:  hosts: files dns |

The /etc/resolv.conf file is the configuration file for DNS resolvers. The information in this file is normally set up by network initialization scripts.

A sample /etc/resolv.conf file looks like the following:

# /etc/resolv.conf

domain sample.com

search sample.com

# central nameserver

nameserver 191.74.10.12

sortlist 191.74.10.0 191.74.40.0

The format of the /etc/resolv.conf is:

*directive value1, value2…*

The configuration directives used in this file are:

| **Option** | **Meaning** |
| --- | --- |
| nameserver | IP address of the name server that the resolver will use  Maximum of 3 servers can be listed |
| domain | Domain name to be used locally |
| search | Search list to be used for hostname lookup |
| sortlist | Allow addresses to be sorted  The list is specified by IP addresses and optionally the netmask |

The nameserver setting is often set to the IP address of the DNS server. The following example uses the host command, which works with DNS to associate a hostname with an IP address. Note that the example server is associated with the IP address 192.168.1.2 by the DNS server:

**sysadmin@localhost:~$** host example.com

example.com has address 192.168.1.2

It is also common to have multiple nameserver settings, in the event that one DNS server isn't responding.

## NetworkManager

NetworkManager provides automatic detection and configuration of network interfaces on a Linux system. The NetworkManager supports some tools for users to interact with it, which are:

nmcli – a command-line tool used to configure networking.

nmtui– a simple curses-based text user interface, which is also used to configure and manage newtwork interface connections.

The nmcli command uses the following syntax:

nmcli [OPTIONS] OBJECT [COMMAND][ARGUMENTS...]

The OPTIONS for the nmcli command can be found by visiting the nmcli man page or by using the nmcli -help command. Commonly used options include the terse -t option that displays concise output and the pretty -p option, which makes the output easily readable by printing headers and aligning values.

The OBJECT field can be one of the following:

| **Object** | **Meaning** |
| --- | --- |
| general | Display information about or modify the status of NetworkManager. |
| networking | Display information about or modify the network managed by NetworkManager. |
| connection | Display information about or modify connections managed by NetworkManager. |
| device | Display information about or modify devices managed by NetworkManager. |
| radio | Display the status of, and enable or disable, the radio switches. |

To demonstrate, the following command can be used to show the existing network device:

The output in the examples below may not match the output in our virtual environment.

[sysadmin@centos ~]$ nmcli device

DEVICE TYPE STATE CONNECTION

**ens3 ethernet connected eth0**

When using the nmcli command, the object can be abbreviated. For example, the nmcli device command used in the example above can be shortened to nmcli dev or nmcli d:

[sysadmin@centos ~]$ nmcli dev

DEVICE TYPE STATE CONNECTION

**ens3 ethernet connected eth0**

[sysadmin@centos ~]$ nmcli d

DEVICE TYPE STATE CONNECTION

**ens3 ethernet connected eth0**

The nmcli command is useful for displaying additional information about a specific connection. In the example below, the nmcli command is used with the con object, the show subcommand and the pretty -p option, to display information about the eth0 connection:

[sysadmin@centos ~]$ nmcli -p con show eth0

The nmcli command can also be used to create a new connection. To add a connection, the following syntax can be used:

nmcli con add {OPTIONS} [IP]/[NETMASK] [GATEWAY]

For example, to add a connection named eth1, define the connection as Ethernet and specify the IP address, network mask, and gateway, the following command can be used:

[sysadmin@centos ~]$ nmcli con add con-name eth1 ifname eth1 type ethernet \ip4 10.0.2.18/24 gw4 10.0.2.2

The nmcli command above uses the con object with the add subcommand followed by a series of options, which are summarized below:

| **Options** | **Meaning** |
| --- | --- |
| con-name | Specifies the name of the network connection. In the example above, the con-name eth1 option adds a connection named eth1. |
| ifname | Name of the interface (device) that is used for the connection. |
| type | Specifies the connection type. Various types of connections exist such as ethernet, wifi, VLAN, bridge, and more. |
| ip4 | Specify an IPv4 IP address and netmask for the connection. |
| gw4 | Specify the gateway used for the connection. |

The con show command can be used to view the new connection:

[sysadmin@centos ~]$ nmcli -p con show eth1