**Introduction to Internet Protocol (IP)**

Internet Protocol (IP) serves as the cornerstone of network communication, defining the rules for sending and receiving packets of data across networks. Every packet carries the sender’s and receiver’s IP addresses, facilitating routing across interconnected networks. The structure of an IP address combines the **Network ID** and **Host ID**, which helps identify specific networks and devices within them.

**Internet Protocol Versions**

**IPv4 (Internet Protocol Version 4)**

* **32-bit Address Structure**: IPv4 addresses are composed of four octets, each 8 bits, represented in decimal (e.g., 192.168.1.1).
* **Address Range**: Each octet can hold values between 0 and 255.
* **Classes of IPv4 Addresses**:
  + **Class A**: 1.0.0.0 to 127.255.255.255 (large networks).
  + **Class B**: 128.0.0.0 to 191.255.255.255 (medium-sized networks).
  + **Class C**: 192.0.0.0 to 223.255.255.255 (small networks).
  + **Class D**: 224.0.0.0 to 239.255.255.255 (multicast).
  + **Class E**: 240.0.0.0 to 255.255.255.255 (experimental).
* **Special Addresses**:
  + **Loopback**: 127.0.0.1, used for internal testing.
  + **Broadcast**: Sends data to all devices in a network.

**IPv6 (Internet Protocol Version 6)**

* **128-bit Address Structure**: IPv6 offers a much larger address space, written in hexadecimal (e.g., 2001:0db8:85a3:0000:0000:8a2e:0370:7334).
* **Benefits**: Enhanced routing, better security, and efficient IP allocation.
* **Address Format**: Eight groups of four hexadecimal digits, separated by colons.
* **Transition from IPv4 to IPv6**:
  + **Dual-Stack**: Running both IPv4 and IPv6 concurrently.
  + **Tunneling**: Encapsulating IPv6 packets within IPv4 for compatibility.

**TCP (Transmission Control Protocol)**

TCP is a connection-oriented protocol, ensuring reliable data transmission:

* **Connection Establishment**: Uses a three-way handshake (SYN, SYN-ACK, ACK).
* **Reliability**: TCP segments include checksums and sequence numbers to guarantee ordered and error-free data.
* **Flow Control**: Implements windowing to manage data flow between sender and receiver.
* **Use Cases**: Web browsing, email, and file downloads.

**UDP (User Datagram Protocol)**

UDP is a lightweight, connectionless protocol:

* **Faster Transmission**: Lacks the reliability mechanisms of TCP, making it faster but less reliable.
* **Real-Time Applications**: Ideal for video streaming, VoIP, and gaming.
* **Header Structure**: Simpler than TCP, with minimal overhead.

**Network Access Layer and MAC Address**

The Network Access Layer works at the hardware level using **MAC (Media Access Control) addresses**, which identify devices on a local network:

* **Structure**: A MAC address is 48 bits, represented in hexadecimal (e.g., 00:1A:2B:3C:4D:5E).
* **Blocks and Bits**: Divided into 6 blocks, each 8 bits.
* **NIC (Network Interface Card)**: Every NIC has a unique MAC address, enabling network communications.

**IANA (Internet Assigned Number Authority)**

IANA is responsible for managing global IP address allocation, DNS root zone, and protocol assignments:

* **Physical Address Management**: Handles MAC and hardware addresses.
* **Logical Address Management**: Oversees IPv4 and IPv6 distribution.

**Subnetting and Subnet Masks**

Subnetting divides a larger network into smaller, manageable sub-networks:

* **Subnet Mask**: Defines the network and host portions of an IP address.
  + Example: The subnet mask 255.255.255.0 corresponds to 11111111.11111111.11111111.00000000 in binary.
* **CIDR (Classless Inter-Domain Routing)**: Represents subnets using / notation (e.g., 192.168.1.0/24).

**Subnetting Example**

* **IP Address**: 192.168.10.0/24.
* **Subnet Calculation**: Splitting the network into subnets with smaller host pools.

**Private and Public IP Addresses**

* **Private IP Ranges**:
  + **Class A**: 10.0.0.0 to 10.255.255.255.
  + **Class B**: 172.16.0.0 to 172.31.255.255.
  + **Class C**: 192.168.0.0 to 192.168.255.255.
* **Public IPs**: Routable on the global internet, typically purchased through ISPs.
* **NAT (Network Address Translation)**: Allows multiple devices on a local network to share a single public IP.

**Communication and Routing**

* **Static Routing**: Predefined paths set by network administrators.
* **Dynamic Routing Protocols**:
  + **OSPF (Open Shortest Path First)**: Interior routing protocol.
  + **BGP (Border Gateway Protocol)**: Used for internet routing between different networks.
* **Gateways**: Act as an access point between networks with different IP classes.

**Subnetting Practical Scenarios**

* **Binary Calculations**:
  + Converting decimal to binary for IP and subnet mask calculations.
* **Design Scenarios**:
  + Creating efficient subnets for departmental separation.

**Broadcast and Multicast**

* **Broadcast**: Sends data to all devices within a network.
* **Multicast**: Sends data to a specific group of devices.
* **IGMP (Internet Group Management Protocol)**: Manages multicast group membership.

**Additional Protocols and Concepts**

* **DHCP (Dynamic Host Configuration Protocol)**: Automates IP address assignment and management.
* **ARP (Address Resolution Protocol)**: Maps IP addresses to MAC addresses.
* **ICMP (Internet Control Message Protocol)**: Used for error reporting and diagnostic tools like ping and traceroute.

**Security Considerations**

* **IP Spoofing**: Prevent unauthorized access and attacks by implementing packet filtering.
* **IPv6 Security**: Includes built-in IPsec for encrypted communications.

**The Future of IP**

* **IPv6 Adoption**: Overcoming the IPv4 address exhaustion.
* **Emerging Technologies**:
  + **SDN (Software-Defined Networking)**: Enhances network flexibility.
  + **IoT (Internet of Things)**: Expands IP usage with interconnected devices.

**Chapter: Network Management**

**Introduction**

Network management is a critical aspect of system administration. Efficient handling of network interfaces ensures reliable connectivity and optimal performance. This chapter delves into using Linux command-line tools like ifconfig, nmcli, and nmtui for managing network interfaces, creating new connections, modifying configurations, and more.

**1. Basic Network Commands**

**Viewing Network Interfaces**

* **View current network configurations**:
* ifconfig

This command displays the current state of all network interfaces.

* **List all network devices and their status**:
* nmcli dev status

**Navigating Network Configuration Files**

* **Access network-scripts directory**:
* cd /etc/sysconfig/network-scripts/
* ls
* **View the configuration of a specific interface (e.g., ifcfg-ens160)**:
* cat ifcfg-ens160

**2. Adding and Managing Network Interfaces in a Virtual Machine**

**Adding Network Interfaces**

* **List all network devices**:
* nmcli dev status
* **Show detailed information about network devices**:
* nmcli dev show

**3. Checking Connection Details**

* **Display all available network connections**:
* nmcli connection show
* **View brief network details using alternative commands**:
* ip addr
* ip a
* nmcli c s

**4. Activating and Deactivating Network Connections**

**Bringing Up and Down Connections**

* **Activate a network connection**:
* nmcli connection up <conNAMe>
* **Deactivate a network connection**:
* nmcli connection down <conNAMe>

**5. Creating New Network Connections**

**Basic Ethernet Connection**

* **Check device status**:
* nmcli device status
* **Add a new Ethernet connection**:
* nmcli con add type ethernet con-name ens224 ifname ens224
* **Verify the new connection**:
* nmcli connection show
* ip a
* **Test network connectivity**:
* ping <IP>

**Bringing Interfaces Up and Down**

* **Manually bring down an interface**:
* ifdown ens224
* **Manually bring up an interface**:
* ifup ens224
* **Deactivate/Activate using nmcli**:
* nmcli connection down ens224
* nmcli connection up ens224

**6. Configuring Static IP Addresses**

**Setting a Static IP**

* **Add an interface with a static IP**:
* nmcli con add type ethernet con-name static2 ifname ens224 ip4 192.168.0.50/24 gw4 192.168.0.1
* **Alternative way to set a static IP (manual method)**:
* nmcli con add type ethernet con-name vinay ifname ens224 ipv4 192.168.0.40/24 gw4 192.168.0.1 ipv4.method manual
* **Verify changes**:
* ls
* ip a
* ping <IP>

**Modifying Connection Properties**

* **Add DNS server configuration**:
* nmcli con mod static2 ipv4.dns "8.8.8.8"
* **Deactivate and reactivate the connection to apply changes**:
* nmcli con down static2
* nmcli con up static2
* nmcli connection show

**7. Advanced Network Management**

**Stopping Auto-connection**

* **Disable auto-connect for a connection**:
* nmcli con mod static2 connection.autoconnect no

**Setting Permissions**

* **Grant permission to a user to view network details**:
* nmcli con mod static2 connection.permission zybi
* **Grant permission to multiple users**:
* nmcli con mod static2 connection.permission user:zybi,arslan

**Reloading Network Connections**

* **Restart NetworkManager to apply changes**:
* systemctl restart NetworkManager

**8. Editing and Removing Configurations**

**Editing Connection Properties**

* **Edit a connection interactively**:
* nmcli con edit static2

**Removing Specific Entries**

* **Remove a specific DNS entry**:
* nmcli con mod static2 -ipv4.dns 8.8.8.8

**Deleting a Connection**

* **Delete a connection**:
* nmcli con del static2

**9. Graphical Network Management with nmtui**

* **Launch nmtui for a text-based GUI**:
* nmtui

Use nmtui for easier configuration and management with a simple interface.

**Network Configuration Management**

In this chapter, we will explore how to manage network connections using nmcli (Network Manager Command Line Interface) in Linux. This includes adding, modifying, and managing connections, as well as assigning IPv4 and IPv6 addresses.

**Checking Current Connections**

Before making changes, it’s important to know the current state of network devices and connections.

nmcli dev status

This command will display the status of all network devices on the system.

**Navigating to Network Scripts**

Network configuration files are usually located in the following directory:

cd /etc/sysconfig/network-scripts

ls

This allows you to list existing configuration files.

**Adding a New Connection**

To add a new Ethernet connection and assign a static IPv4 address:

nmcli connection add type ethernet con-name ens224 ifname ens224 ipv4.address 192.168.0.145/24 ipv4.dns 8.8.8.8 gw4 192.168.0.1 ipv4.method manual

Then, bring the connection up:

nmcli connection up ens224

Verify the connection status:

nmcli dev status

ip a

**Modifying an Existing Connection to Add Multiple IPv4 Addresses**

You can modify an existing connection to add multiple IPv4 addresses:

nmcli connection modify ens256 +ipv4.addresses 192.168.0.155/24 +ipv4.dns 8.8.4.4

After modifying, reload the connection:

nmcli connection reload

ifup ens256

nmcli connection up ens256

ip a

**Removing Multiple IPv4 Addresses**

To remove an IPv4 address from a connection:

nmcli connection modify ens256 -ipv4.addresses 192.168.0.155/24 -ipv4.dns 8.8.4.4

Reload the connection again:

nmcli connection reload

nmcli connection up ens256

ip a

**Setting Up IPv6 Addresses**

To set up an IPv6 address, you must first ensure that your hostname is correctly set:

hostnamectl set-hostname nehraclasses.local

Next, add a new Ethernet connection for IPv6:

nmcli connection add type ethernet con-name ens224 ifname ens224

Modify the connection to assign an IPv6 address:

nmcli connection modify ens224 ipv6.addresses 'dfdd:df34:abc1::c0ad:8/64' ipv6.method manual

Reload the connection:

nmcli connection reload

nmcli connection up ens224

ip a

**Summary of Commands**

**Key Commands for Managing Network Connections**

* **Check device status:**
* nmcli dev status
* **Navigate to network scripts:**
* cd /etc/sysconfig/network-scripts
* ls
* **Add a new connection:**
* nmcli connection add type ethernet con-name <name> ifname <interface> ipv4.address <IP>/<CIDR> ipv4.dns <DNS> gw4 <Gateway> ipv4.method manual
* **Bring up a connection:**
* nmcli connection up <connection\_name>
* **Modify existing connection to add IPv4 addresses:**
* nmcli connection modify <connection\_name> +ipv4.addresses <IP>/<CIDR>
* **Remove IPv4 addresses:**
* nmcli connection modify <connection\_name> -ipv4.addresses <IP>/<CIDR>
* **Set up IPv6:**
* nmcli connection modify <connection\_name> ipv6.addresses '<IPv6 Address>/<CIDR>' ipv6.method manual
* **Reload all connections:**

nmcli connection reload