

LPI 109.1 - Persistent network configuration

Curs 2021 - 2022

ASIX M01-ISO 109 Networking Fundamentals

Persistent network configuration	2
Description	2
TCP/IP Configuration	2
Interfaces	2
ifconfig	3
ip link	4
nmcli device	5
Setting the hostname	5
Configuring DNS	6
Routing Table	8
Network Interfaces	9
Red Hat Interface Configuration	10
Debian Interface configuration	11
Network Tools: NetworkManager and iproute2	12
Network Manager	12
iproute2 tools	15
Systemd-networkd	15
Example Exercises	15

Persistent network configuration

Description

Key concepts:

- ❑ Understand basic TCP/IP host configuration.
- ❑ Configure ethernet and wi-fi network using NetworkManager.
- ❑ Awareness of systemd-networkd.

Commands and files:

- ❑ /etc/hostname
- ❑ /etc/hosts
- ❑ /etc/nsswitch.conf
- ❑ /etc/resolv.conf
- ❑ nmcli
- ❑ hostnamectl
- ❑ ifup
- ❑ ifdown

TCP/IP Configuration

Configuration:

- interfaces
 - eth0 enp1s0 (wired)
 - wlp0s0 (wifi)
 - loopback lo 127.0.0.1
 - virtual VM bridges virbr0
- IP address / Mask
- gateway
- Resolver DNS

Interfaces

The main purpose of a network interface is to provide a route through which local data can be sent and remote data can be received. The the operating system uses the loopback interface when it needs to establish a connection with itself

ifconfig

Transmission Control Protocol (TCP) provides connection-oriented service between two applications exchanging data. To configure a network port, or interface, on legacy systems, use the [ifconfig](#) command.

This command is used for the following functions:

- Assigning static IP address
- Viewing current network configuration
- Setting the netmask
- Setting the broadcast address
- Enable/disable network interfaces

```
ifconfig [INTERFACE] [OPTIONS]
```

- ifconfig
- ifconfig device
- ifconfig device IP netmask mask [broadcast brd]

Fedora home example:

```
$ ifconfig
br-38fb360b0b9f: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    inet 192.168.49.1 netmask 255.255.255.0 broadcast 192.168.49.255
    ether 02:42:4c:8f:03:67 txqueuelen 0 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

docker0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    inet 172.17.0.1 netmask 255.255.0.0 broadcast 172.17.255.255
    inet6 fe80::42:a1ff:fe60:ffcc prefixlen 64 scopeid 0x20<link>
    ether 02:42:a1:60:ff:cc txqueuelen 0 (Ethernet)
    RX packets 18127 bytes 952046 (929.7 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 29020 bytes 61685421 (58.8 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

enpls0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    ether c0:18:50:14:97:b3 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    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 48 bytes 5307 (5.1 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 48 bytes 5307 (5.1 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

virbr0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    inet 192.168.122.1 netmask 255.255.255.0 broadcast 192.168.122.255
    ether 52:54:00:6c:e0:04 txqueuelen 1000 (Ethernet)
    RX packets 289 bytes 26902 (26.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
```

```

TX packets 396 bytes 50583 (49.3 KiB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlp0s20f3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.45 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 fe80::2062:d7b2:5a59:f895 prefixlen 64 scopeid 0x20<link>
    ether 84:1b:77:00:78:c8 txqueuelen 1000 (Ethernet)
    RX packets 1241115 bytes 1771487246 (1.6 GiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 551534 bytes 75079937 (71.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

```

$ ifconfig lo
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 48 bytes 5307 (5.1 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 48 bytes 5307 (5.1 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

$ ifconfig enp1s0
enp1s0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    ether c0:18:50:14:97:b3 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

```

$ sudo ifconfig eth0 up

$ sudo ifconfig eth0 192.168.1.3

$ sudo ifconfig eth0 netmask 255.255.255.192

$ sudo ifconfig eth0 broadcast 192.168.1.63

$ sudo ifconfig eth0 down

```

ip link

Debian example:

```

pue@debian:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: ens3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 52:54:00:b3:03:02 brd ff:ff:ff:ff:ff:ff
    inet 172.16.5.2/24 brd 172.16.5.255 scope global dynamic noprefixroute ens3
        valid_lft 2985sec preferred_lft 2985sec
    inet6 fe80::5054:ff:feb3:302/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:55:12:02:45 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
        valid_lft forever preferred_lft forever

```

nmcli device

```
# nmcli device
DEVICE    TYPE        STATE         CONNECTION
ens3      ethernet    conectado     Wired connection 1
docker0    bridge      sin gestión   --
lo         loopback    sin gestión   --
```

Setting the hostname

The hostname is used to identify the system by applications such as web servers:

- /etc/hostname
- /etc/sysconfig/network (legacy)

hostname command

- hostname [-s -f]
- short hostname / FQDN Fully Qualified Domain Name
- setting the hostname using the hostname command results in a change that is only persistent until the next system boot.

```
$ hostname
localhost.localdomain

$ hostname
buster-64
```

hostnamectl command

Systemd systems use an alternative to the hostname command, the hostnamectl command. provides additional categories for hostnames; static, pretty, and transient:

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.

- `hostnamectl`
- `hostnamectl status`
- `hostnamectl set-hostname`

If neither the `--pretty` nor `--transient` option is used, then all three hostname types will be set to the given name. To set the static hostname, but not the pretty and transient names, the option `--static` should be used instead. In all cases, only the static hostname is stored in the `/etc/hostname` file.

```
$ hostnamectl
  Static hostname: localhost.localdomain
        Icon name: computer-laptop
        Chassis: laptop
        Machine ID: f8bfff953853f56b4963ed3aa06461c80
        Boot ID: 022e36fa92be4213a4ea82fa4eb8d128
  Operating System: Fedora 32 (Workstation Edition)
        CPE OS Name: cpe:/o:fedoraproject:fedora:32
        Kernel: Linux 5.11.22-100.fc32.x86_64
  Architecture: x86-64
```

```
$ sudo hostnamectl set-hostname mylaptop

$ hostname
mylaptop

$ sudo hostnamectl set-hostname mylaptop.edt.org

$ hostname
mylaptop.edt.org

$ hostname -s
mylaptop

$ cat /etc/hostname
mylaptop.edt.org
```

Configuring DNS

- static (files) : `/etc/hosts`
- dinàmic (dns): `/etc/resolv.conf`
- `/etc/nsswitch`

The DNS (Domain Name System) is the mapping table for the internet, allowing any computer or device to access by using a name instead of an IP address. The DNS implementation is based on a distributed database of network names and IP addresses and query interfaces to retrieve information.

[/etc/hosts](#)

- static hostnames resolution

```
$ cat /etc/hosts
127.0.0.1    localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.122.28 centos20
172.17.0.1   ldap.edt.org
```

[/etc/resolv.conf](#)

- The `/etc/resolv.conf` file is the configuration file for DNS resolvers (the client DNS part). The information in this file is normally set up by network initialization scripts.
- The configuration directives used in this file are:
 - `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
 - `options`
 - Used to modify the resolver's internal variables using certain keywords
 - E.g. `attempts: 3` will set the retry count for querying the name servers to 3

```
$ cat /etc/resolv.conf
# Generated by NetworkManager
search Home edt.org
nameserver 80.58.61.250
nameserver 80.58.61.254
```

[/etc/nsswitch](#)

The Name Service Switch (NSS) is used by the system administrator to specify which name information source (i.e., local files, LDAP, etc.) to use for different categories (i.e., `passwd`, `shadow`, `hosts`, etc.), and in which order the sources are searched.

The `/etc/nsswitch.conf` file is used to store the information used for name service switching. It is a text file with columns that contain the following information:

- Database name
- Lookup order of sources
- Actions permitted for the lookup result

Example fragment of /etc/nsswitch

```
passwd:      sss files systemd
shadow:      files sss
group:        sss files systemd

#hosts:       db files nisplus nis dns
hosts:        files mdns4_minimal [NOTFOUND=return] dns myhostname
```

By changing the order of the name services listed for a particular database, like hosts, the administrator could change whether the local /etc/hosts file is consulted before or after the DNS servers listed in /etc/resolv.conf.

```
hosts:        files dns
hosts:         dns files
hosts: dns [NOTFOUND=return] files
```

Routing Table

outing tables are used by the kernel to store information about how to reach a network directly or indirectly. The [route](#) and [ip route](#) command is used to view, as well as update, the IP routing table.

Static routes in the kernel's routing table can be set using the route or ip route commands.

[route](#) command

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

```
$ route
Kernel IP routing table
Destination    Gateway         Genmask         Flags Metric Ref    Use Iface
default        _gateway        0.0.0.0         UG    600    0      0 wlp0s20f3
172.17.0.0     0.0.0.0         255.255.0.0     U      0      0      0 docker0
192.168.1.0    0.0.0.0         255.255.255.0   U      600    0      0 wlp0s20f3
192.168.49.0   0.0.0.0         255.255.255.0   U      0      0      0 br-38fb360b0b9f
192.168.122.0  0.0.0.0         255.255.255.0   U      0      0      0 virbr0
```

```
$ sudo route add -net 192.56.78.0 netmask 255.255.255.0 gw 192.168.1.1 dev eth0
```



```
$ sudo route add default gw 192.168.1.1

$ sudo route del -net 192.56.78.0 netmask 255.255.255.0 gw 192.168.1.1 dev eth0

$ sudo route del default gw 192.168.1.1
```

Network Interfaces

The term network interface refers to the point of connection between a computer and a network. It can be implemented in either hardware or software. The network interface card (NIC) is an example of the hardware interface, while the loopback interface (127.0.0.1) is an example of the software interface.

Examine network interface information:

```
$ lspci -v -s 01:00.0
01:00.0 Ethernet controller: Realtek Semiconductor Co., Ltd. RTL8111/8168/8411 PCI
Express Gigabit Ethernet Controller (rev 15)
Subsystem: Lenovo Device 3852
Flags: bus master, fast devsel, latency 0, IRQ 16
I/O ports at 2000 [size=256]
Memory at 81304000 (64-bit, non-prefetchable) [size=4K]
Memory at 81300000 (64-bit, non-prefetchable) [size=16K]
Capabilities: <access denied>
Kernel driver in use: r8169
Kernel modules: r8169

$ modinfo r8169
filename:
/lib/modules/5.11.22-100.fc32.x86_64/kernel/drivers/net/ethernet/realtek/r8169.ko.xz
firmware:      rtl_nic/rtl8125b-2.fw
firmware:      rtl_nic/rtl8125a-3.fw
...

$ lsmod | grep r8169
r8169          102400  0

$ lshw -c network
description: Ethernet interface
product: RTL8111/8168/8411 PCI Express Gigabit Ethernet Controller
vendor: Realtek Semiconductor Co., Ltd.
physical id: 0
bus info: pci@0000:01:00.0
logical name: enp1s0
version: 15
serial: c0:18:50:14:97:b3
capacity: 1Gbit/s
width: 64 bits
clock: 33MHz
capabilities: bus_master cap_list ethernet physical tp mii 10bt 10bt-fd 100bt
100bt-fd 1000bt-fd autonegotiation
configuration: autonegotiation=on broadcast=yes driver=r8169
driverversion=5.11.22-100.fc32.x86_64 firmware=rtl8168h-2_0.0.2 02/26/15 latency=0
link=no multicast=yes port=twisted pair
resources: irq:16 ioport:2000(size=256) memory:81304000-81304fff
memory:81300000-81303fff

$ dmesg | grep r8169
[ 7.685183] libphy: r8169: probed
[ 7.685348] r8169 0000:01:00.0 eth0: RTL8168h/8111h, c0:18:50:14:97:b3, XID 541, IRQ
```

```

129
[ 7.685352] r8169 0000:01:00.0 eth0: jumbo features [frames: 9194 bytes, tx
checksumming: ko]
[ 7.780189] r8169 0000:01:00.0 enp1s0: renamed from eth0
[ 12.117117] Generic FE-GE Realtek PHY r8169-100:00: attached PHY driver
(mii_bus:phy_addr=r8169-100:00, irq=IGNORE)
[ 12.282200] r8169 0000:01:00.0 enp1s0: Link is Down

```

Device information:

RX errors	Number of received packets which were damaged
RX dropped	Number of packets dropped due to reception errors
RX overruns	Number of received packets which experienced data overrun
RX frame	Number of received packets which experienced frame errors
TX errors	Number of packets which experienced data transmission errors
TX dropped	Number of packets dropped due to transmission errors
TX overruns	Number of transmitted packets which experienced data overrun
TX carriers	Number of transmitted packets which experienced loss of carriers
TX collisions	Number of transmitted packets which experienced Ethernet collisions possibly due to network congestion

```

$ ifconfig enp1s0
enp1s0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    ether c0:18:50:14:97:b3 txqueuelen 1000  (Ethernet)
    RX packets 0  bytes 0 (0.0 B)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 0  bytes 0 (0.0 B)
    TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

$ ip -s a show enp1s0
2: enp1s0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel state DOWN group
default qlen 1000
    link/ether c0:18:50:14:97:b3 brd ff:ff:ff:ff:ff:ff
    RX: bytes    packets  errors  dropped overrun mcast
         0         0         0        0         0         0
    TX: bytes    packets  errors  dropped carrier collsns
         0         0         0        0         0         0

```

Interface Names

Older Linux distributions named ethernet network interfaces as eth0, eth1, etc., numbered according to the order in which the kernel identifies the devices. The wireless interfaces were named wlan0, wlan1, etc. This naming convention, however, does not clarify which specific ethernet port matches with the interface eth0, for example. Depending on how the hardware was detected, it was even possible for two network interfaces to swap names after a reboot.

In Linux distributions that use the systemd naming scheme, all interface names start with a two-character prefix that signifies the interface type:

en	Ethernet
ib	InfiniBand
sl	Serial line IP (slip)
wl	Wireless local area network (WLAN)
ww	Wireless wide area network (WWA)

From higher to lower priority, the following rules are used by the operating system to name and number the network interfaces:

1. Name the interface after the index provided by the BIOS or by the firmware of embedded devices, e.g. eno1.
2. Name the interface after the PCI express slot index, as given by the BIOS or firmware, e.g. ens1.
3. Name the interface after its address at the corresponding bus, e.g. enp3s5.
4. Name the interface after the interface's MAC address, e.g. enx78e7d1ea46da.
5. Name the interface using the legacy convention, e.g. eth0.

```
network interface enp3s
device address 03:05.0

$ lspci | fgrep Ethernet
03:05.0 Ethernet controller: Realtek Semiconductor Co., Ltd. RTL-8110SC/8169SC Gigabit Ethernet (rev 10)
```

Red Hat Interface Configuration

- /etc/sysconfig/network (legacy)
- /etc/sysconfig/network-scripts/ifcfg-<interface-name>
- static configuration
- dynamic configuration via DHCP

```
# cat /etc/sysconfig/network-scripts/ifcfg-lo
DEVICE=lo
IPADDR=127.0.0.1
NETMASK=255.0.0.0
NETWORK=127.0.0.0
# If you're having problems with gated making 127.0.0.0/8 a martian,
# you can change this to something else (255.255.255.255, for example)
BROADCAST=127.255.255.255
ONBOOT=yes
NAME=loopback
```

```
# cat /etc/sysconfig/network-scripts/ifcfg-enp1s0
TYPE=Ethernet
PROXY_METHOD=none
BROWSER_ONLY=no
BOOTPROTO=dhcp
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_FAILURE_FATAL=no
IPV6_ADDR_GEN_MODE=stable-privacy
NAME=enp1s0
UUID=a5606faa-358c-3f2d-a032-cd423a68f4c7
ONBOOT=yes
AUTOCONNECT_PRIORITY=-999
DEVICE=enp1s0
```

```
# cat /etc/sysconfig/network-scripts/ifcfg-MiFibra-13C3
ESSID=MiFibra-13C3
MODE=Managed
KEY_MGMT=WPA-PSK
SECURITYMODE=open
MAC_ADDRESS_RANDOMIZATION=default
TYPE=Wireless
PROXY_METHOD=none
BROWSER_ONLY=no
BOOTPROTO=dhcp
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_FAILURE_FATAL=no
IPV6_ADDR_GEN_MODE=stable-privacy
NAME=MiFibra-13C3
UUID=02541f08-fdc8-4203-ad21-4a3aef3acd0b
DEVICE=wlp0s20f3
ONBOOT=yes
```

Debian Interface configuration

- /etc/network/interfaces
- /etc/network/interfaces.d

```
pue@debian:~$ cat /etc/network/interfaces
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

root@debian:/home/pue# locate ens3
/var/lib/NetworkManager/dhclient-a8ae621c-26dd-4ff8-a18d-4bfc29f2ebd5-ens3.lease
/var/lib/NetworkManager/dhclient-ens3.conf

root@debian:/home/pue# cat /var/lib/NetworkManager/dhclient-ens3.conf
# Creado por NetworkManager
# Mezclado de /etc/dhcp/dhclient.conf
option rfc3442-classless-static-routes code 121 = array of unsigned integer 8;
#send dhcp-client-identifier 1:0:a0:24:ab:fb:9c;
#send dhcp-lease-time 3600;
#supersede domain-name "fugue.com home.vix.com";
#prepend domain-name-servers 127.0.0.1;
#require subnet-mask, domain-name-servers;
#timeout 60;
#retry 60;
#reboot 10;
#select-timeout 5;
#initial-interval 2;
#script "/sbin/dhclient-script";
#media "-link0 -link1 -link2", "link0 link1";
#reject 192.33.137.209;
send host-name "debian"; # added by NetworkManager

option rfc3442-classless-static-routes code 121 = array of unsigned integer 8;
option ms-classless-static-routes code 249 = array of unsigned integer 8;
option wpad code 252 = string;

request; # override dhclient defaults
also request subnet-mask;
```

```
also request broadcast-address;
also request time-offset;
also request routers;
also request domain-name;
also request domain-name-servers;
also request domain-search;
also request host-name;
also request dhcp6.name-servers;
also request dhcp6.domain-search;
also request dhcp6.fqdn;
also request dhcp6.sntp-servers;
also request netbios-name-servers;
also request netbios-scope;
also request interface-mtu;
also request rfc3442-classless-static-routes;
also request ntp-servers;
also request ms-classless-static-routes;
also request static-routes;
also request wpad;
also request root-path;
```

```
# cat /etc/networks
default      0.0.0.0
loopback     127.0.0.0
link-local   169.254.0.0
```

Example dynamic configuration:

```
auto lo
iface lo inet loopback

auto enp3s5
iface enp3s5 inet dhcp
```

Example static configuration

```
iface enp3s5 inet static
    address 192.168.1.2/24
    gateway 192.168.1.1
```

Network Tools: NetworkManager and iproute2

Network Manager

NetworkManager provides automatic detection and configuration of network interfaces on a Linux system. It works for both wired and wireless interfaces as well as having support for some modems and Virtual Private Network (VPN) connections.

NetworkManager uses the network management command line interface [nmcli](#) tool to manage network connections and display information about devices on a network. The nmcli command uses the following syntax:

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

The OBJECT field can be one of the following:

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.

```
[root@mylaptop ~]# nmcli general
STATE      CONNECTIVITY  WIFI-HW  WIFI    WWAN-HW  WWAN
connected  full           enabled  enabled  enabled  enabled
```

```
root@debian:/home/pue# nmcli device
DEVICE     TYPE      STATE      CONNECTION
ens3       ethernet  conectado  Wired connection 1
docker0    bridge    sin gestión --
lo         loopback  sin gestión --
```

```
[root@localhost ~]# nmcli device
DEVICE     TYPE      STATE      CONNECTION
wlp0s20f3  wifi      connected  MOVISTAR_5AF0
docker0    bridge    connected  docker0
br-38fb360b0b9f  bridge    connected  br-38fb360b0b9f
virbr0     bridge    connected  virbr0
p2p-dev-wlp0s20f3  wifi-p2p  disconnected --
enpls0     ethernet  unavailable --
vboxnet0   ethernet  unmanaged  --
lo         loopback  unmanaged  --
virbr0-nic  tun       unmanaged  --
```

```
root@debian:/home/pue# nmcli device show ens3
GENERAL.DEVICE: ens3
GENERAL.TYPE: ethernet
GENERAL.HWADDR: 52:54:00:B3:03:02
GENERAL.MTU: 1500
GENERAL.STATE: 100 (conectado)
GENERAL.CONNECTION: Wired connection 1
GENERAL.CON-PATH: /org/freedesktop/NetworkManager/ActiveConnection/1
WIRED-PROPERTIES.CARRIER: encendido
IP4.ADDRESS[1]: 172.16.5.2/24
IP4.GATEWAY: 172.16.5.254
IP4.ROUTE[1]: dst = 0.0.0.0/0, nh = 172.16.5.254, mt = 100
IP4.ROUTE[2]: dst = 172.16.5.0/24, nh = 0.0.0.0, mt = 100
IP4.DNS[1]: 172.16.5.254
IP4.DOMAIN[1]: class.local
IP6.ADDRESS[1]: fe80::5054:ff:feb3:302/64
IP6.GATEWAY: --
IP6.ROUTE[1]: dst = fe80::/64, nh = ::, mt = 100
IP6.ROUTE[2]: dst = ff00::/8, nh = ::, mt = 256, table=255
```

```
[root@localhost ~]# nmcli -p connection show enpls0
```

```
=====
Connection profile details (enpls0)
```

```

=====
connection.id:                               enpls0
connection.uuid:                             a5606faa-358c-3f2d-a032-cd423a68f4c7
connection.stable-id:                        --
connection.type:                             802-3-ethernet
connection.interface-name:                   enpls0
connection.autoconnect:                      yes
connection.autoconnect-priority:             -999
connection.autoconnect-retries:              -1 (default)
connection.multi-connect:                    0 (default)
connection.auth-retries:                     -1
connection.timestamp:                        1636205619
...

```

```
root@debian:/home/pue# nmcli -p connection show
```

```
=====
Perfiles de conexiones NetworkManager
=====
```

NAME	UUID	TYPE	DEVICE
Wired connection 1	a8ae621c-26dd-4ff8-a18d-4bfc29f2ebd5	ethernet	ens3

```
root@debian:/home/pue# nmcli con add con-name eth1 ifname eth1 type ethernet \ip4
10.0.2.18/24 gw4 10.0.2.2
```

```
Conexión 'eth1' (aaaf6945-e063-43a4-aff7-f94a9a218384) agregada con éxito.
```

```
root@debian:/home/pue# nmcli -p connection show eth1
```

```
=====
Detalles de perfil de conexiones (eth1)
=====
```

```

connection.id:                               eth1
connection.uuid:                             aaaf6945-e063-43a4-aff7-f94a9a218384
connection.stable-id:                        --
connection.type:                             802-3-ethernet
connection.interface-name:                   eth1
connection.autoconnect:                      sí
connection.autoconnect-priority:             0
connection.autoconnect-retries:              -1 (default)
connection.multi-connect:                    0 (default)
connection.auth-retries:                     -1
connection.timestamp:                        0
connection.read-only:                        no

```

Wireless interfaces

To configure a wireless interface, first determine the name by using the nmcli d command to view interface devices.

```

$ nmcli radio wifi on

$ nmcli device wifi list

$ nmcli d wifi connect Chippewa_Guest password 1882

```

```
[root@localhost ~]# nmcli device wifi list
```

IN-USE	BSSID	SSID	MODE	CHAN	RATE	SIGNAL	BARS	SECURITY
	78:94:B4:C2:0F:81	vodafone0F80	Infra	5	130 Mbit/s	45	█	WPA2
	78:94:B4:C2:0F:85	vodafone0F80_5G	Infra	48	540 Mbit/s	37	█	WPA2
	78:29:ED:0A:09:84	MOVISTAR_0983	Infra	9	130 Mbit/s	35	█	WPA2
	F4:69:42:6B:7D:BF	MOVISTAR_7DB0	Infra	1	130 Mbit/s	32	█	WPA2
	78:81:02:68:79:51	vodafone7950	Infra	11	130 Mbit/s	30	█	WPA2
	88:5D:FB:C6:C0:1B	MIWIFI_2G_4FrA	Infra	1	130 Mbit/s	29	█	WPA1 WPA2
	78:29:ED:0A:09:91	MOVISTAR_PLUS_0983	Infra	40	540 Mbit/s	27	█	WPA2

```

7C:8F:DE:20:AB:81 sercommBB1824 Infra 1 195 Mbit/s 25 WPA2
8A:5D:FB:C6:C0:1C MIWIFI_2G_4FrA Infra 36 540 Mbit/s 24 WPA1 WPA2
88:5D:FB:C6:C0:1C MIWIFI_5G_4FrA Infra 36 540 Mbit/s 24 WPA1 WPA2
44:FE:3B:39:DC:BF -- Infra 100 540 Mbit/s 19 WPA2
7C:8F:DE:20:AB:85 sercommBB1824 Infra 64 540 Mbit/s 17 WPA2
60:8D:26:EC:7B:BB Livebox6-7BBC Infra 1 195 Mbit/s 15 WPA2

$ nmcli device wifi connect timofon

$ nmcli device wifi connect timofon password MyPassword

$ nmcli device wifi connect timofon password MyPassword hidden yes

$ nmcli device wifi connect Hypnotoad password MyPassword ifname wlo1

[root@mylaptop ~]# nmcli connection show
NAME                UUID                                TYPE      DEVICE
timfon              20925a66-11fe-42e5-800a-bc82d69b7bd7 wifi      wlp0s20f3
br-38fb360b0b9f    d3c0ec45-84cb-43bb-8cfe-4cd4a358aadb bridge    br-38fb360b0b9f
docker0            5ebaa9e9-0d84-4826-ada4-87bf65a8a706 bridge    docker0
virbr0             6a3ad5cd-49ff-4aec-ae0d-f86499fc763a bridge    virbr0
enpls0             a5606faa-358c-3f2d-a032-cd423a68f4c7 ethernet  --
MiFibra-13C3       02541f08-fdc8-4203-ad21-4a3aef3acd0b wifi      --

$ nmcli connection down timofon
$ nmcli connection up timofon
$ nmcli connection delete timofon

$ nmcli device disconnect wlo2
$ nmcli device connect wlo2

$ nmcli radio wifi off
$ nmcli radio wifi on

```

```

[root@mylaptop ~]# iwconfig wlp0s20f3
wlp0s20f3 IEEE 802.11 ESSID:"MOVISTAR_5AF0"
Mode:Managed Frequency:2.437 GHz Access Point: E2:41:36:0E:5A:F0
Bit Rate=86.7 Mb/s Tx-Power=22 dBm
Retry short limit:7 RTS thr:off Fragment thr:off
Encryption key:off
Power Management:on
Link Quality=50/70 Signal level=-60 dBm
Rx invalid nwid:0 Rx invalid crypt:0 Rx invalid frag:0
Tx excessive retries:1 Invalid misc:29 Missed beacon:0

```

```

[root@mylaptop ~]# iwlist
Usage: iwlist [interface] scanning [essid NNN] [last]
[interface] frequency
[interface] channel
[interface] bitrate
[interface] rate
[interface] encryption
[interface] keys
[interface] power
[interface] txpower
[interface] retry
[interface] ap
[interface] accesspoints
[interface] peers
[interface] event
[interface] auth
[interface] wpakeys
[interface] genie
[interface] modulation

```


iproute2 tools

Commands:

- `ip link` Configure links
- `ip address` Configure address
- `ip route` Manage routing tables
- `ip neighbors` Display and manage neighbors
- `ip tunnel` Manage tunnels
- `ip maddr` Manage multicast
- `ip -s`
- `ss / netstat`

< next chapter >

Systemd-networkd

The `systemd-networkd` daemon functions through configuration files, which reside in the `/usr/lib/systemd/network/`, `/run/systemd/network`, and `/etc/systemd/network` directories.

- `systemd-networkd`
- `systemd-resolved`
- `resolvectl`

Name resolution services on systems that use `systemd` are handled by `systemd-resolved`. This `systemd` service tells local applications where to find domain name information on a network. The `systemd-resolved` service can operate in four different modes, which are:

- Using a `systemd` DNS stub file located at `/run/systemd/resolve/stub-resolv.conf`.
- Preserving the legacy `resolv.conf` file we learned about earlier in this chapter.
- Automatic configuration with a network manager.
- Manual, or local DNS stub mode where alternate DNS servers are provided in the `resolved.conf` file.

The purpose of each configuration file depends on its suffix. File names ending in `.netdev` are used by `systemd-networkd` to create virtual network devices, such as bridge or tun devices. Files ending in `.link` set low-level configurations for the corresponding network interface. `systemd-networkd` detects and configures network devices automatically as they appear — as well as ignore devices already configured by other means — so there is little need to add these files in most situations.

The most important suffix is `.network`. Files using this suffix can be used to setup network addresses and routes.

Example Exercises

1. What commands can be used to list the network adapters present in the system?
2. What is the type of a network adapter whose interface name is wlo1?
3. What entry in `/etc/network/interfaces` configures the interface `eno1` to obtain its IP settings with DHCP?
4. IN Red Hat systems, which file configures the interface `lo`? and the interface `eth0` ?
5. Configure `nsswitch` to do local name resolution before DNS resolution.
6. Which file contains the configuration of the resolver?
7. Show the hostname information, the tree kind of hostnames.
8. Establish only the static hostname to `myserver`.
9. How do we associate the local address `127.0.0.1` to the local name `myprettyhost`?
10. What `nmcli` subcommand shows interface detailed information?
11. How can a user run the command `nmcli` to delete an unused connection named `Hotel Internet`?
12. Realitza els exercicis indicats a: [109.2 Persistent network configuration](#)
13. Realitza els exercicis del Question-Topics 109.2