GETTING STARTED

WITH

NETWORK SETUP

➢ NETWORK:

- A network is defined as a **Group of two or more computer systems** linked together.
- Types of networks are:
  - **LAN** (LOCAL AREA NETWORK)
  - **WAN** (WIDE AREA NETWORK)
  - **MAN** (METROPOLITAN AREA NETWORK)
- Network management is fairly easy when it comes to RedHat.
- Most of the network configuration is kept in files; therefore, adjusting these settings are simple.

MANAGING HOSTNAME:

- The host name can either be a **fully qualified domain name (FQDN)** in the format **hostname.domainname**, or a short host name without the domain.

  FQDN = hostname + domain_name
- Many networks have a **Dynamic Host Configuration Protocol (DHCP)** service that automatically supplies connected systems with a domain name.
- The value **localhost.localdomain** means that no specific static host name for the target system is configured, and the actual host name of the installed system is configured during the processing of the network configuration.
- Host names can only contain alphanumeric characters and - or .. Host name should be equal to or less than 64 characters. Host names cannot start or end with - and ..

→ **To verify the system hostname:**
#hostname (or) #hostnamectl

→ **To change hostname temporarily:**
#hostname server.example.com

→ **To change hostname permanently:**
#vim /etc/hostname
server.example.com

→ **To update hostname:**
#bash

CONFIGURING AN ETHERNET CONNECTION:

- If you connect a host to the network over Ethernet, you can manage the connection's settings on the command line by using the **nmcli** utility.
- A physical or virtual Ethernet **Network Interface Controller (NIC)** exists in the server's configuration.

→ **List the NetworkManager connection profiles:**

#nmcli connection show

NOTE: By default, NetworkManager creates a profile for each NIC in the host.

→ **Display the current settings of the connection profile:**

#nmcli connection show ens160

→ **If you want to create an additional connection profile, enter:**

#nmcli connection add con-name <connection-name> ifname <device-name> type ethernet

#nmcli connection add con-name ens170 ifname ens170 type ethernet

→ **Display the current settings of the connection profile:**

#nmcli connection show ens170

→ **To verify the device status:**

#nmcli device status

CONFIGURE THE IPV4 SETTINGS VIA DHCP:

#nmcli connection modify ens160 ipv4.method auto

→ **Bring UP or the load the configuration:**

#nmcli con up ens160

→ **To down the interface changes:**

#nmcli con down ens160

#nmcli con up ens160

→ **To verify the IP-Address of the host:**

#ifconfig (or) #ip a

#ip address show ens160

CONFIGURE THE IPV4 SETTINGS VIA STATIC:

→ **Setting static IPv4 address, network mask, default gateway, DNS servers:**

#nmcli connection modify ens160 ipv4.method manual ipv4.addresses 192.168.10.254/24 ipv4.gateway 192.168.10.1 ipv4.dns 192.168.10.2 ipv4.dns-search example.com

→ **To Up and down the interface changes:**

#nmcli con down ens160

#nmcli con up ens160

#ifconfig

→ **Display the DNS settings:**

#cat /etc/resolv.conf

SETTING IP-ADDRESS VIA CONFIGURATION FILE:

- From RHEL9 NetworkManager stores new network configurations to **/etc/NetworkManager/system-connections/** in a key-file format.
- Previously, NetworkManager stored new network configurations to **/etc/sysconfig/network-scripts/** in the ifcfg format.

#vim /etc/NetworkManager/system-connections/ens160.nmconnection

[ipv4]

address1=192.168.10.254/24,192.168.10.1

dns=192.168.10.2;

dns-search=example.com;

method=manual

→ **Bring down and up the interface changes:**

#nmcli con down ens160

#nmcli con up ens160

→ **To control network driver and hardware settings:**
#ethtool ens160

PING:

- Use the ping utility to verify that this host can send packets to other hosts:

SYNTAX: #ping <host-name-or-IP-address>

#ping 192.168.10.254

TRACEROUTE:

- It provides a map of how data on the internet travels from its source to its destination.

#traceroute 192.168.10.254

MANAGING HOSTS:

- This file contains IP addresses and their corresponding hostnames.
- When your system tries to resolve a hostname to an IP address or tries to determine the hostname for an IP address, it refers to the **/etc/hosts** file before using the name servers.

#vim /etc/hosts

<IP>	<Fully Qualified Domain Name>	<Short Name>
192.168.10.254	server.example.com	server

NOTE: Do not remove the **localhost** entry. Even if the system does not have a network connection or have a network connection running constantly, some programs need to connect to the system via the **localhost loopback interface.**

→ **Testing with hostname:**
#ping 192.168.10.254
#ping server.example.com
#ping server

ROUTE:

- Route manipulates the kernel's IP routing tables.
- Its primary use is to set up static routes to specific hosts or networks via an interface after it has been configured with the **ifconfig** program.
- The IP/kernel routing table acts as a crucial map, determining how network packets are forwarded between different hosts and networks.

→ **To display the IP/kernel routing table:**

#route

→ **To display routing table in full numeric form:**

#route -n

→ **Use the ip route command without options to display the IP routing table:**

#ip route

NOTE: Both **`route`** and **`ip route`** commands can be used to display and manipulate the routing table. However, `ip route` is considered more modern and flexible. It provides additional features and is recommended for users working with newer Linux systems.

NETSTAT:

- The network statistics (netstat) command is a networking tool used for troubleshooting and configuration, that can also serve as a monitoring tool for connections over the network.
- Both incoming and outgoing connections, routing tables, port listening, and usage statistics are common uses for this command.

→ **List all listening ports:**

#netstat -a

→ **List only TCP port connections:**

#netstat -at

→ **List only UDP port connections:**

#netstat -au

→ **List all actively listening ports:**

#netsta -l

→ **List all service ports by PID:**

#netstat -pantl

NOTE: Generally, the net-tools package provides the ifconfig, route, and netstat commands for most of the popular Linux distributions.

Use below command to install the netstat command on your system.

#yum install net-tools -y

MANAGING SYSTEM SERVICES:

- As a system administrator, you can manage system services by using the **systemctl** utility.
- You can perform various tasks, such as starting, stopping, restarting running services, enabling and disabling services to start at boot, listing available services, and displaying system services statuses.

SYNTAX: #systemctl [status/start/restart/reload/stop] <service_name>

→ **List all currently loaded service units:**

#systemctl list-units --type service

→ **Display detailed information about a sshd service:**

#systemctl status sshd

→ **Start a system service in the current session:**

#systemctl start sshd

→ **Stop a system service:**

#systemctl stop sshd

→ **Restart a system service:**

#systemctl restart sshd

ENABLING A SYSTEM SERVICE TO START AT BOOT:

- You can enable a service to start automatically at boot, these changes apply with the next reboot.

  SYNTAX: #systemctl <enable/disable> <service_name>

  → **Disable a service to start at boot:**

  #systemctl disable sshd

  → **Enable a service to start at boot:**

  #systemctl enable sshd

FIREWALLD:

- A firewall is a way to protect machines from any unwanted traffic from outside. It enables users to control incoming network traffic on host machines by defining a set of firewall rules. These rules are used to sort the incoming traffic and either block it or allow through.
- Services use one or more ports or addresses for network communication. Firewalls filter communication based on ports. To allow network traffic for a service, its ports must be open.
- firewalld blocks all traffic on ports that are not explicitly set as open.
  → **To stop firewalld service:**
  #systemctl stop firewalld
  #systemctl disable firewalld
  #systemctl status firewalld

SECURITY-ENHANCED LINUX (SELINUX):

- It is a security architecture for Linux® systems that allows administrators to have more control over who can access the system.
- It defines access controls for the apps, processes, and files on a system.
- It uses security policies, which are a set of rules that tell SELinux what can or can't be accessed, to enforce the access allowed by a policy.
  → **To disable SE-Linux set** SELINUX=disabled **in following file:**
  #vim /etc/selinux/config
  → **Take reboot for SE-Linux changes:**
  #reboot