

System Administration (2)

IT401

Prepared by
Dr. Ahmed El-Douh

Lecture 7



System Administration 2

Lecture 7

IT401

Managing Linux Networking

Prepared by
Dr. Ahmed El-Douh

Objectives

Upon completion of this lecture, you will be able to:

- Understand key computer networking concepts.
- Check and test current network configuration using basic tools.
- Manage network devices using nmcli and NetworkManager.
- Edit configuration files to modify network settings.
- Configure and test host names and name resolution.

Outlines

- Networking Concepts and Practice
- Validating Network Configuration
- Configuring Networking with nmcli
- Editing Network Configuration Files
- Managing Host Names and Name Resolution

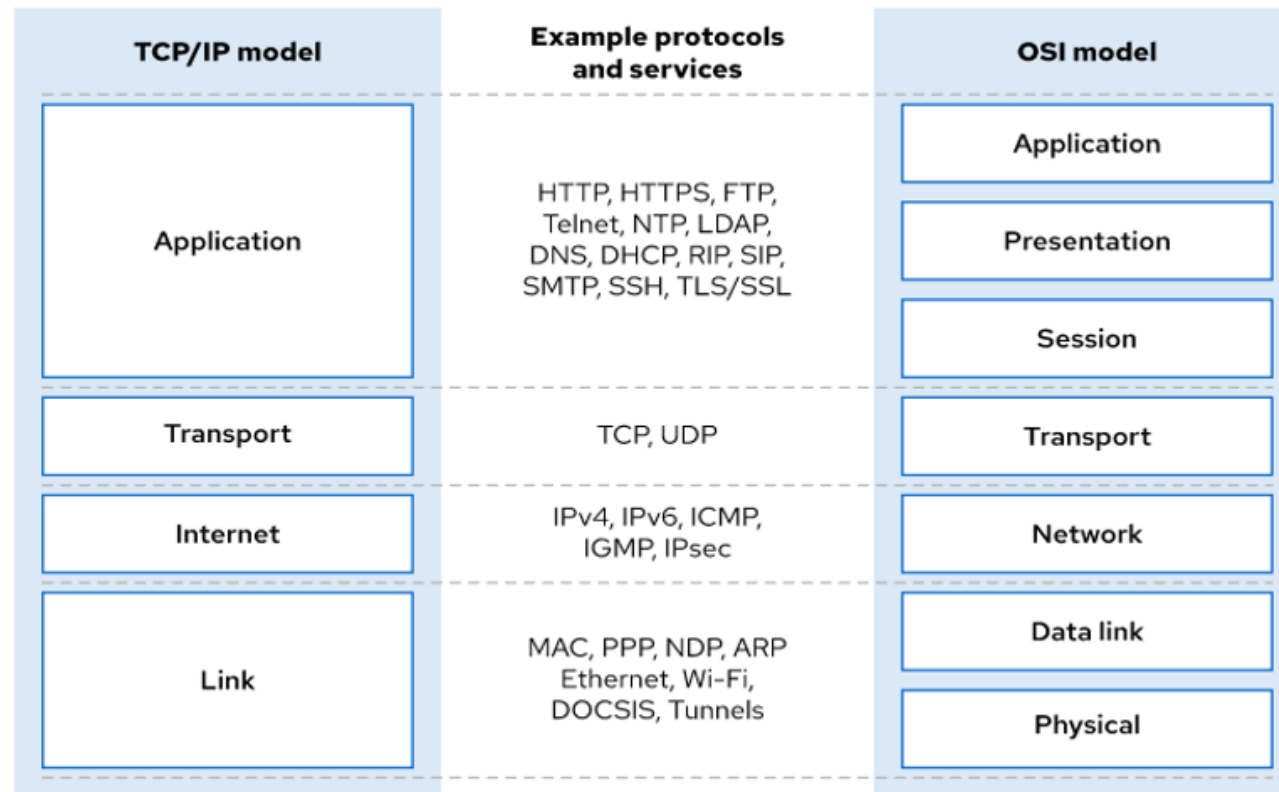


Networking Concepts and Practice



IPv4 Networking

- IPv4 Networking:
The TCP/IP standards are based on a **four-layer network model**, as defined in RFC 1122



The TCP/IP standards are based on a four-layer network model

- **Application Layer:**

- Each application has communication standards to let clients and servers exchange data across platforms.
- Common protocols: **SSH** (remote login), **HTTPS** (secure web), **NFS/CIFS** (file sharing), **SMTP** (email delivery).

- **Transport Layer:**

- The main transport protocols are **TCP** and **UDP**.
 - ✓ **TCP** provides reliable, connection-oriented communication.
 - ✓ **UDP** is a connectionless **datagram** protocol.
- Applications use **TCP** or **UDP ports** for communication.
- A list of common ports is stored in the **/etc/services** file.
- When data packets are sent, the **port number** and **IP address** together form a **socket** — each packet has a **source socket** and a **destination socket**, used for **monitoring and filtering**.

The TCP/IP standards are based on a four-layer network model

■ Internet Layer:

- Moves data from the source host to the destination host.
- Each host has an **IP address** and a **network prefix**.
- **Routers** connect different networks.
- Uses the **ICMP protocol** for control messages.
- The **ping** command sends **ICMP Echo Request** and receives **Echo Reply** to test connectivity.

■ Link Layer:

- Provides the physical connection to the network medium.
- Common types: **Ethernet (802.3)** and **WLAN (802.11)**.
- Each device has a **MAC address**, used to identify the destination of packets within a local network.

IPv4 addresses

- An IPv4 address is a **32-bit number** written as four **octets** (0–255) separated by dots.
- It's divided into two parts: **network part** and **host part**. IP: 192.168.1.10
- Devices in the same subnet share the same network part.
- The **network part** identifies the subnet; the **host part** identifies the device.
- The **netmask** defines how many bits belong to the network and how many to the host.
- The **lowest address** in a subnet (host part = 0) is the **network address**.
- The **highest address** (host part = 1s) is the **broadcast address** used to send data to all devices on the subnet.

IPv4 Networking

IP Address:

172.17.5.3 = 10101100.00010001.00000101.00000011

Prefix: /16

Netmask:

255.255.0.0 = 11111111.11111111.00000000.00000000

10101100.00010001.00000101.00000011

Network

Host

IP Address:

192.168.5.3 = 11000000.10101000.00000101.00000011

Prefix: /24

Netmask:

255.255.255.0 = 11111111.11111111.11111111.00000000

11000000.10101000.00000101.00000011

Network

Host

Figure 11.1: IPv4 addresses and netmasks

Network Masks and CIDR Notation

- Network masks can be written in two forms:
 - Old form: 255.255.255.0
 - CIDR form: /24
- Both forms mean the same thing: the first 24 bits represent the **network part**.
- The **prefix (/n)** shows how many bits in the IP address are used for the network.
- These examples show the relationship between the **IP address**, **netmask (prefix)**, **network part**, and **host part**.

Calculating the Network Address (Example)

Calculating the network address for 192.168.1.107/24

Host addr	192.168.1.107	11000000.10101000.00000001.01101011
Network prefix	/24 (255.255.255.0)	11111111.11111111.11111111.00000000
Network addr	192.168.1.0	11000000.10101000.00000001.00000000
Broadcast addr	192.168.1.255	11000000.10101000.00000001.11111111

IPv4 Networking

Calculating the network address for 10.1.1.18/8

Host addr	10.1.1.18	00001010.00000001.00000001.00010010
Network prefix	/8 (255.0.0.0)	11111111.00000000.00000000.00000000
Network addr	10.0.0.0	00001010.00000000.00000000.00000000
Broadcast addr	10.255.255.255	00001010.11111111.11111111.11111111

Calculating the network address for 172.16.181.23/19

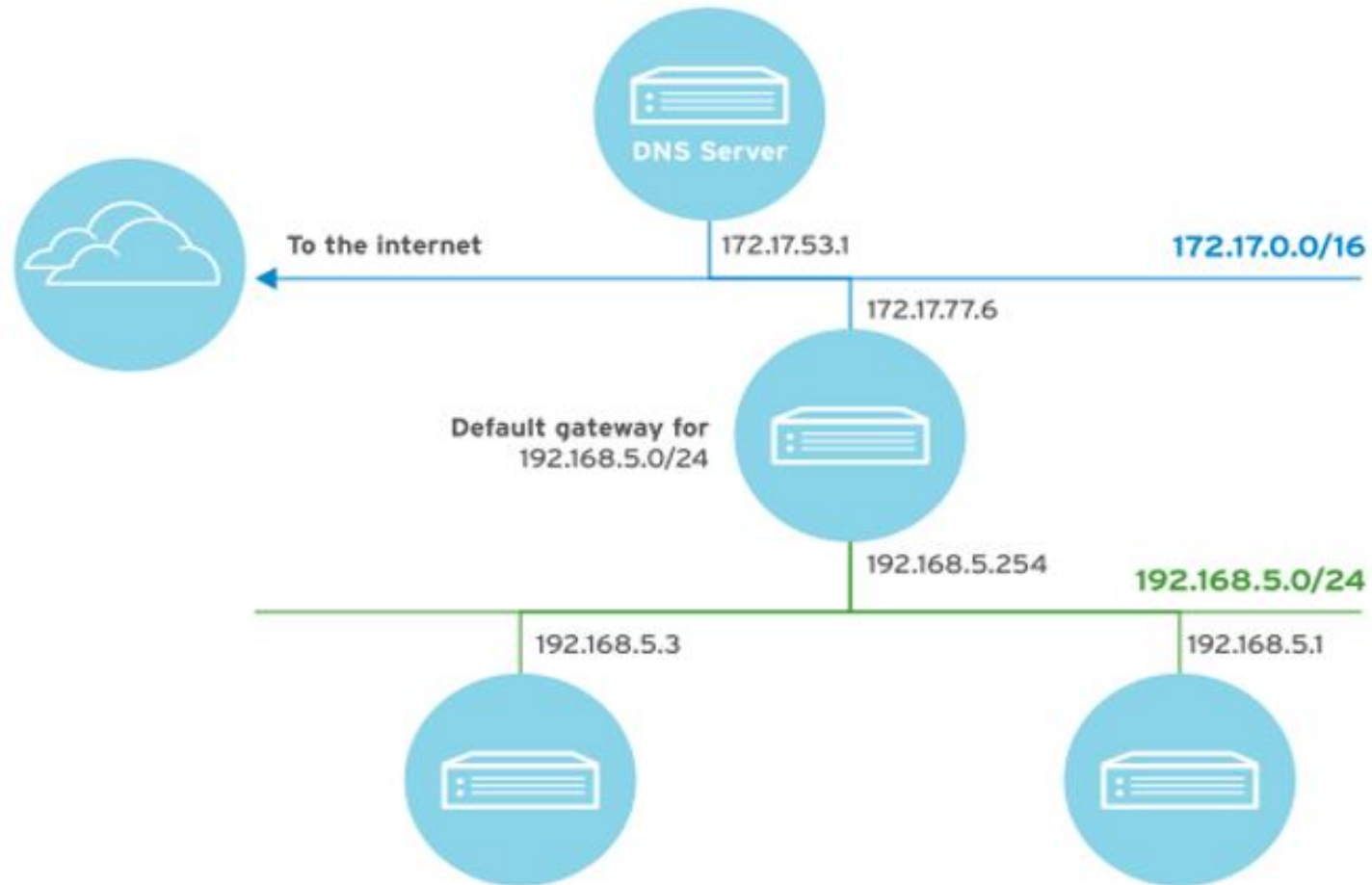
Host addr	172.168.181.23	10101100.10101000.10110101.00010111
Network prefix	/19 (255.255.224.0)	11111111.11111111.11100000.00000000
Network addr	172.168.160.0	10101100.10101000.10100000.00000000
Broadcast addr	172.168.191.255	10101100.10101000.10111111.11111111



Configuring and Securing OpenSSH Service



Network routing and DNS concepts



IPv4 Routing

- Every host has a **routing table** that decides how to send packets to other networks.
- Each route specifies:
 - **Destination network**
 - **Interface** to send traffic
 - **Router (gateway)** if needed
- If no specific route matches, the system uses the **default route (0.0.0.0/0)** to send traffic to the Internet through a router.
- Routers **forward** packets using their own routing tables until the packet reaches its final destination.

Example routing table

DESTINATION	INTERFACE	ROUTER (IF NEEDED)
192.0.2.0/24	wlo1	
192.168.5.0/24	enp3s0	
0.0.0.0/0 (default)	enp3s0	192.168.5.254

IPv4 Networking

❖ Names and IP Addresses

- Computers use **IP addresses** to communicate.
- Humans prefer using **names** instead of numbers.
- **DNS (Domain Name System)** maps names to IP addresses using **nameservers**.
- The nameserver doesn't need to be on the same network, only reachable.

❖ DHCP or Static Network Configuration

- **DHCP** automatically assigns network settings at system startup.
- If no **DHCP server** is available, a **static configuration** must be used manually.
- DHCP tracks devices by their **hardware (MAC) address**.
- A network interface may have one or more IP addresses (IPv4 and IPv6).

IPv4 Networking

Network Interface Names

- Older Linux systems used simple names like **eth0**, **eth1**, **eth2**.
- In **RHEL 7**, interfaces get **predictable names** based on firmware, topology, and device type.

Naming Rules:

- **en** → Ethernet interfaces
- **wl** → Wireless LAN (Wi-Fi) interfaces
- **ww** → WWAN (cellular) interfaces

Additional letters:

- **o** → Onboard
- **s** → Hotplug slot
- **p** → PCI location
- **x** → Uses MAC address

Example:

- **eno1** → Onboard Ethernet interface
- **enp2s0** → PCI Ethernet interface on slot 2

➤ If the system cannot determine a fixed name, traditional names like **ethN** are used.



Validating Network Configuration



Validating Network Configuration

Objective:

Students should be able to test and review current network configuration using basic utilities.

Displaying IP Addresses:

The command **/sbin/ip** is used to display device and address details.

Example:

```
ip addr show eth0
```

Explanation:

- 1 **UP** → Interface is active
- 2 **link/ether** → MAC address of the device
- 3 **inet** → IPv4 address and prefix
- 4 **brd** → Broadcast address and scope
- 5 **inet6** → IPv6 address information

Validating Network Configuration

Checking Network Statistics

- The **ip** command can also display network performance data:
 - **RX (Received)** and **TX (Transmitted)** packets, bytes, and errors.
- **Command:** `ip -s link show eth0`

```
[student@desktopX ~]$ ip -s link show eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen
1000
link/ether 52:54:00:00:00:0a brd ff:ff:ff:ff:ff:ff
  RX: bytes  packets  errors  dropped overrun mcast
    269850    2931     0      0        0        0
  TX: bytes  packets  errors  dropped carrier collsns
    300556    3250     0      0        0        0
```

Troubleshooting Routing

The **/sbin/ip** command is also used to show routing information.

```
[student@desktopX ~]$ ip route  
default via 172.25.0.254 dev eth0  proto static  metric 1024  
172.25.X.0/24 dev eth0  proto kernel  scope link  src 172.25.X.10  
10.0.0.0/8 dev eth1  proto kernel  scope link  src 10.0.0.11
```

Troubleshooting Routing

❖ Tracing Network Path

- Use **traceroute** or **tracpath** to trace the route to a remote host.
 - These commands show all network hops between your system and the destination.
 - **tracpath** works with **UDP packets** by default.
-
- **traceroute** can use:
 - ✓ **UDP** (default)
 - ✓ **ICMP** with `-I`
 - ✓ **TCP** with `-T`

```
[student@desktopX ~]$ tracpath access.redhat.com
...
 4: 71-32-28-145.rcmt.qwest.net          48.853ms asymm  5
 5: dcp-brdr-04.inet.qwest.net          100.732ms asymm  7
 6: 206.111.0.153.ptr.us.xo.net          96.245ms asymm  7
 7: 207.88.14.162.ptr.us.xo.net          85.270ms asymm  8
 8: ae1d0.cir1.atlanta6-ga.us.xo.net      64.160ms asymm  7
 9: 216.156.108.98.ptr.us.xo.net         108.652ms
10: bu-ether13.atlmgamq46w-bcr00.tbone.rr.com 107.286ms asymm 12
...
```

Troubleshooting Ports and Services

- TCP services use **IP addresses**, **protocols**, and **ports** to communicate.
- Services **listen** on specific ports for connections.
- Standard ports are listed in **/etc/services**.

Command: `ss -ta`

- The `ss` command shows socket and connection details.
- It replaces the old **netstat** command.

```
[student@desktopX ~]$ ss -ta
State      Recv-Q  Send-Q   Local Address:Port      Peer Address:Port
LISTEN     0        128      *:sunrpc                 *:*
LISTEN     0        128      ①*:ssh                   *:*
LISTEN     0        100      ②127.0.0.1:smtp          *:*
LISTEN     0        128      *:36889                  *:*
ESTAB      0        0        ③172.25.X.10:ssh         172.25.254.254:59392
LISTEN     0        128      :::sunrpc                :::*
LISTEN     0        128      ④:::ssh                  :::*
```

- ① The port used for SSH is listening on all IPv4 addresses. The "*" is used to represent "all" when referencing IPv4 addresses or ports.
- ② The port used for SMTP is listening on the 127.0.0.1 IPv4 loopback interface.
- ③ The established SSH connection is on the 172.25.X.10 interface and originates from a system with an address of 172.25.254.254.
- ④ The port used for SSH is listening on all IPv6 addresses. The ":::" syntax is used to represent all IPv6 interfaces.



Configuring Networking with nmcli



Configuring Networking with **nmcli**

Objectives:

- Learn how to manage network settings and devices using **nmcli** and **NetworkManager**.

NetworkManager:

- A **daemon** that monitors and manages network connections.
- Provides automatic network configuration and status monitoring.
- Works with both **command-line** tools (like nmcli) and **graphical interfaces**.
- Saves configuration files in **/etc/sysconfig/network-scripts**.

Viewing Network Information with **nmcli**

❖ Viewing Network Information with **nmcli**

- To list all saved connections, use: **nmcli con show**
- To show only active connections, use: **nmcli con show --active**

○ Output Columns:

- **NAME:** Connection name
- **UUID:** Unique identifier for each connection
- **TYPE:** Connection type (wired or wireless)
- **DEVICE:** Network interface used (e.g., eth0, wlp3s0)

```
[root@desktopX ~]# nmcli con show
```

NAME	UUID	TYPE	DEVICE
static-eth0	f3e8dd32-3c9d-48f6-9066-551e5b6e612d	802-3-ethernet	eth0
System eth0	5fb06bd0-0bb0-7ffb-45f1-d6edd65f3e03	802-3-ethernet	--
guest	f601ca8a-6647-4188-a431-dab48cc63bf4	802-11-wireless	wlp3s0

```
[root@desktopX ~]# nmcli con show --active
```

NAME	UUID	TYPE	DEVICE
static-eth0	f3e8dd32-3c9d-48f6-9066-551e5b6e612d	802-3-ethernet	eth0
guest	f601ca8a-6647-4188-a431-dab48cc63bf4	802-11-wireless	wlp3s0

Viewing Connection Details with **nmcli**

- To see the detailed configuration of a specific connection, use:

nmcli con show "connection-name"

Example:

nmcli con show "static-eth0"

```
[root@desktopX ~]# nmcli con show "static-eth0"
...
ipv4.method:                manual
ipv4.dns:                   172.25.254.254, 8.8.8.8
ipv4.dns-search:
ipv4.addresses:             { ip = 172.25.X.10/24, gw = 172.25.X.254 }
ipv4.routes:
ipv4.ignore-auto-routes:    no
```

○ **Explanation of Key Fields:**

- **ipv4.method:** Connection type (manual or auto via DHCP).
- **ipv4.dns:** DNS servers used for name resolution.
- **ipv4.addresses:** IP address and subnet mask.
- **ipv4.routes:** Gateway or routing information.
- **ipv4.dhcp-send-hostname:** Whether the hostname is sent via DHCP.

Viewing Connection Details with **nmcli**

The **nmcli** command can also be used to show device status and details.

```
[root@desktopX ~]# nmcli dev status
DEVICE   TYPE      STATE      CONNECTION
eth0     ethernet  connected  static-eth0
wlp3s0   wifi      connected  guest
lo       loopback  unmanaged  --
[root@desktopX ~]# nmcli dev show eth0
GENERAL.DEVICE:                eth0
GENERAL.TYPE:                  ethernet
GENERAL.HWADDR:                52:54:00:00:00:0A
GENERAL.MTU:                   1500
GENERAL.STATE:                 100 (connected)
GENERAL.CONNECTION:            static-eth0
GENERAL.CON-PATH:               /org/freedesktop/NetworkManager/
ActiveConnection/1
WIRED-PROPERTIES.CARRIER:     on
IP4.ADDRESS[1]:                ip = 172.25.X.10/24, gw = 172.25.X.254
IP4.DNS[1]:                    172.25.254.254
IP6.ADDRESS[1]:                ip = fe80::5054:ff:fe00:b/64, gw = ::
```

Creating Network Connections with **nmcli**

- When creating a new connection, the **order of arguments** matters.
- Start with common options: **type** and **interface name**.
- Then specify connection details like **IP address**, **prefix**, and **gateway**.
- You can assign **multiple IPs** to a single device.
- Extra settings (e.g., **DNS server**) can be added later by modifying the connection

Examples of creating new connections

Follow along with the next steps while your instructor discusses **nmcli** syntax.

1. Define a new connection named "default" which will autoconnect as an Ethernet connection on the eth0 device using DHCP.

```
[root@desktopX ~]#  
nmcli con add con-name "default" type ethernet ifname eth0
```

2. Create a new connection named "static" and specify the IP address and gateway. Do not autoconnect.

```
[root@desktopX ~]#  
nmcli con add con-name "static" ifname eth0 autoconnect no type ethernet  
ip4 172.25.X.10/24 gw4 172.25.X.254
```

3. The system will autoconnect with the DHCP connection at boot. Change to the static connection.

```
[root@desktopX ~]#  
nmcli con up "static"
```

4. Change back to the DHCP connection.

```
[root@desktopX ~]#  
nmcli con up "default"
```

Type Options

- Connection options depend on the type (e.g., ethernet, wifi, bridge, bond, VPN, VLAN).
- Ethernet connections may include optional settings like MAC address or MTU.
- Wi-Fi connections must specify SSID and may include extra options.
- To view all available options, use: [nmcli con add help](#)

```
[root@desktopX ~]# nmcli con add help
Usage: nmcli connection add { ARGUMENTS | help }

ARGUMENTS := COMMON_OPTIONS TYPE_SPECIFIC_OPTIONS IP_OPTIONS

COMMON_OPTIONS:
    type <type>
    ifname <interface name> | ""
    [con-name <connection name>]
    [autoconnect yes|no]

    [save yes|no]

TYPE_SPECIFIC_OPTIONS:
    ethernet:    [mac <MAC address>]
                 [cloned-mac <cloned MAC address>]
                 [mtu <MTU>]
    ...
```

Modifying Network Interfaces with nmcli

```
[root@desktopX ~]# nmcli con show "static"
connection.id:                static
connection.uuid:              f3e8dd32-3c9d-48f6-9066-551e5b6e612d
connection.interface-name:    eth0
connection.type:              802-3-ethernet
connection.autoconnect:       yes
connection.timestamp:         1394905322
connection.read-only:         no
...
```

Examples of connection modifications

Follow along with the next steps while your instructor discusses **nmcli** syntax.

1. Turn off autoconnect.

```
[root@desktopX ~]#
nmcli con mod "static" connection.autoconnect no
```

2. Specify a DNS server.

```
[root@desktopX ~]#
nmcli con mod "static" ipv4.dns 172.25.X.254
```

3. Some configuration arguments may have values added or removed. Add a +/- symbol in front of the argument. Add a secondary DNS server.

```
[root@desktopX ~]#
nmcli con mod "static" +ipv4.dns 8.8.8.8
```

4. Replace the static IP address and gateway.

```
[root@desktopX ~]#
nmcli con mod "static" ipv4.addresses "172.25.X.10/24 172.25.X.254"
```

5. Add a secondary IP address without a gateway.

```
[root@desktopX ~]#
nmcli con mod "static" +ipv4.addresses 10.10.10.10/16
```

Summary of **nmcli** Commands

nmcli commands

COMMAND	USE
nmcli dev status	List all devices.
nmcli con show	List all connections.
nmcli con up "<ID>"	Activate a connection.
nmcli con down "<ID>"	Deactivate a connection. The connection will restart if autoconnect is yes.
nmcli dev dis <DEV>	Bring down an interface and temporarily disable autoconnect.
nmcli net off	Disable all managed interfaces.
nmcli con add ...	Add a new connection.
nmcli con mod "<ID>" ...	Modify a connection.
nmcli con del "<ID>"	Delete a connection.



Configuring Host Names and Name Resolution



Configuring Host Names and Name Resolution

OBJECTIVES

After completing this section, students should be able to configure and test system host name and name resolution.

CHANGING THE SYSTEM HOST NAME

The **hostname** command displays or temporarily modifies the system's fully qualified host name.

```
[root@desktopX ~]# hostname  
desktopX.example.com
```

```
[root@desktopX ~]# cat /etc/hostname  
desktopX.example.com
```

Configuring Name Resolution

The *stub resolver* is used to convert host names to IP addresses or the reverse. The contents of the file **/etc/hosts** are checked first.

```
[root@desktopX ~]# cat /etc/hosts
127.0.0.1      localhost localhost.localdomain localhost4 localhost4.localdomain4
::1           localhost localhost.localdomain localhost6 localhost6.localdomain6

172.25.254.254 classroom.example.com
```

Configuring Name Resolution

❖ Host Name Resolution with `getent`

- The command **`getent hosts hostname`** is used to test how the system resolves host names.
- It first checks the **`/etc/hosts`** file for the entry.
- If not found, it queries a **DNS nameserver** as defined in **`/etc/resolv.conf`**.

❖ `/etc/resolv.conf` parameters:

- **nameserver:** Defines the IP address of the DNS server (up to 3 can be listed for backup).
- **search:** Specifies domain names to append when resolving short hostnames.
- **domain:** Used for defining a default domain (should not be used with *search* in the same file).



Thank You