Network Tools & commands
Part - I

Topic:

CAT-5/CAT-6 cable preparation with RJ-45 connector; both straight and cross cabling.

Explanation:

Materials Required: To create a network cable, we will first need the equipment listed below.

1. **Unshielded twisted pair (UTP) patch cable i.e., Cat 5, Cat 5e, Cat 6, or Cat 7 cable** - This cabling can be purchased in large spindles at stores that specialize in cabling. Cat5 cabling is the most commonly used cable used today for networks.

2. **RJ-45 connectors** - These connectors can be purchased at most electronic stores and computer stores and usually come in bulk packages. It is always a good idea to get more than you think you need.

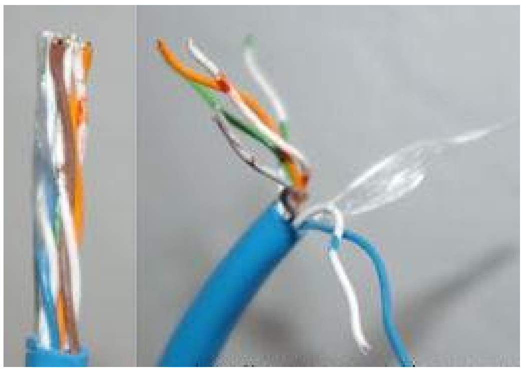
3. **Crimping tool** - These tools are often purchased at electronic stores. To create a network cable you need a crimper that is capable of crimping a RJ-45 cable

4. **Wire stripper or Knife** - If we plan on making several networks cables, we should also consider getting a wire stripper cable of stripping Cat 5, Cat 6, or our cable of choice. If we do not plan on creating many network cables, a knife will suffice. To prevent potential issues, we recommend a wire stripper.

5. **Cable Tester** (optional, but recommended)

Once we have the necessary equipment needed to create a network cable, we need to determine the type of network cable we want to create. There are two major network cables: a straight through cable and a crossover cable.

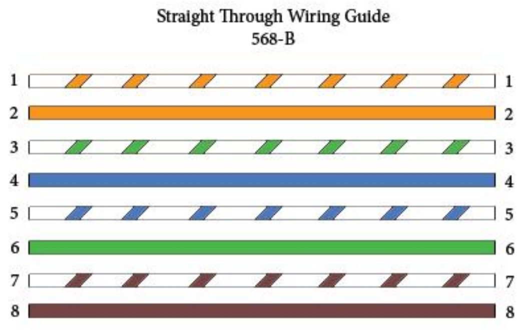
Straight-through, Crossover, and Rollover Wiring



When talking about cable pinouts, we often get questions as to the difference in Straight-through, Crossover, and Rollover wiring of cables and the intended use for each type of cable. These terms are referring to the way the cables are wired (which pin on one end is connected to which pin on the other end). Below are three different types of wiring:

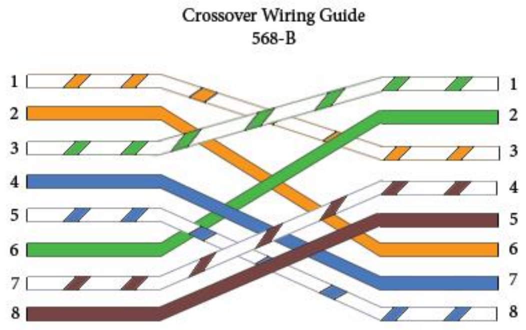
Straight-Through Wired Cables:

Straight-Through refers to cables that have the pin assignments on each end of the cable. In other words, Pin 1 connector A goes to Pin 1 on connector B, Pin 2 to Pin 2, etc. Straight-Through wired cables are most commonly used to connect a host to a client. When we talk about cat5e patch cables, the Straight-Through wired cat5e patch cable is used to connect computers, printers, and other network client devices to the router switch or hub (the host device in this instance).



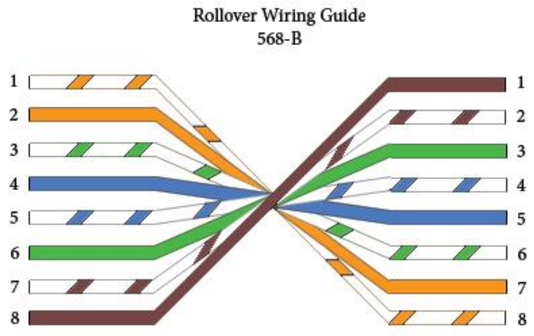
Crossover Wired Cables:

Crossover wired cables (commonly called crossover cables) are very much like Straight-Through cables with the exception that TX and RX lines are crossed (they are at opposite positions on either end of the cable. Using the 568-B standard as an example, we see that Pin 1 on connector A goes to Pin 3 on connector B. Pin 2 on connector A goes to Pin 6 on connector B, etc. Crossover cables are most commonly used to connect two hosts directly. Examples would be connecting a computer directly to another computer, connecting a switch directly to another switch, or connecting a router to a router.



Rollover Wired Cables:

Rollover wired cables, most commonly called rollover cables, have opposite Pin assignments on each end of the cable or, in other words, it is "rolled over." Pin 1 of connector A would be connected to Pin 8 of connector B. Pin 2 of connector A would be connected to Pin 7 of connector B and so on. Rollover cables, sometimes referred to as Yost cables are most commonly used to connect to a device's console port to make programming changes to the device. Unlike crossover and straight-wired cables, rollover cables are not intended to carry data but instead create an interface with the device.



Now days, LAN card has intelligence, so that both cables can work. They have a feature on lots of switches and hubs etc called "auto-mdix" or "auto mdi/mdix", that is the new thing where it doesn’t matter what kind of cable you use, it will just auto detect the proper connection type no matter which cable you use.

The cable can be categorized as Cat 5, Cat 5e, and Cat 6 UTP cable.
Cat 5 UTP cable can support 10/100 Mbps Ethernet network, whereas Cat 5e and Cat 6 UTP cable can support Ethernet network running at 10/100/1000 Mbps. You might hear about Cat 3 UTP cable, it's not popular anymore since it can only support 10 Mbps Ethernet network.

Straight and crossover cable can be Cat3, Cat 5, Cat 5e or Cat 6 UTP cable, the only difference is each type will have different wire arrangement in the cable for serving different purposes. Ethernet network cables are straight and crossover cable. This Ethernet network cable is made of 4 pair high performance cable that consists of twisted pair conductors that used for data transmission. Both end of cable is called RJ45 connector. There are two types of network cables commonly used in PC networks - Straight-through and cross-over.

Straight Cable (T568A)

Usually use straight cable to connect different type of devices. This type of cable will be used most of the time and can be used to:

1) Connect a computer to a switch/hub's normal port.
2) Connect a computer to a cable/DSL modem's LAN port.
3) Connect a router's WAN port to a cable/DSL modem's LAN port.
4) Connect a router's LAN port to a switch/hub's uplink port. (Normally used for expanding network)
5) Connect two switches/hubs with one of the switch/hubs using an uplink port and the other one using normal port.

If you need to check how straight cable looks like, it's easy. Both sides (side A and side B) of cable have wire arrangement with same color.

Crossover Cable (T568A & T568B)

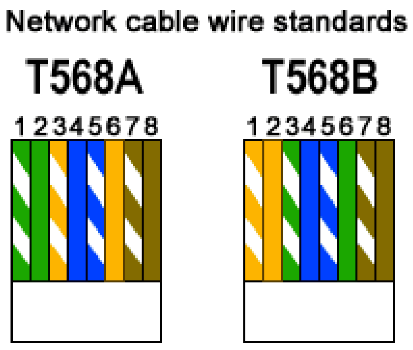
Sometimes you will use crossover cable, it's usually used to connect same type of devices. A crossover cable can be used to:

1) Connect two computers directly.
2) Connect a router's LAN port to a switch/hub's normal port. (Normally used for expanding network)
3) Connect two switches/hubs by using normal port in both switches/hubs.

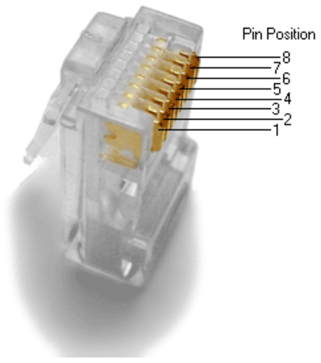
In you need to check how crossover cable looks like, both side (side A and side B) of cable have wire arrangement with following different color. This cable (either straight cable or cross cable) has total 8 wires (or we can say lines), i.e. four twisted pairs (4x2=8) with different color codes.

Once you have determined the type of network cable, strip the cable. We recommend stripping at least a half of an inch off of the cable to expose the inner wires. Don't be worried about stripping too much of the network cable jacket off since you can always cut the wires down more if needed later. After the network cable jacket is removed, separate the wires in the cable so they can be put into the RJ-45 connector.

The CAT5 twisted-pair cables consist of four twisted wires, each color-coded; one a solid color, and one a striped color. As seen below, most network cables consist of a green, blue, orange, and brown pair of cables. There are two cable standards T568A and T568B, each twisted-pair must be broken apart to create the layout as shown above. If you want to create a straight through cable, *both* ends of the cable should be identical and should match the T568A example shown above. If you want to create a crossover cable, one end of the cable should match T568A, and one should match T568B. The two standards for wiring Ethernet cables are T568A and T568B. T568B is the most common and is what we'll be using for our straight Ethernet cable.



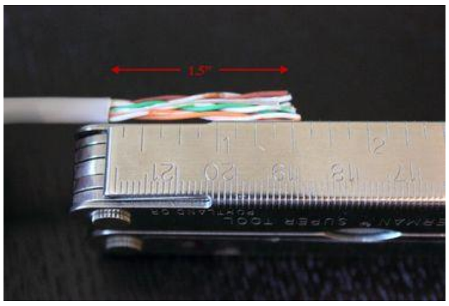
Procedure or steps for connection:



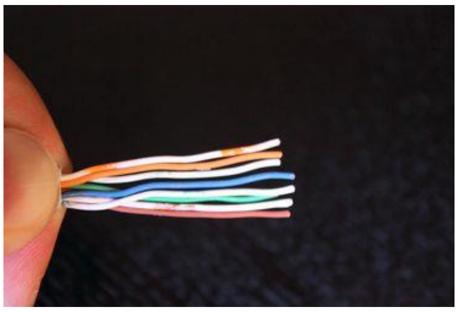
After separating the ends of the cable to match one of the above examples, insert the cables into the RJ-45 connector and crimp (using a crimping tool) the connector

There are four pairs of wires in an Ethernet cable, and an Ethernet connector (8P8C) has eight pin slots. Each pin is identified by a number, starting from left to right, with the clip facing away from you.

Step 1: Strip the cable jacket about 1.5 inch down from the end.

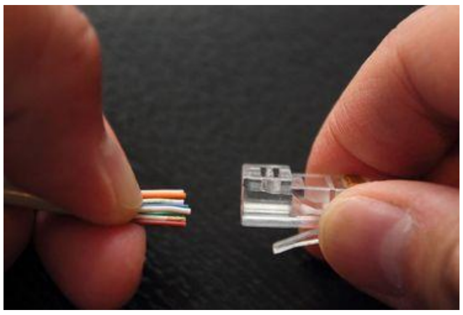


Step 2: Spread the four pairs of twisted wire apart. For Cat 5e, you can use the pull string to strip the jacket farther down if you need to, then cut the pull string. Cat 6 cables have a spine that will also need to be cut.

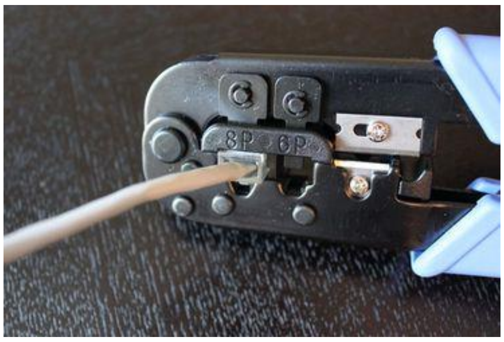


Step 3: Untwist the wire pairs and neatly align them in the T568B orientation. Be sure not to untwist them any farther down the cable than where the jacket begins; we want to leave as much of the cable twisted as possible.

Step 4: Cut the wires as straight as possible, about 0.5 inch above the end of the jacket.



Step 5: Carefully insert the wires all the way into the modular connector, making sure that each wire passes through the appropriate guides inside the connector.

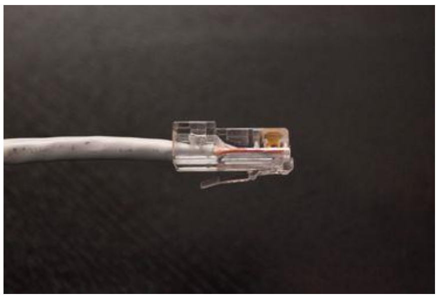


Step 6: Push the connector inside the crimping tool and squeeze the crimper all the way down.

Step 7: Repeat steps 1-6 for the other end of the cable.

Step 8: To make sure you've successfully terminated each end of the cable, use a cable tester to test each pin.

When you're all done, the connectors should look like this:



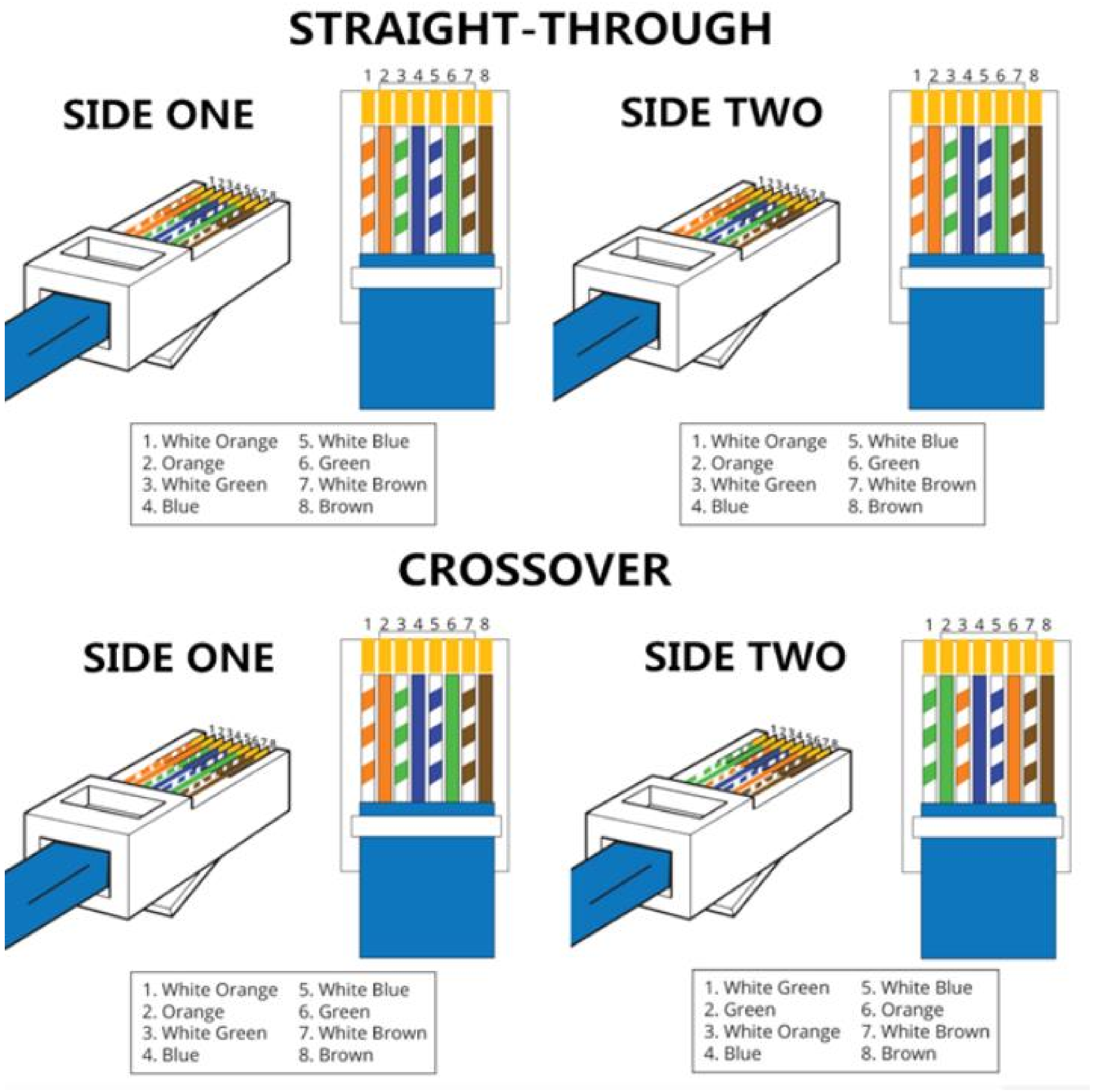
That's it. For crossover cables, simply make one end of the cable a T568A and the other end a T568B. Now you can make Ethernet cables of any length, fix broken connectors, or make yourself a crossover cable. Happy crimping!

Purpose:

Purpose of the cross cable is RX (receiving terminal) connects to TX (transmitting) of one pc to another PC and vice versa. As we use two PCs (same devices), straight cable will connect TX to TX and RX to RX of two computers, so cross cable is required. If you use HUB or switch, then straight cable will work because it has internal arrangement like cross cable. We use cross cable to connect two similar devices.

A straight cable will not work to connect two computers together. Crossover used to connect to PCs directly together, also used for connecting networking devices together like Switch to Switch etc.

Straight cables connect two DIFFERENT types of devices. Whereas crossover cables connect two of the SAME type.



Uses and Application:

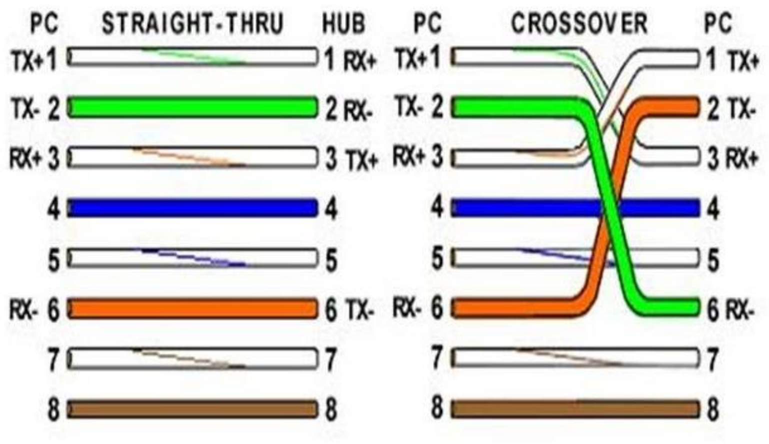
	Hub	Switch	Router	Workstation
Hub	Crossover	Crossover	Straight	Straight
Switch	Crossover	Crossover	Straight	Straight
Router	Straight	Straight	Crossover	Crossover
Workstation	Straight	Straight	Crossover	Crossover

Communication between Switches and Workstations:

When a workstation connects to a LAN, it transmits data independently of the other devices connected to the LAN media. The workstation simply transmits data frames from a NIC to the network medium.

If desired, the workstation can be attached directly to another workstation by using a crossover cable. Crossover cables connect the following devices:

- Workstation to workstation
- Switch to switch
- Switch to hub
- Hub to hub
- Router to router
- Router to PC



Straight-through cables connect the following devices:

- Switch to router
- Switch to workstation or server
- Hub to workstation or server

Many modern switches now automatically adjust the port pinout to support the particular cable attached, whether it is a crossover or straight-through cable.

Switches, which are Layer 2 devices, use intelligence to learn the MAC addresses of the devices that are attached to its ports. This data is entered into a switching table. After the table is complete, the switch can read the destination MAC address of an incoming data frame on a port and immediately forward it. Until a device transmits, the switch does not know its MAC address. Switches provide significant scalability on a network. Switches are normally connected to each other by way of trunk links.

Network Tools & commands
Part - II

Topic:

IP address configuration (both Static and DHCP) on Linux and Windows systems.

IP address Configuration (static and DHCP) on Ubuntu 18.04:

Generally, IP addresses are assigned dynamically by our router DHCP server. Setting a static IP address on our Ubuntu machine may be required in different situations, such as configuring port forwarding or running a media server on our network.

Configuring Static IP address using DHCP

The easiest and the recommended way to assign a static IP address to a device on our LAN is by setting up a Static DHCP on our router. Static DHCP or DHCP reservation is a feature found on most routers which makes the DHCP server to automatically assign the same IP address to a specific network device, every time the device requests an address from the DHCP server. This works by assigning a static IP to the device's unique MAC address. The steps for configuring a DHCP reservation vary from router to router, and it's advisable to consult the vendor's documentation.

Netplan

Starting with 17.10 release, Netplan is the default network management tool on Ubuntu, replacing the configuration file /etc/network/interfaces that had previously been used to configure the network on Ubuntu.

Netplan uses configuration files in YAML syntax. To configure a network interface with Netplan, you need to create a YAML description for that interface, and Netplan will generate the required configuration files for your chosen renderer tool.

Netplan currently supports two renderers NetworkManager and Systemd-networkd. Network-Manager is mostly used on Desktop machines while the Systemd-networkd is used on servers without a GUI.

Configuring Static IP address on Ubuntu Server

The newer versions of Ubuntu use 'Predictable Network Interface Names' that, by default, start with en[letter][number].

The first step is to identify the name of the ethernet interface we want to configure. To do so use the ip link command, as shown below:

```
ip link
```

The command will print a list of all the available network interfaces. In this case, the name of the interface is

```
ens3:
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT
group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
3: ens3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP mode DEFAULT
group default qlen 1000
    link/ether 56:00:00:60:20:0a brd ff:ff:ff:ff:ff:ff
```

Netplan configuration files are stored in the /etc/netplan directory and have the extension .yaml. We'll probably find one or two YAML files in this directory. The file may differ from setup to setup. Usually, the file is named either 01-netcfg.yaml, 50-cloud-init.yaml, or NN_interfaceName.yaml, but in one's system it may be different.

Open the YAML configuration file with your text editor:

```
sudo nano /etc/netplan/01-netcfg.yaml
```

```
/etc/netplan/01-netcfg.yaml
network:
  version: 2
  renderer: networkd
  ethernets:
    ens3:
      dhcp4: yes
```

Each Netplan Yaml file starts with the network key that has at least two required elements. The first required element is the version of the network configuration format, and the second one is the device type. The device type can be ethernets, bonds, bridges, or vlans.

The configuration above also includes the renderer type. Out of the box, if we installed Ubuntu in server mode, the renderer is configured to use networkd as the back end.

Under the device's type (in this case ethernets), you can specify one or more network interfaces. In this example, we have only one interface ens3 that is configured to obtain IP addressing from a DHCP server dhcp4: yes.

To assign a static IP address to ens3 interface, edit the file as follows:

- Set DHCP to no dhcp4: yes
- Specify the static IP address 192.168.121.199/24. Under addresses: you can add one or more IPv4 or IPv6 IP addresses that will be assigned to the network interface.
- Specify the gateway gateway4: 192.168.121.1
- Under nameservers, set the IP addresses of the nameservers addresses: [8.8.8.8, 1.1.1.1]

```
/etc/netplan/01-netcfg.yaml
network:
  version: 2
  renderer: networkd
  ethernets:
    ens3:
      dhcp4: no
      addresses:
        - 192.168.121.199/24
      gateway4: 192.168.121.1
      nameservers:
          addresses: [8.8.8.8, 1.1.1.1]
```

When editing Yaml files, make sure you follow the YAML code indent standards. If there are syntax errors in the configuration, the changes will not be applied.

Once done save and close the file and apply the changes with:

```
sudo netplan apply
```

Verify the changes by typing:

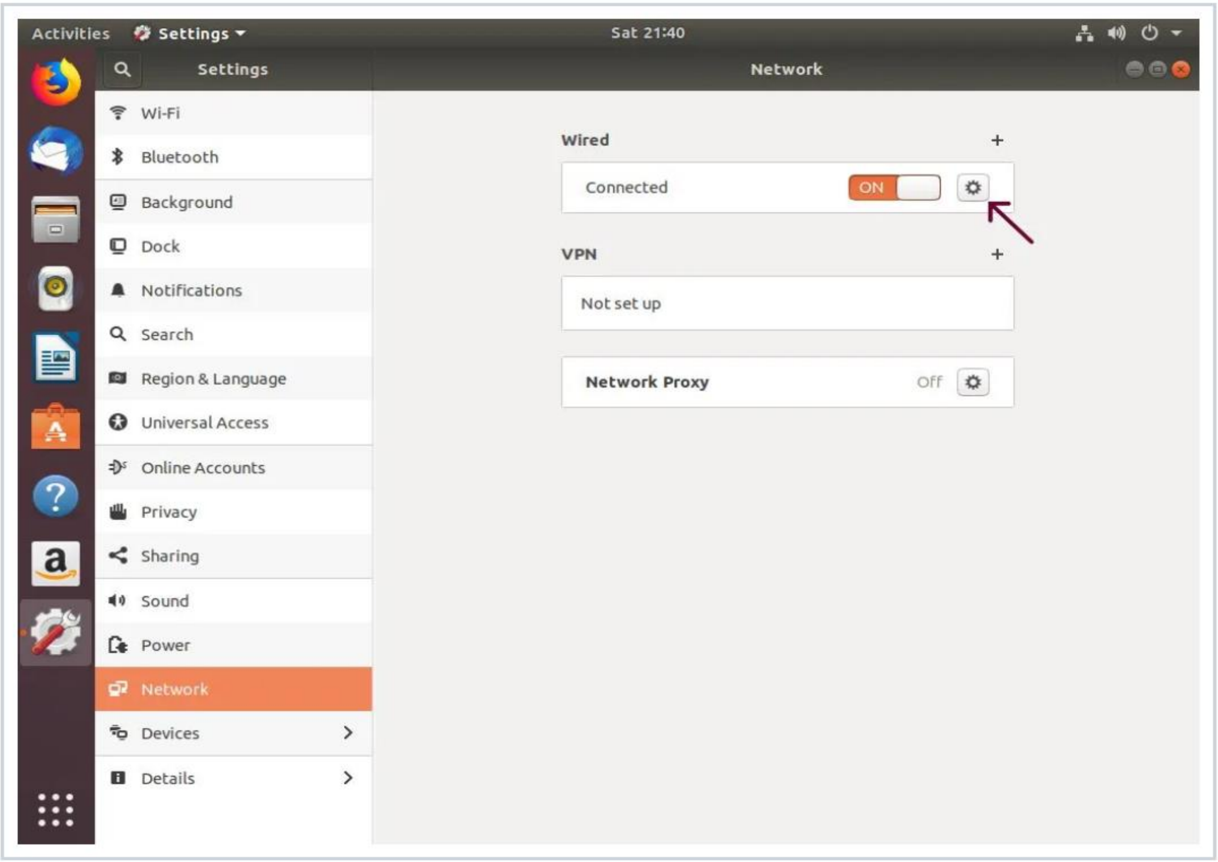
```
ip addr show dev ens3

3: ens3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default
qlen 1000
    link/ether 56:00:00:60:20:0a brd ff:ff:ff:ff:ff:ff
    inet 192.168.121.199/24 brd 192.168.121.255 scope global dynamic ens3
       valid_lft 3575sec preferred_lft 3575sec
    inet6 fe80::5054:ff:feb0:f500/64 scope link
       valid_lft forever preferred_lft forever
```

This is how we assign a static IP to our Ubuntu server.

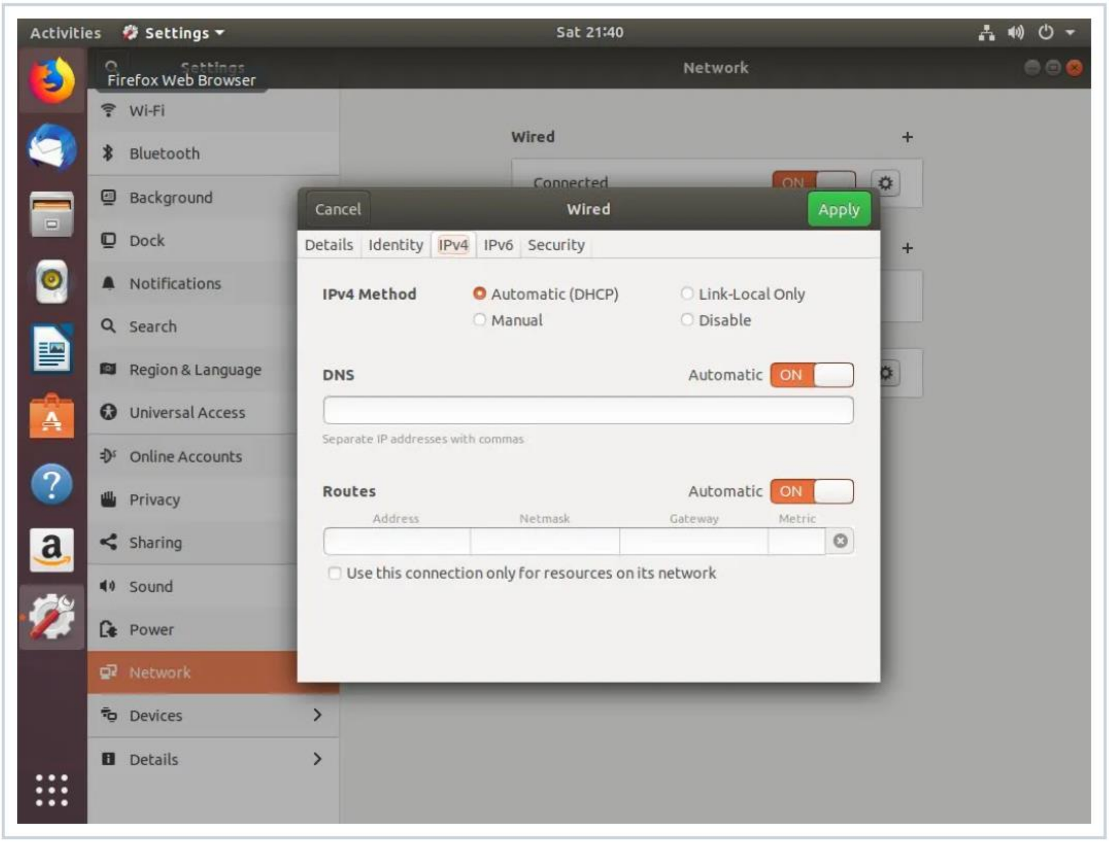
- **Configuring Static IP address on Ubuntu Desktop**

Setting up a static IP address on Ubuntu Desktop computers requires no technical knowledge.

1. In the Activities screen, search for “network” and click on the Network icon. This will open the GNOME Network configuration settings. Click on the cog icon.



2. The Network interface settings dialog box will be opened:



IPv4 Method [✓ Automatic (DHCP)] Link-Local Only Manual Disable

DNS Automatic ON

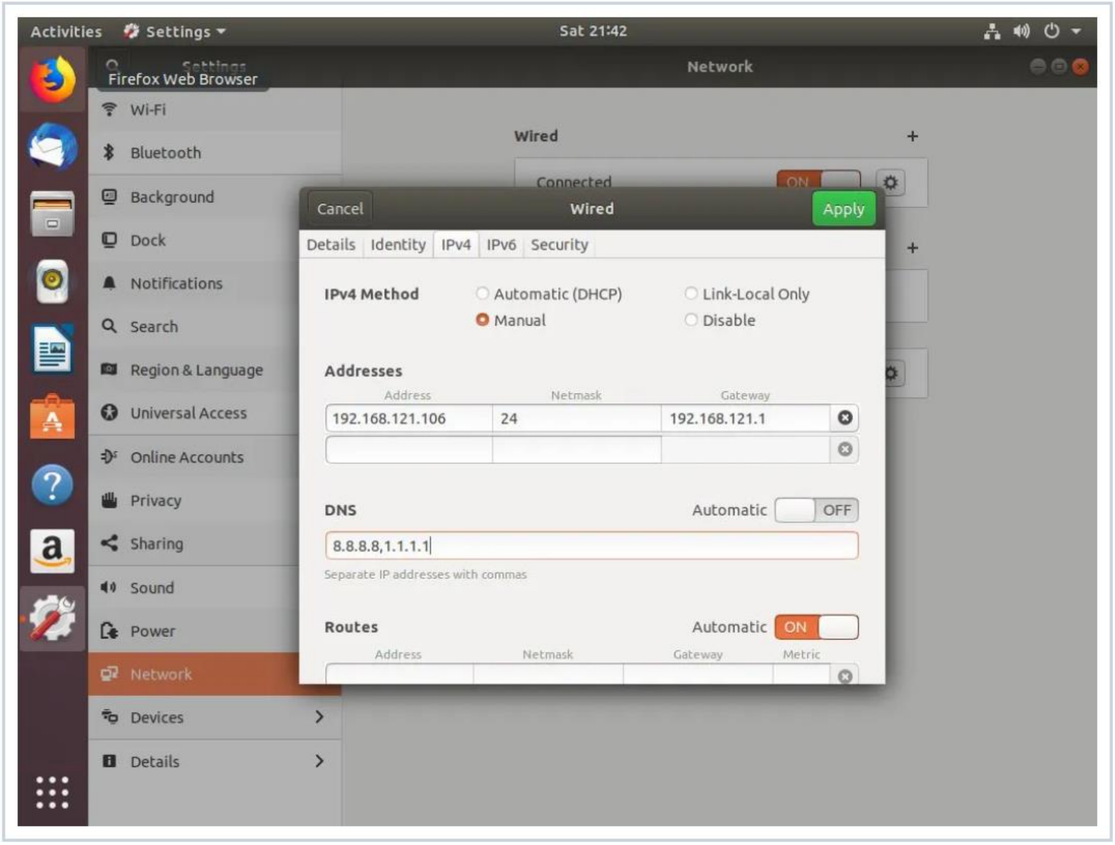
Separate IP addresses with commas

Routes Automatic ON

Address	Netmask	Gateway	Metric

☐ Use this connection only for resources on its network

3. In "IPV4" Method" section, select "Manual" and enter your static IP address, Netmask and Gateway. Once done, click on the "Apply" button.



IPv4 Method Automatic (DHCP) Link-Local Only [✓ Manual] Disable

Addresses

Address	Netmask	Gateway
192.168.121.106	24	192.168.121.1

DNS Automatic OFF

8.8.8.8,1.1.1.1

Separate IP addresses with commas

Routes Automatic ON

Address	Netmask	Gateway	Metric

Now that you have set up a static IP Address, open your terminal either by using the Ctrl+Alt+T keyboard shortcut or by clicking on the terminal icon and verify the changes by typing:

```
ip addr
```

The output will show the interface IP address:

```
...
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group
default qlen 1000
    link/ether 52:54:00:e9:40:f2 brd ff:ff:ff:ff:ff:ff
    inet 192.168.121.106/24 brd 192.168.121.255 scope global dynamic noprefixroute
eth0
       valid_lft 3523sec preferred_lft 3523sec
    inet6 fe80::5054:ff:fee9:40f2/64 scope link
       valid_lft forever preferred_lft forever
```

Explanation and Advantages of DHCP:

Instead of storing network configuration information in local files on each system, DHCP (Dynamic Host Configuration Protocol) enables client systems to retrieve network configuration information each time they connect to the network. A DHCP server assigns IP addresses from a pool of addresses to clients as needed. Assigned addresses are typically temporary, but need not be.

This technique has several advantages over storing network configuration information in local files:

- A new user can set up an Internet connection without having to deal with IP addresses, netmasks, DNS addresses, and other technical details. An experienced user can set up a connection more quickly.
- DHCP facilitates assignment and management of IP addresses and related network information by centralizing the process on a server. A system administrator can configure new systems, including laptops that connect to the network from different locations, to use DHCP; DHCP then assigns IP addresses only when each system connects to the network. The pool of IP addresses is managed as a group on the DHCP server.
- IP addresses can be used by more than one system, reducing the total number of IP addresses needed. This conservation of addresses is important because the Internet is quickly running out of IPv4 addresses. Although a particular IP address can be used by only one system at a time, many end-user systems require addresses only occasionally, when they connect to the Internet. By reusing IP addresses, DHCP lengthens the life of the IPv4 protocol. DHCP applies to IPv4 only, as IPv6 forces systems to configure their IP addresses automatically (called autoconfiguration) when they connect to a network (page 373).

DHCP is particularly useful for administrators who are responsible for maintaining a large number of systems because individual systems no longer need to store unique configuration information. With DHCP, the administrator can set up a master system and deploy new systems with a copy of the master's hard disk. In educational establishments and other open access facilities, the hard disk image may be stored on a shared drive, with each workstation automatically restoring itself to pristine condition at the end of each day.

How DHCP works

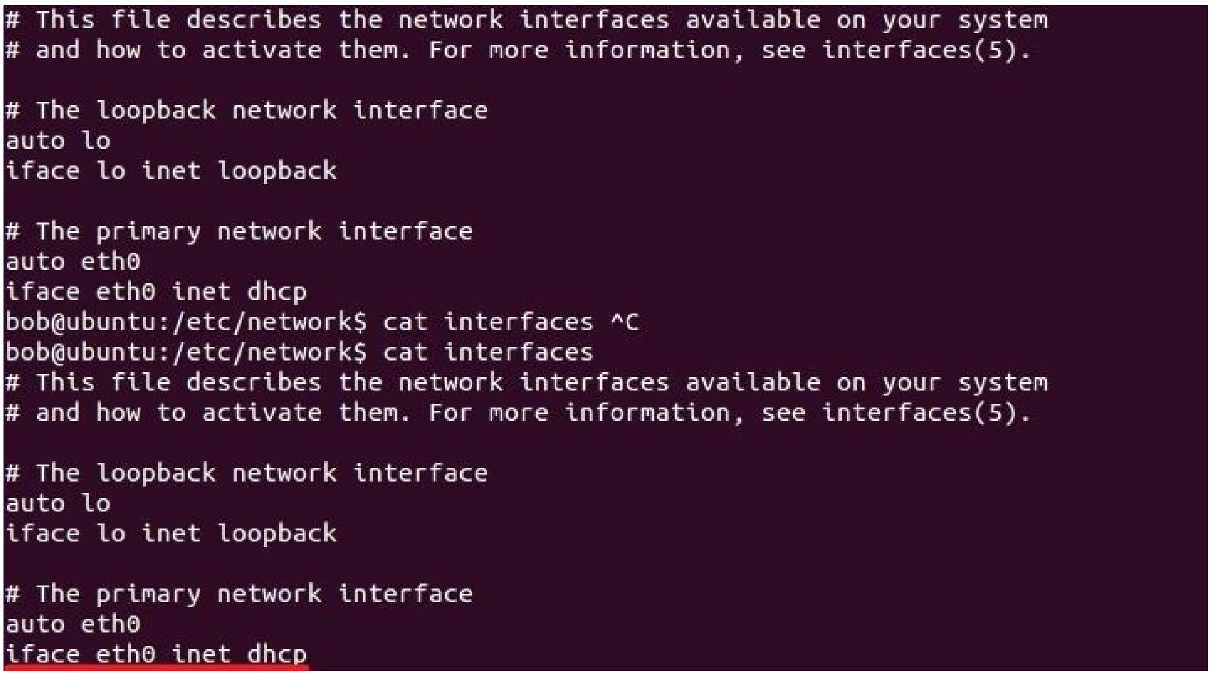
The client daemon, dhclient (part of the dhcp package), contacts the server daemon, dhcpd, to obtain the IP address, netmask, broadcast address, nameserver address, and other networking parameters. The server provides a lease on the IP address to the client. The client can request the specific terms of the lease, including its duration; the server can, in turn, limit these terms. While connected to the network, a client typically requests extensions of its lease as necessary so its IP address remains the same. The lease can expire once the client is disconnected from the network, with the server giving the client a new IP address when it requests a new lease. You can also set up a DHCP server to provide static IP addresses for specific clients. DHCP is broadcast based, so both client and server must be on the same subnet.

Configure DHCP client on Ubuntu

To configure your Ubuntu distribution to be a DHCP client, you need to modify the **/etc/network/interfaces** file. You will need to add the following line to the file:

iface INTERFACE inet dhcp

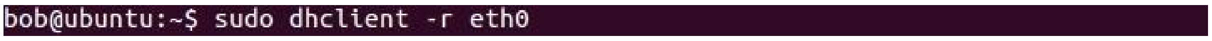
For example, to configure the **eth0** interface as a DHCP client, we would add the following configuration:



```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
iface eth0 inet dhcp
bob@ubuntu:/etc/network$ cat interfaces ^C
bob@ubuntu:/etc/network$ cat interfaces
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
iface eth0 inet dhcp
```

The system should now request network parameters from the DHCP server when booting.

To run the DHCP process manually, you can use the *dhclient* command. For example, to run the DHCP process on the **eth0** interface, use the following command:



```
bob@ubuntu:~$ sudo dhclient -r eth0
```

We have used the *-r* option to release the current interface configuration.

IP address Configuration (static and DHCP) on Windows 10:

Setting a Static IP Address in Windows 10

It's important to setup a static IP address if you are planning on forwarding ports. When you setup port forwarding your router forwards ports to the IP address you specify. If you don't setup a static IP address the port forward will probably work the first time. But after restarting your computer it may use a different IP address. When this happens, the ports are no longer forwarded to that computer's IP address.

What is an IP Address?

IP addresses are four sets of numbers that are separated by periods that allow computers to identify each other. Every computer on your network has at least one ip address. Two computers on the same network should never have the same IP Address. If two computers end up with the same IP address neither will be able to connect to the Internet.

Dynamic IP vs. Static IP

Your router most likely assigns dynamic IP addresses by default. Routers do this because having a dynamic IP address network requires no configuration on your part. You can simply plug in your computer and the network will work. When IP addresses are assigned dynamically, it is the router's job to assign them. Every time a computer reboots it asks the router for an IP address. The router then hands the computer an IP address that has not already been handed out to another computer. This is a very important because when you set your computer to a static IP address, the router doesn't know that a computer is already using that IP address. That same IP address could be handed out to another computer later on. This will prevent both computers from connecting to the Internet. It's important to assign an IP address that will not be handed out to a different computer by the dynamic IP address server. The dynamic IP address server is generally referred to as the DHCP server.

TCP/IP defines how your PC communicates with other PCs. To make it easier to manage TCP/IP settings, we recommend using automated Dynamic Host Configuration Protocol (DHCP). DHCP automatically assigns Internet Protocol (IP) addresses to the computers on your network if your network supports it. If you use DHCP, then you don't have to change your TCP/IP settings if you move your PC to another location, and DHCP doesn't require you to manually configure TCP/IP settings, such as Domain Name System (DNS) and Windows Internet Name Service (WINS).

To enable DHCP or change other TCP/IP settings:

1. Select **Start** , then select **Settings** > **Network & Internet** .
2. Do one of the following:
   - For a Wi-Fi network, select **Wi-Fi** > **Manage known networks**. Choose the network you want to change the settings for, then select **Properties.**
   - For an Ethernet network, select **Ethernet**, then select the Ethernet network you're connected to.
3. Under **IP assignment**, select **Edit**.
4. Under **Edit IP settings**, select **Automatic (DHCP)** or **Manual**.

To specify IPv4 settings manually

1. Under **Edit IP settings**, choose **Manual**, then turn on **IPv4**.
2. To specify an IP address, in the **IP address, Subnet prefix length**, and **Gateway** boxes, type the IP address settings.
3. To specify a DNS server address, in the **Preferred DNS** and **Alternate DNS** boxes, type the addresses of the primary and secondary DNS servers.

To specify IPv6 settings manually

1. Under **Edit IP settings**, choose **Manual**, then turn on **IPv6**.
2. To specify an IP address, in the **IP address, Subnet prefix length**, and **Gateway** boxes, type the IP address settings.
3. To specify a DNS server address, in the **Preferred DNS** and **Alternate DNS** boxes, type the addresses of the primary and secondary DNS servers.
    - When you select **Automatic (DHCP)**, the IP address settings and DNS server address setting are set automatically by your router or other access point (recommended).
    - When you select **Manual**, you can manually set your IP address settings and DNS server address.

5. When you're done, select **Save**.

Note: To install IPv4, run Command Prompt as an administrator, type **netsh interface ipv4 install**, and then press **Enter**.

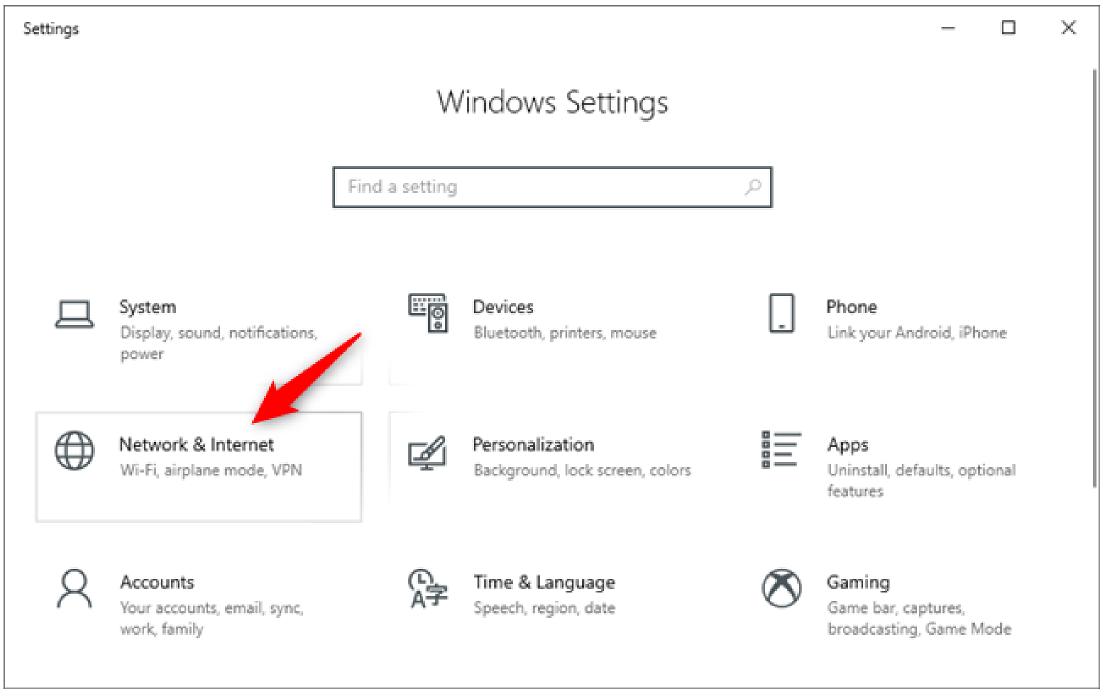
3 ways to change the IP address in Windows 10

NOTE: To be able to change your IP address in Windows 10, you must log in to Windows 10 using an administrator account.

1. How to change the IP address in Windows 10, using the Settings app

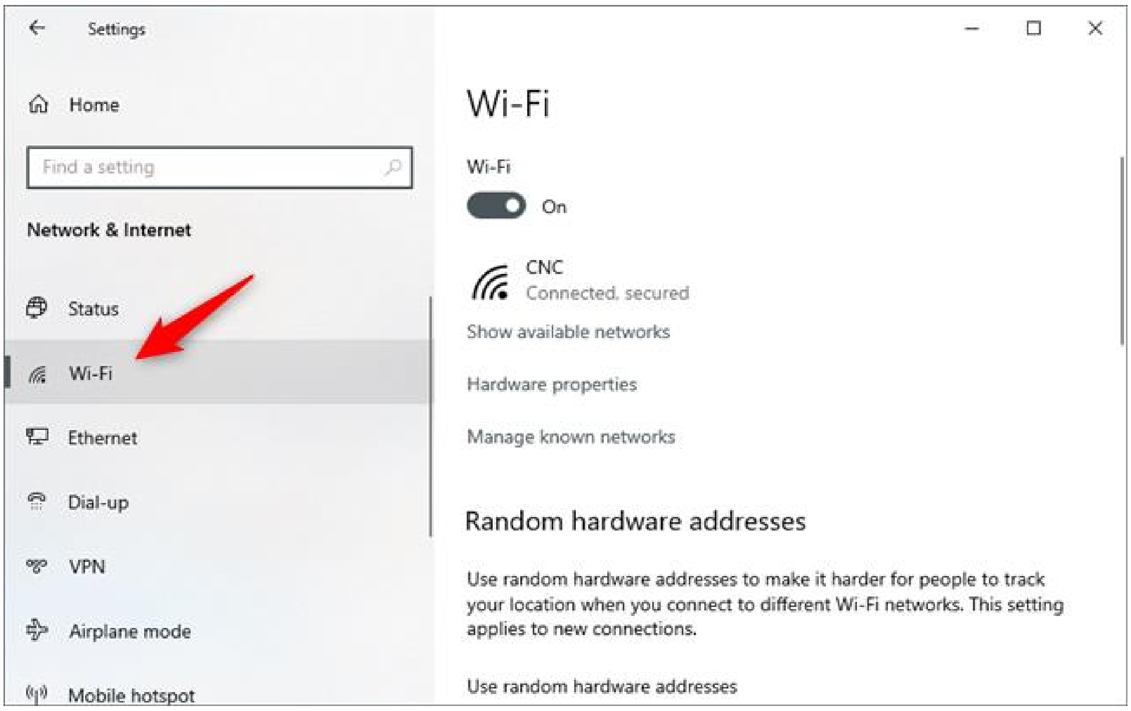
Because it relies on the visual interface of the operating system, this is probably the easiest method for changing the IP address of your Windows 10 computer or device. Here's what you need to do:

Open the Settings app: a quick way to do it is to push the *Settings* button from the *Start Menu* or to press *Windows + I* on your keyboard simultaneously. In the *Settings* app, open the *Network & Internet* category.


Settings
Windows Settings
Find a setting
System
Display, sound, notifications, power
Devices
Bluetooth, printers, mouse
Phone
Link your Android, iPhone
Network & Internet
Wi-Fi, airplane mode, VPN
Personalization
Background, lock screen, colors
Apps
Uninstall, defaults, optional features
Accounts
Your accounts, email, sync, work, family
Time & Language
Speech, region, date
Gaming
Game bar, captures, broadcasting, Game Mode

The Network & Internet category of settings

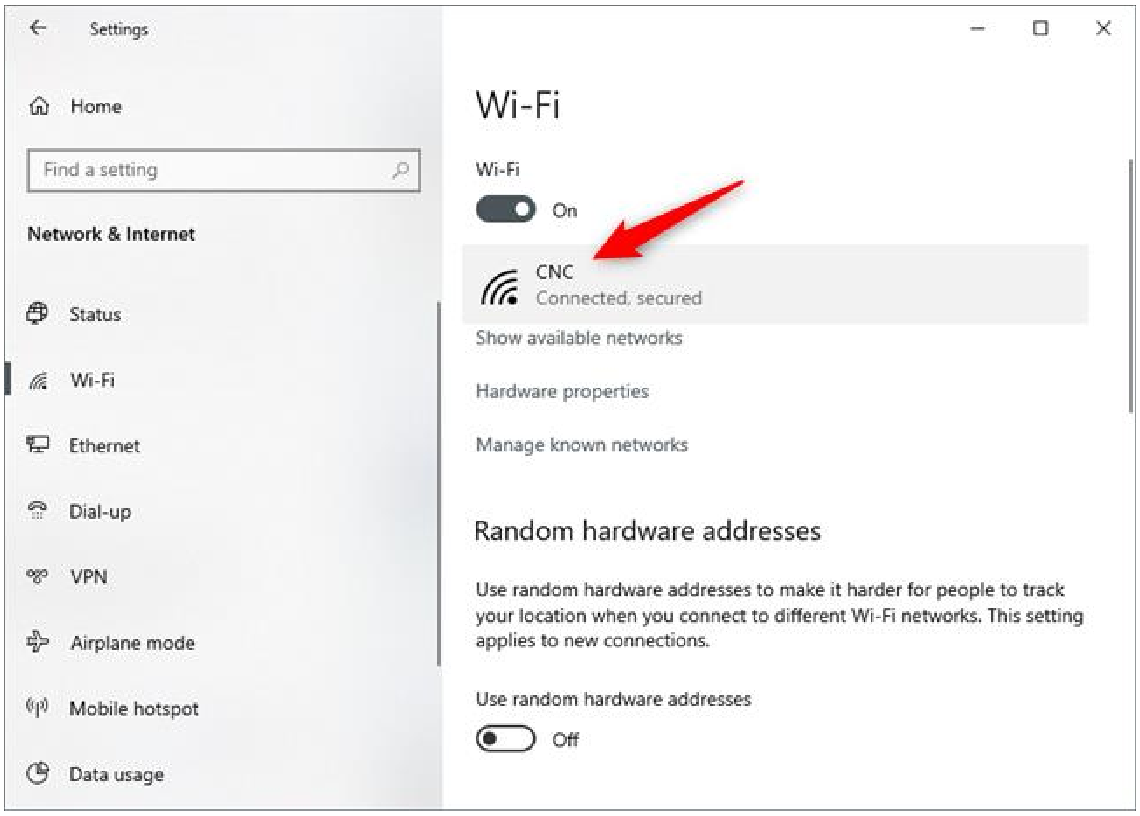
On the left sidebar, select your network type. If you connect to the internet (or local area network) using a wireless card, select *Wi-Fi*. If you're accessing the internet (or LAN) using a wired connection, click or tap on *Ethernet*.



Use random hardware addresses to make it harder for people to track your location when you connect to different Wi-Fi networks. This setting applies to new connections.

The Wi-Fi network connection

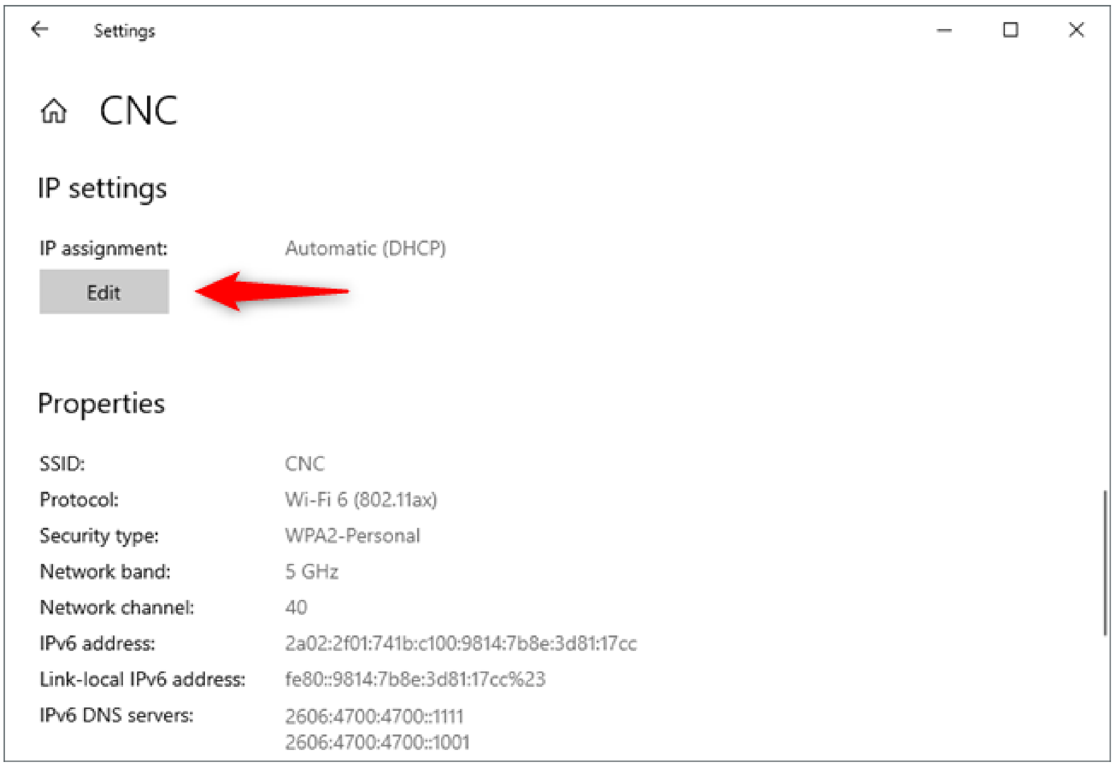
On the right side of the window, click or tap on the name of your network connection.



Use random hardware addresses to make it harder for people to track your location when you connect to different Wi-Fi networks. This setting applies to new connections.

The currently connected network

Scroll down on your network connection details page until you find the section called *IP settings*. Then, click or tap on *Edit*, under *IP assignment*.


Settings

CNC

IP settings

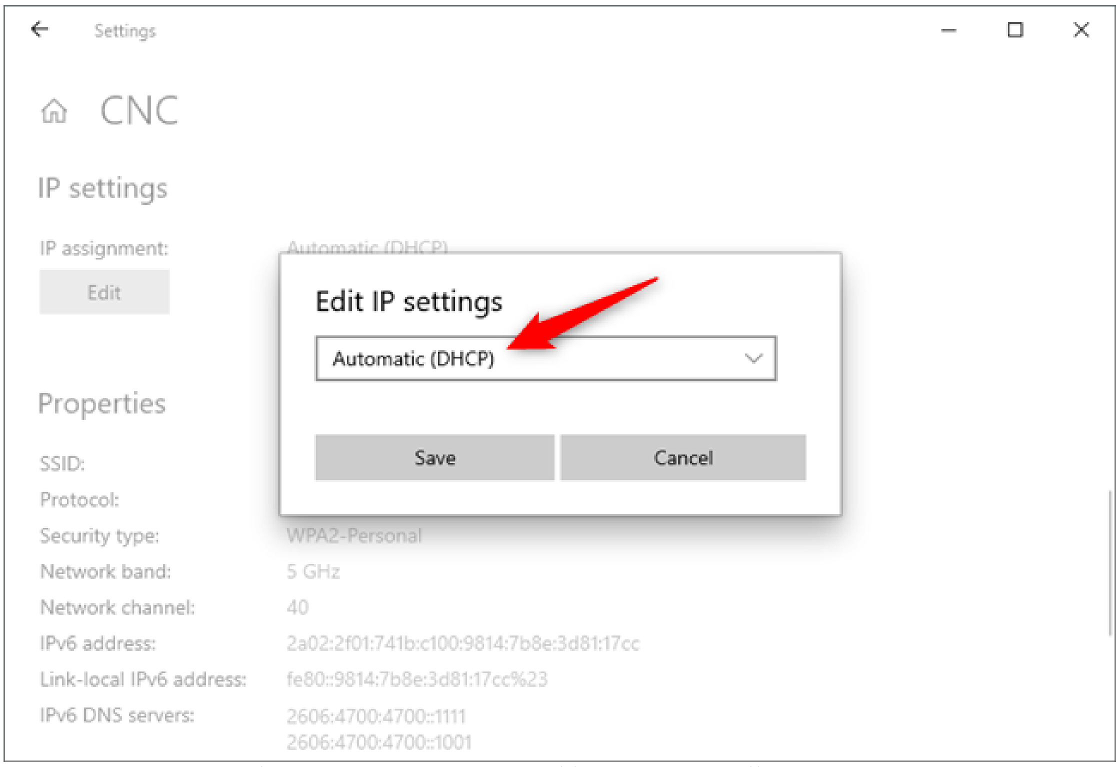
IP assignment:	Automatic (DHCP)

Properties

SSID:	CNC
Protocol:	Wi-Fi 6 (802.11ax)
Security type:	WPA2-Personal
Network band:	5 GHz
Network channel:	40
IPv6 address:	2a02:2f01:741b:c100:9814:7b8e:3d81:17cc
Link-local IPv6 address:	fe80::9814:7b8e:3d81:17cc%23
IPv6 DNS servers:	2606:4700:4700::1111 2606:4700:4700::1001

The Edit button from the IP assignment area

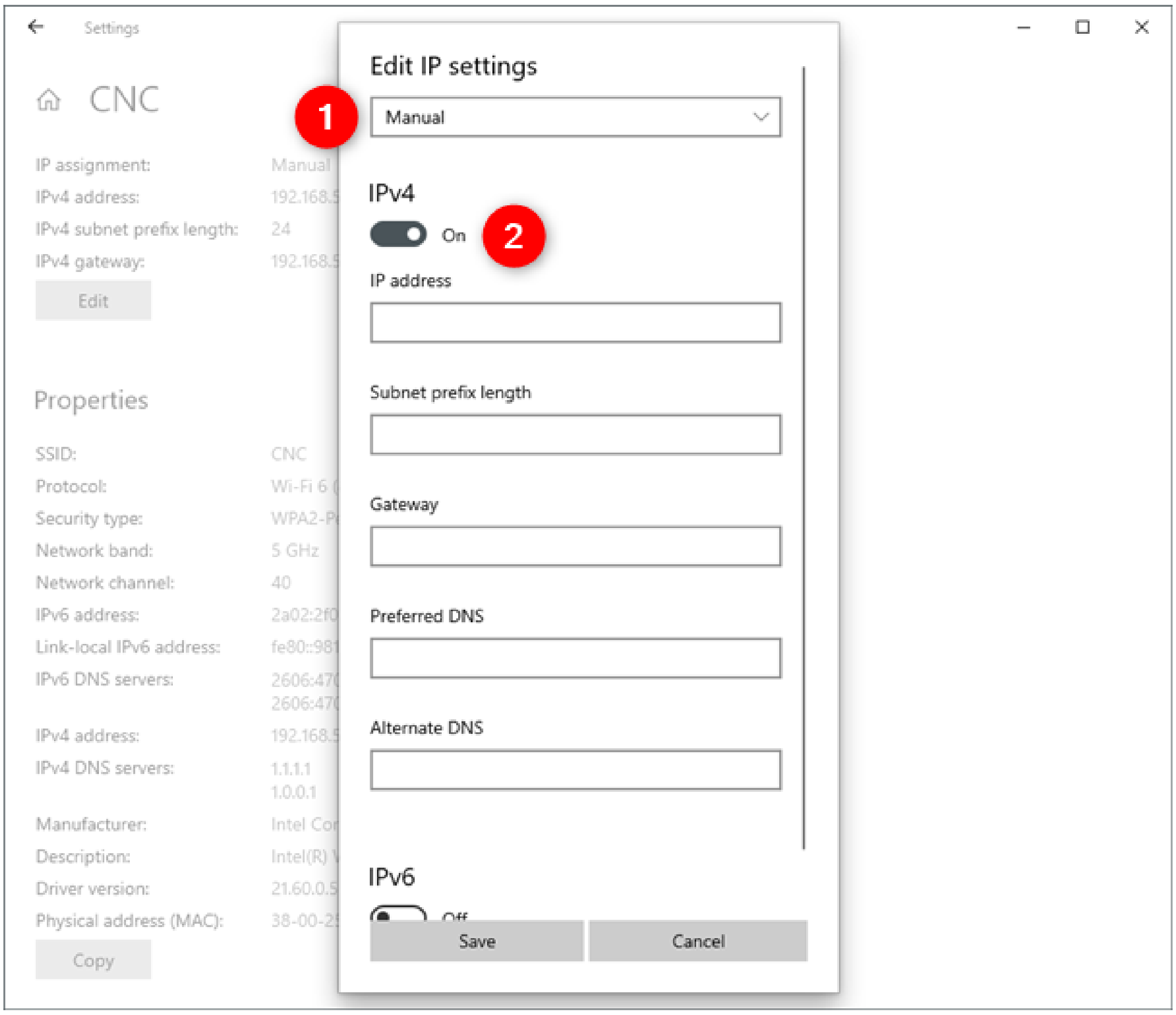
The *Settings* app now shows the *"Edit IP settings"* dialog. This is where you can change the IP address of your computer or device. If you want the IP address of your Windows 10 PC to be assigned automatically by your router, select *Automatic (DHCP)*. This is also called a dynamic IP address.


Edit IP settings

Automatic (DHCP)

Edit IP settings to get an IP address automatically via DHCP

If you want to set your own **static IP address**, select *Manual* and then enable the *IPv4* and/or *IPv6* switches, depending on what internet protocols you want to use. Note that each of them has its own distinct IP address, so you must enter the required details for both *IPv4* and *IPv6* if you choose to enable both.


Settings

CNC

IP assignment: Manual
IPv4 address: 192.168.5
IPv4 subnet prefix length: 24
IPv4 gateway: 192.168.5

Properties

SSID: CNC
Protocol: Wi-Fi 6 (
Security type: WPA2-P
Network band: 5 GHz
Network channel: 40
IPv6 address: 2a02:2f0
Link-local IPv6 address: fe80::981
IPv6 DNS servers: 2606:470 2606:470
IPv4 address: 192.168.5
IPv4 DNS servers: 1.1.1.1 1.0.0.1
Manufacturer: Intel Co
Description: Intel(R)
Driver version: 21.60.0.5
Physical address (MAC): 38-00-2

Edit IP settings

Manual

IPv4

On

IP address

Subnet prefix length

Gateway

Preferred DNS

Alternate DNS

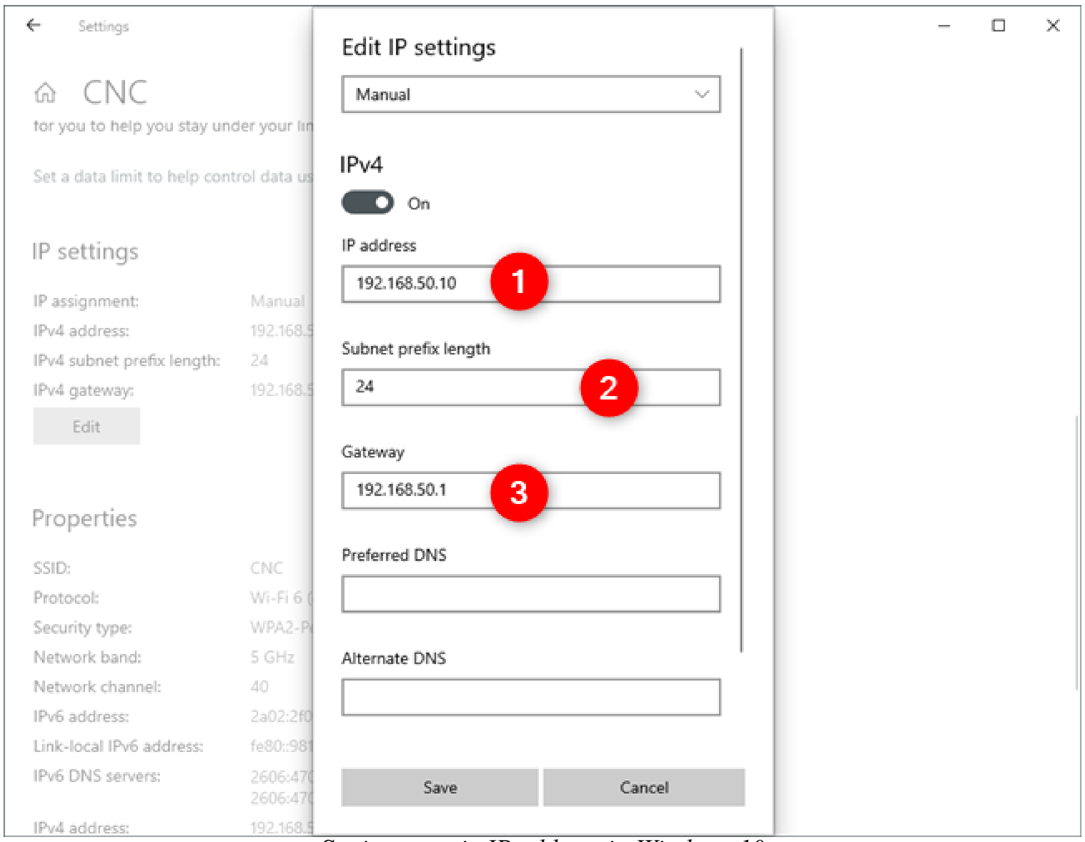
IPv6

Off

Edit IP settings to change the IP address for IPv4

To change your IP address to a static one, regardless of whether you set it for your *IPv4* or *IPv6* protocols, you have to enter the following details:

- **IP address:** type the static IP address that you want to use. For example, we want to change the IP address (IPv4) of our Windows 10 PC to 192.168.50.10.
- **Subnet prefix length:** type the prefix length that determines the size of the subnet. For example, we configured our router to use a subnet mask of 255.255.255.0, which means that we have to enter a *"Subnet prefix length"* of 24 (the number of 1 bits in the netmask). If we would've had a subnet mask of 255.255.0.0, the prefix length would have been 16, and so on. You can find more details about subnets here: What are IP addresses, subnet masks, and how do you change them in Windows?.
- **Gateway:** type the IP address of your router. In our case, that's 192.168.50.1.


Settings
CNC
for you to help you stay under your lim
Set a data limit to help control data us
IP settings
Edit
Properties
1
2
3

IP assignment: Manual
IPv4 address: 192.168.5
IPv4 subnet prefix length: 24
IPv4 gateway: 192.168.5

SSID: CNC
Protocol: Wi-Fi 6 (
Security type: WPA2-P
Network band: 5 GHz
Network channel: 40
IPv6 address: 2a02:2f0
Link-local IPv6 address: fe80::981
IPv6 DNS servers: 2606:470 2606:470
IPv4 address: 192.168.5

Edit IP settings

Manual

IPv4

On

IP address: 192.168.50.10

Subnet prefix length: 24

Gateway: 192.168.50.1

Preferred DNS:

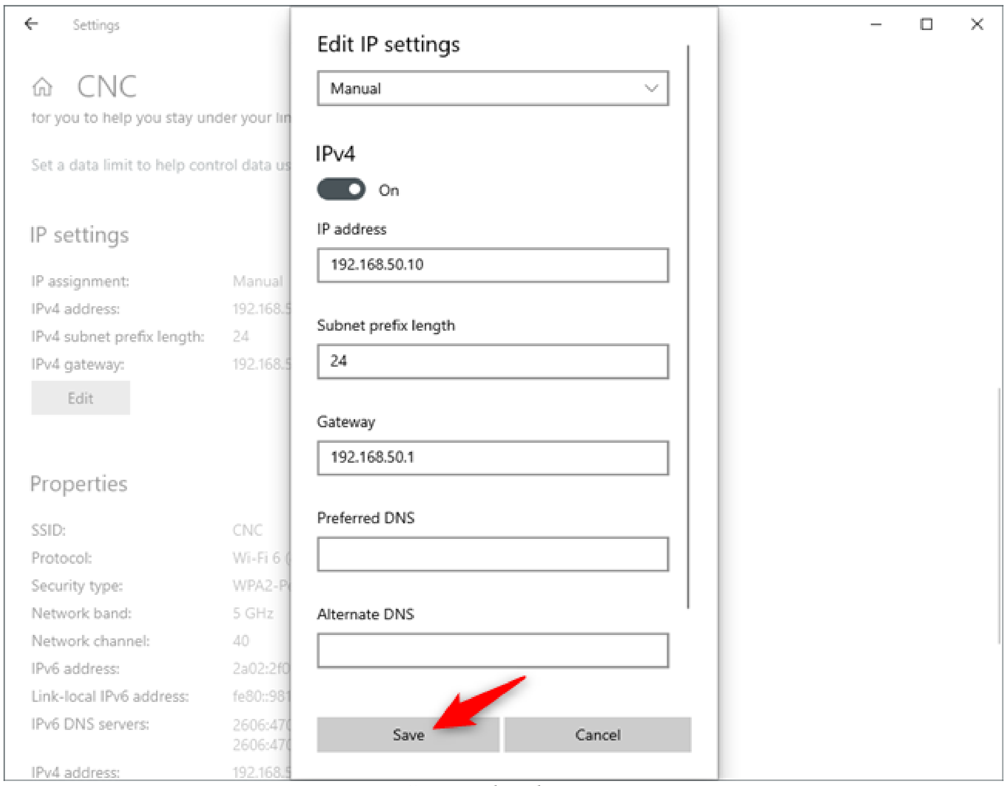
Alternate DNS:

Save | Cancel

Setting a static IP address in Windows 10

The *Preferred DNS* and *Alternate DNS* settings are not mandatory - if you leave them blank, they are automatically assigned by your router. However, if you want to change them too, you can do so.

Once you've entered all the details, click or tap on *Save* so that your IP address is changed by Windows 10.


Settings
CNC
for you to help you stay under your lim
Set a data limit to help control data us
IP settings
Edit
Properties

IP assignment: Manual
IPv4 address: 192.168.5
IPv4 subnet prefix length: 24
IPv4 gateway: 192.168.5

SSID: CNC
Protocol: Wi-Fi 6 (
Security type: WPA2-P
Network band: 5 GHz
Network channel: 40
IPv6 address: 2a02:2f0
Link-local IPv6 address: fe80::981
IPv6 DNS servers: 2606:470 2606:470
IPv4 address: 192.168.5

Edit IP settings

Manual

IPv4

On

IP address: 192.168.50.10

Subnet prefix length: 24

Gateway: 192.168.50.1

Preferred DNS:

Alternate DNS:

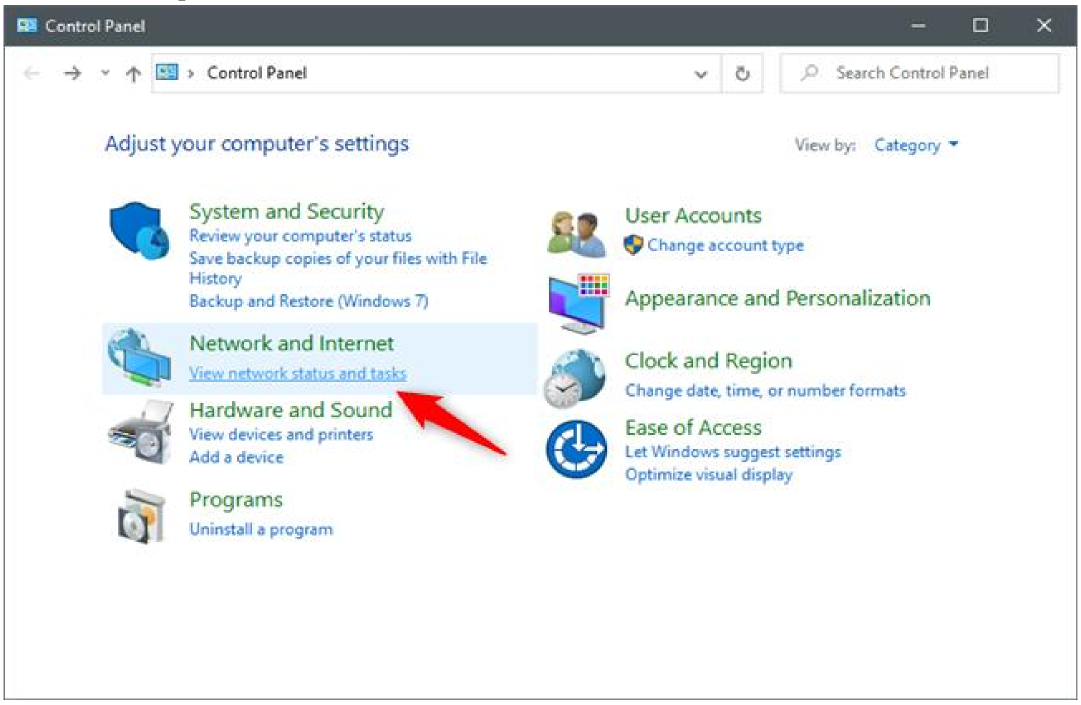
Save | Cancel

Saving the changes

NOTE: If you choose to use a static IP address, make sure that all the details you enter are right; otherwise, your Windows 10 PC loses internet connectivity. If that happens, change your IP address back to *Automatic (DHCP)* so that your router can change it to something that works.

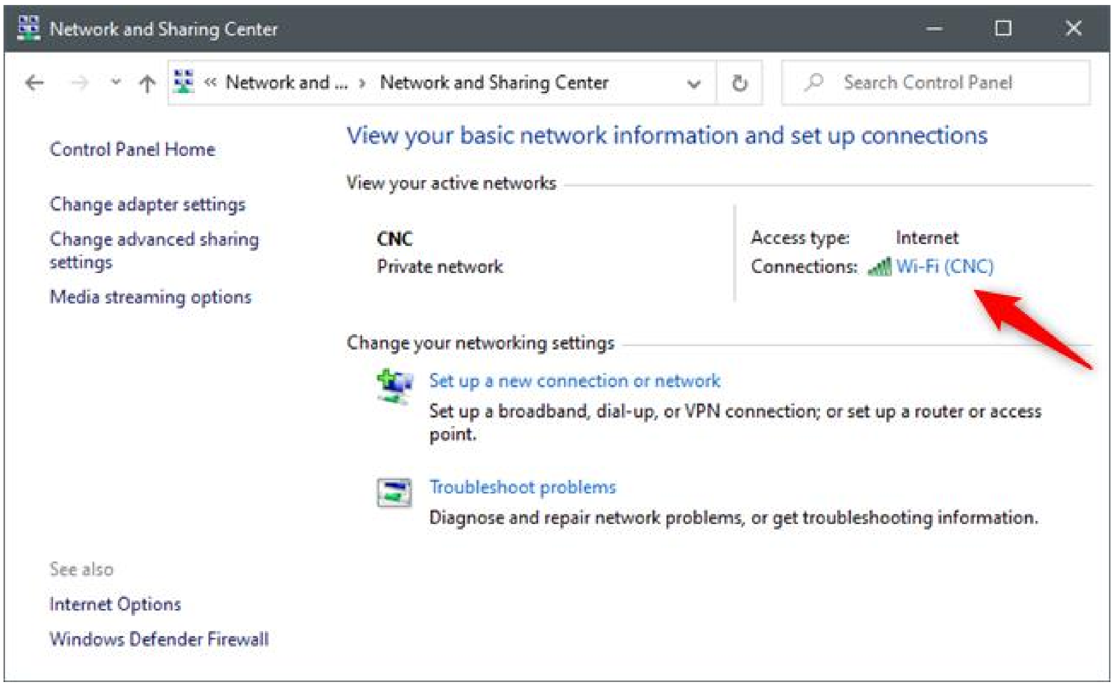
2. How to change the IP address in Windows 10, using the Control Panel

In Windows 10, you can also change your IP address from the *Control Panel*. Open the Control Panel and click or tap on *"View network status and tasks"* under *"Network and Internet."*


Control Panel
Control Panel
Search Control Panel
Adjust your computer's settings
View by: Category
System and Security
Review your computer's status
Save backup copies of your files with File History
Backup and Restore (Windows 7)
Network and Internet
View network status and tasks
Hardware and Sound
View devices and printers
Add a device
Programs
Uninstall a program
User Accounts
Change account type
Appearance and Personalization
Clock and Region
Change date, time, or number formats
Ease of Access
Let Windows suggest settings
Optimize visual display

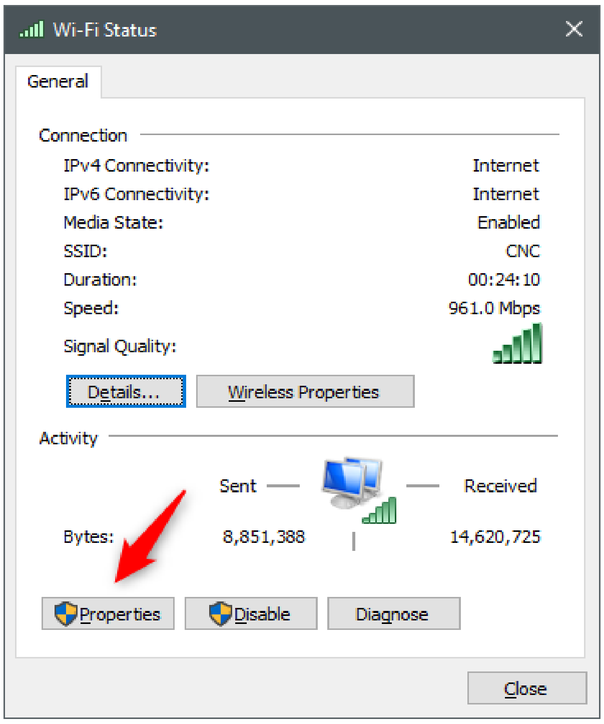
View network status and tasks in Control Panel

In the Network and Sharing Centre, click or tap on your internet connection from the *"View your active networks"* area.


Network and Sharing Center
« Network and ... › Network and Sharing Center
Search Control Panel
Control Panel Home
Change adapter settings
Change advanced sharing settings
Media streaming options
View your basic network information and set up connections
View your active networks
CNC
Private network
Access type: Internet
Connections: Wi-Fi (CNC)
Change your networking settings
Set up a new connection or network
Set up a broadband, dial-up, or VPN connection; or set up a router or access point.
Troubleshoot problems
Diagnose and repair network problems, or get troubleshooting information.
See also
Internet Options
Windows Defender Firewall

The currently used network connection

In your network's *Status* window, click or tap on *Properties*.



Wi-Fi Status

General

Connection

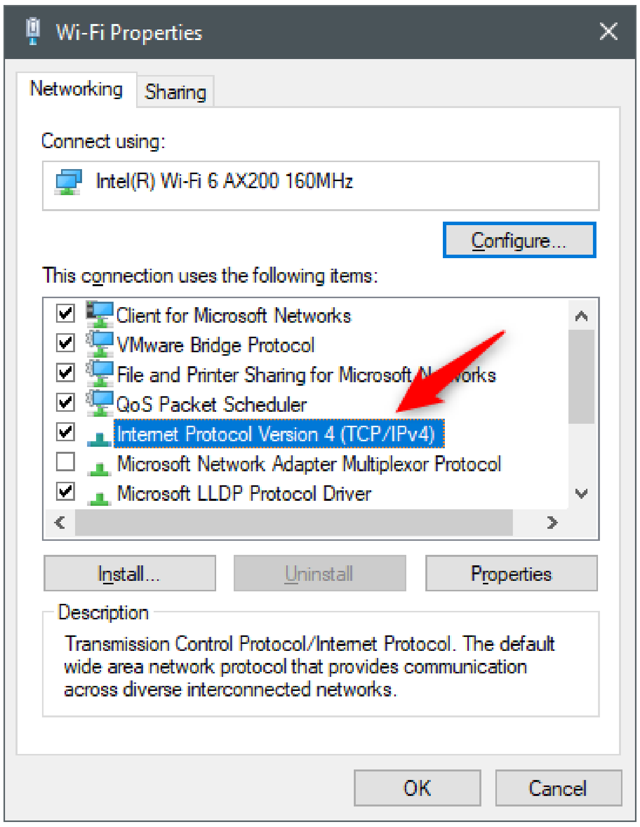
IPv4 Connectivity:	Internet
IPv6 Connectivity:	Internet
Media State:	Enabled
SSID:	CNC
Duration:	00:24:10
Speed:	961.0 Mbps
Signal Quality:	

Activity

	Sent	Received
Bytes:	8,851,388	14,620,725

The Properties button from the network's Status window

In the *Properties* window, select *Internet Protocol Version 4 (TCP/IPv4)* or *Internet Protocol Version 6 (TCP/IPv6)*, depending on the IP address that you want to change. If you want to change both, repeat the next steps for each of them.



Wi-Fi Properties

Networking Sharing

Connect using:

Intel(R) Wi-Fi 6 AX200 160MHz

This connection uses the following items:

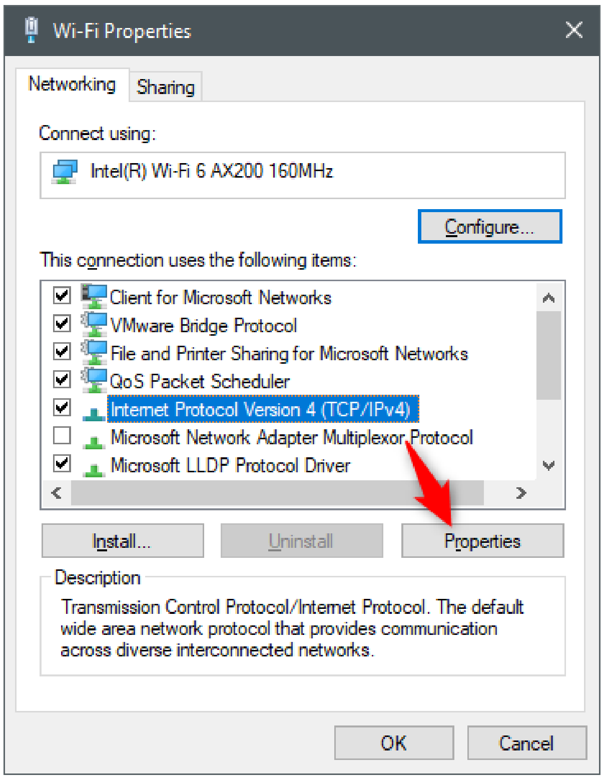
- [✓] Client for Microsoft Networks
- [✓] VMware Bridge Protocol
- [✓] File and Printer Sharing for Microsoft Networks
- [✓] QoS Packet Scheduler
- [✓] Internet Protocol Version 4 (TCP/IPv4)
- [ ] Microsoft Network Adapter Multiplexor Protocol
- [✓] Microsoft LLDP Protocol Driver

Description

Transmission Control Protocol/Internet Protocol. The default wide area network protocol that provides communication across diverse interconnected networks.

Selecting Internet Protocol Version 4 (TCP/IPv4)

Click or tap on *Properties*.



Wi-Fi Properties

Networking | Sharing

Connect using:

Intel(R) Wi-Fi 6 AX200 160MHz

Configure...

This connection uses the following items:

- [✓] Client for Microsoft Networks
- [✓] VMware Bridge Protocol
- [✓] File and Printer Sharing for Microsoft Networks
- [✓] QoS Packet Scheduler
- [✓] Internet Protocol Version 4 (TCP/IPv4)
- [ ] Microsoft Network Adapter Multiplexor Protocol
- [✓] Microsoft LLDP Protocol Driver

Install... | Uninstall | Properties

Description

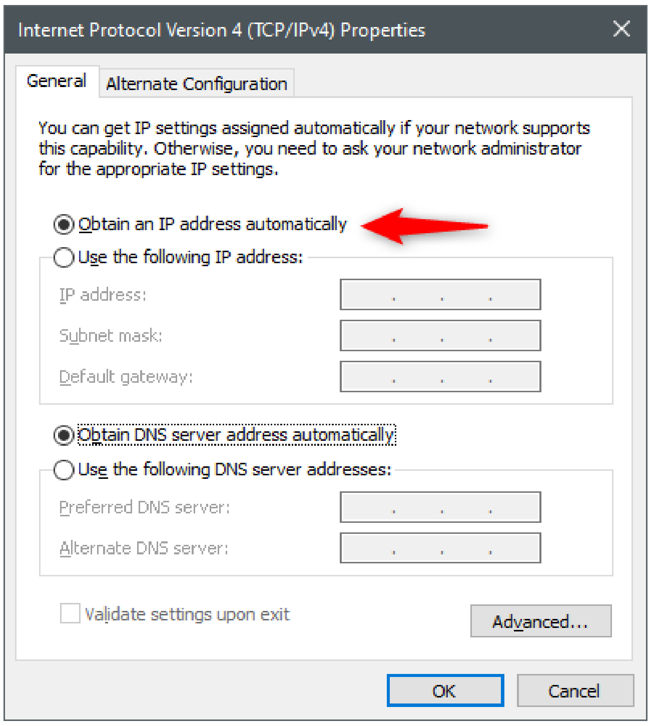
Transmission Control Protocol/Internet Protocol. The default wide area network protocol that provides communication across diverse interconnected networks.

OK | Cancel

Opening the Properties of Internet Protocol Version 4 (TCP/IPv4)

The previous action opens a window called *Internet Protocol Version 4 (TCP/IPv4) Properties* or *Internet Protocol Version 6 (TCP/IPv6) Properties*, depending on what you chose earlier. This is where you can change your IP address:

If you want to use a dynamic IP address that's automatically assigned to your Windows 10 PC by your router, select *"Obtain an IP address automatically."* Then, click or tap on *OK* and close all the windows you've opened.



Internet Protocol Version 4 (TCP/IPv4) Properties

General | Alternate Configuration

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

● Obtain an IP address automatically

○ Use the following IP address:

IP address: . . .

Subnet mask: . . .

Default gateway: . . .

● Obtain DNS server address automatically

○ Use the following DNS server addresses:

Preferred DNS server: . . .

Alternate DNS server: . . .

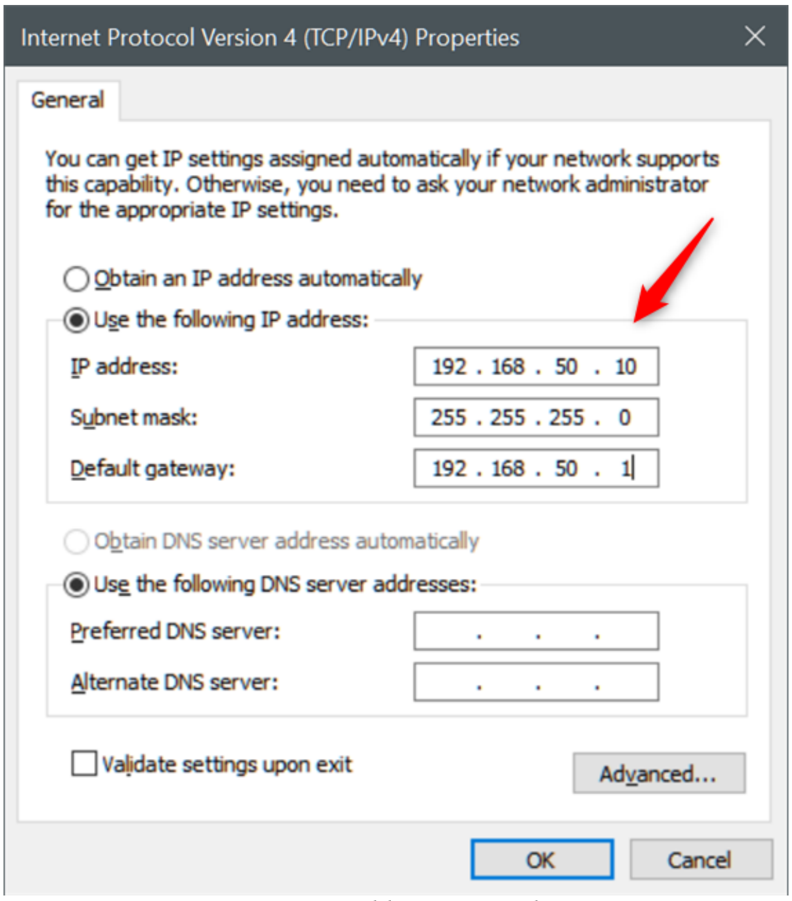
[ ] Validate settings upon exit

Advanced...

OK | Cancel

Obtain an IP address automatically

If you want to set a static IP address for your network adapter, select *"Use the following IP address."* Then, enter the required details manually: *IP address, Subnet mask*, and *Default gateway*, just like we've shown you in the previous method from this guide.



Internet Protocol Version 4 (TCP/IPv4) Properties

General

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

Obtain an IP address automatically [✓ Use the following IP address:]

IP address: 192 . 168 . 50 . 10
Subnet mask: 255 . 255 . 255 . 0
Default gateway: 192 . 168 . 50 . 1

Obtain DNS server address automatically [✓ Use the following DNS server addresses:]

Preferred DNS server: . . .
Alternate DNS server: . . .

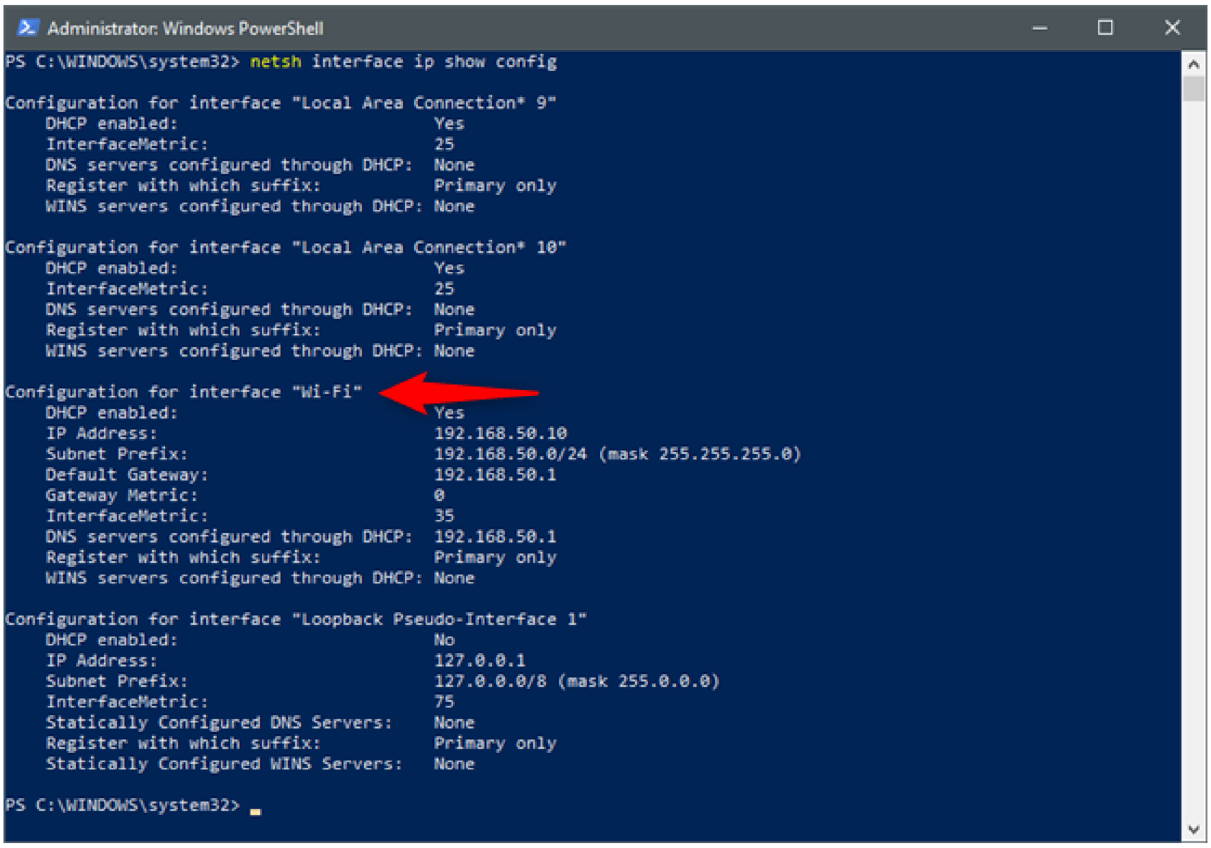
Validate settings upon exit

Set a static IP address in Windows 10

Save your new IP address settings by clicking or tapping on *OK*, then close all the windows you've opened during the process.

3. How to change the IP address in Windows 10, using PowerShell or Command Prompt (cmd)

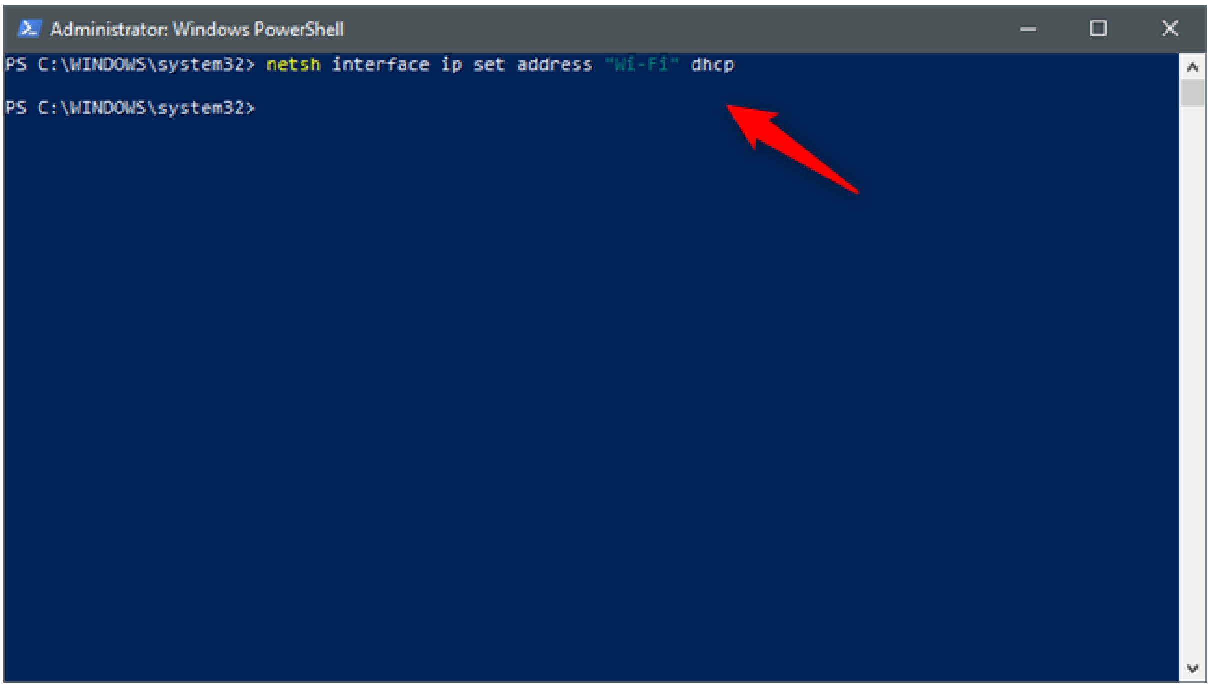
If you prefer using a command-line environment, you can also change the IP address in Windows 10 using PowerShell or Command Prompt. Regardless of what your favorite console is, run the **netsh interface ip show config** command. In the results that are displayed, identify the network adapter for which you want to change the IP address. For example, the name of our network interface is *Wi-Fi*, as you can see in the screenshot below.



```
PS C:\WINDOWS\system32> netsh interface ip show config

Configuration for interface "Local Area Connection* 9"
    DHCP enabled:                         Yes
    InterfaceMetric:                      25
    DNS servers configured through DHCP:  None
    Register with which suffix:           Primary only
    WINS servers configured through DHCP: None

Configuration for interface "Local Area Connection* 10"
    DHCP enabled:                         Yes
    InterfaceMetric:                      25
    DNS servers configured through DHCP:  None
    Register with which suffix:           Primary only
    WINS servers configured through DHCP: None

Configuration for interface "Wi-Fi"
    DHCP enabled:                         Yes
    IP Address:                           192.168.50.10
    Subnet Prefix:                        192.168.50.0/24 (mask 255.255.255.0)
    Default Gateway:                      192.168.50.1
    Gateway Metric:                       0
    InterfaceMetric:                      35
    DNS servers configured through DHCP:  192.168.50.1
    Register with which suffix:           Primary only
    WINS servers configured through DHCP: None

Configuration for interface "Loopback Pseudo-Interface 1"
    DHCP enabled:                         No
    IP Address:                           127.0.0.1
    Subnet Prefix:                        127.0.0.0/8 (mask 255.0.0.0)
    InterfaceMetric:                      75
    Statically Configured DNS Servers:    None
    Register with which suffix:           Primary only
    Statically Configured WINS Servers:   None

PS C:\WINDOWS\system32> _
```

netsh interface ip show config

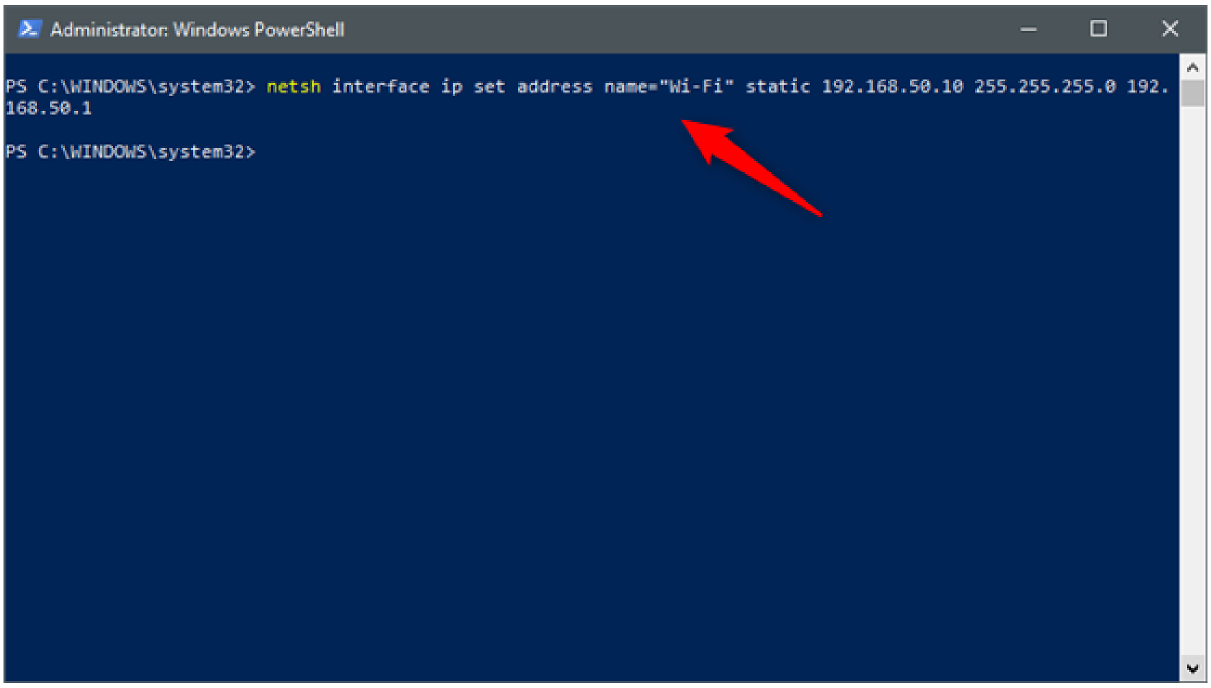
If you want to change the IP address to a dynamic one that's automatically assigned by your router via DHCP, run the following command: **netsh interface ip set address "Network Interface Name" dhcp**. Replace *"Network Interface Name"* with the name of your network connection. For example, we need to run *netsh interface ip set address "Wi-Fi" dhcp*.



```
PS C:\WINDOWS\system32> netsh interface ip set address "Wi-Fi" dhcp

PS C:\WINDOWS\system32>
```

netsh interface ip set address "Network Interface Name" dhcp

If you want to set a static IP address, run this command: **netsh interface ip set address name= "Network Interface Name" static [IP address] [Subnet Mask] [Gateway]**. Replace *[IP address] [Subnet Mask] [Gateway]* with the ones that match your network configuration. In our example, if we'd want to change our IP address to 192.168.50.10, we'd have to run *netsh interface ip set address name="Wi-Fi" static 192.168.50.10 255.255.255.0 192.168.50.1*.



```
PS C:\WINDOWS\system32> netsh interface ip set address name="Wi-Fi" static 192.168.50.10 255.255.255.0 192.
168.50.1

PS C:\WINDOWS\system32>
```

netsh interface ip set address name="Network Interface Name" static [IP address] [Subnet Mask] [Gateway]

Network Tools & commands
Part - III

Topic:

Introduction to important network related tools and commands, e.g. ifconfig, ip, hostname, ping, netstat, route, tcpdump, Wireshark, etc.

Important network related tools and commands:

- ***ifconfig***

The use of **/sbin/ifconfig** without any arguments will present you with a list of all active network interface cards, including wireless and the loopback interface. In the screenshot below **eth0** has no ip address.

```
root@ubu1010:~# ifconfig
eth0 Link encap:Ethernet HWaddr 00:26:bb:5d:2e:52
UP BROADCAST MULTICAST MTU:1500 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)
Interrupt:43 Base address:0xe000
eth1 Link encap:Ethernet HWaddr 00:26:bb:12:7a:5e
inet addr:192.168.1.30 Bcast:192.168.1.255 Mask:255.255.255.0
inet6 addr: fe80::226:bbff:fe12:7a5e/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:11141791 errors:202 dropped:0 overruns:0 frame:11580126
TX packets:6473056 errors:3860 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:3476531617 (3.4 GB) TX bytes:2114919475 (2.1 GB)
Interrupt:23
lo Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING MTU:16436 Metric:1
RX packets:2879 errors:0 dropped:0 overruns:0 frame:0
TX packets:2879 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:486510 (486.5 KB) TX bytes:486510 (486.5 KB)
```

We can also use **ifconfig** to obtain information about just one network card.

```
[root@rhel6 ~]# ifconfig eth0
eth0 Link encap:Ethernet HWaddr 08:00:27:DD:0D:5C
inet addr:192.168.1.99 Bcast:192.168.1.255 Mask:255.255.255.0
inet6 addr: fe80::a00:27ff:fedd:d5c/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:2969 errors:0 dropped:0 overruns:0 frame:0
TX packets:1918 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:335942 (328.0 KiB) TX bytes:190157 (185.7 KiB)
```

When **/sbin** is not in the **$PATH** of a normal user you will have to type the full path.

```
paul@debian5:~$ /sbin/ifconfig eth3
eth3 Link encap:Ethernet HWaddr 08:00:27:ab:67:30
inet addr:192.168.1.29 Bcast:192.168.1.255 Mask:255.255.255.0
inet6 addr: fe80::a00:27ff:feab:6730/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:27155 errors:0 dropped:0 overruns:0 frame:0
```

```
TX packets:30527 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:13095386 (12.4 MiB) TX bytes:25767221 (24.5 MiB)
```

1. *up and down*

You can also use **ifconfig** to bring an interface up or down. The difference with **ifup** is that **ifconfig eth0 up** will re-activate the nic keeping its existing (current) configuration, whereas **ifup** will read the correct file that contains a (possibly new) configuration and use this config file to bring the interface up.

```
[root@rhel6 ~]# ifconfig eth0 down
[root@rhel6 ~]# ifconfig eth0 up
[root@rhel6 ~]# ifconfig eth0
eth0 Link encap:Ethernet HWaddr 08:00:27:DD:0D:5C
inet addr:192.168.1.99 Bcast:192.168.1.255 Mask:255.255.255.0
inet6 addr: fe80::a00:27ff:fedd:d5c/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:2995 errors:0 dropped:0 overruns:0 frame:0
TX packets:1927 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:339030 (331.0 KiB) TX bytes:191583 (187.0 KiB)
```

2. *setting ip address*

You can **temporary** set an ip address with **ifconfig**. This ip address is only valid until the next **fup/ bifdown** cycle or until the next **reboot**.

```
[root@rhel6 ~]# ifconfig eth0 | grep 192
inet addr:192.168.1.99 Bcast:192.168.1.255 Mask:255.255.255.0
[root@rhel6 ~]# ifconfig eth0 192.168.33.42 netmask 255.255.0.0
[root@rhel6 ~]# ifconfig eth0 | grep 192
inet addr:192.168.33.42 Bcast:192.168.255.255 Mask:255.255.0.0
[root@rhel6 ~]# ifdown eth0 && ifup eth0
[root@rhel6 ~]# ifconfig eth0 | grep 192
inet addr:192.168.1.99 Bcast:192.168.1.255 Mask:255.255.255.0
```

3. *setting mac address*

You can also use **ifconfig** to set another **mac address** than the one hard coded in the network card. This screenshot shows you how.

```
[root@rhel6 ~]# ifconfig eth0 | grep HWaddr
eth0 Link encap:Ethernet HWaddr 08:00:27:DD:0D:5C
[root@rhel6 ~]# ifconfig eth0 hw ether 00:42:42:42:42:42
[root@rhel6 ~]# ifconfig eth0 | grep HWaddr
eth0 Link encap:Ethernet HWaddr 00:42:42:42:42:42
```

• *ip*

The **ifconfig** tool is deprecated on some systems. Use the **ip** tool instead. To see ip addresses on RHEL7 for example, use this command:

```
[root@rhel71 ~]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
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: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
qlen 1000
link/ether 08:00:27:89:22:33 brd ff:ff:ff:ff:ff:ff
inet 192.168.1.135/24 brd 192.168.1.255 scope global dynamic enp0s3
valid_lft 6173sec preferred_lft 6173sec
inet6 fe80::a00:27ff:fe89:2233/64 scope link
valid_lft forever preferred_lft forever
[root@rhel71 ~]#
```

• dhclient

Home and client Linux desktops often have **/sbin/dhclient** running. This is a daemon that enables a network interface to lease an ip configuration from a **dhcp server**. When your adapter is configured for **dhcp** or **bootp**, then **/sbin/ifup** will start the **dhclient** daemon. When a lease is renewed, **dhclient** will override your **ifconfig** set ip address!

• hostname

Every host receives a **hostname**, often placed in a **DNS name space** forming the **fqdn** or Fully Qualified Domain Name. This screenshot shows the **hostname** command and the configuration of the hostname on Red Hat/Fedora.

```
[root@rhel6 ~]# grep HOSTNAME /etc/sysconfig/network
HOSTNAME=rhel6
[root@rhel6 ~]# hostname
rhel6
```

Starting with RHEL7/CentOS7 this file is empty. The hostname is configured in the standard **/etc/hostname** file.

```
[root@rhel71 ~]# cat /etc/hostname
rhel71.linux-training.be
[root@rhel71 ~]#
```

Ubuntu/Debian uses the **/etc/hostname** file to configure the **hostname**.

```
paul@debian8:~$ cat /etc/hostname
server42
paul@debian8:~$ hostname
server42
```

On all Linux distributions you can change the **hostname** using the **hostname $newname** command. This is not a permanent change.

```
[root@rhel6 ~]# hostname server42
[root@rhel6 ~]# hostname
server42
```

On any Linux you can use **sysctl** to display and set the hostname.

```
[root@rhel6 ~]# sysctl kernel.hostname
kernel.hostname = server42
[root@rhel6 ~]# sysctl kernel.hostname=rhel6
kernel.hostname = rhel6
[root@rhel6 ~]# sysctl kernel.hostname
kernel.hostname = rhel6
[root@rhel6 ~]# hostname
rhel6
```

• route

We can see the computer's local routing table with the **/sbin/route** command (also with **netstat -r**).

```
root@RHEL4b ~]# 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
[root@RHEL4b ~]# 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
[root@RHEL4b ~]#
```

It appears this computer does not have a **gateway** configured, so we use **route add default gw** to add a **default gateway** on the fly.

```
[root@RHEL4b ~]# route add default gw 192.168.1.1
[root@RHEL4b ~]# 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
[root@RHEL4b ~]#
```

- ***ping***

If we can **ping** to another host, then **tcp/ip** is configured.

```
[root@RHEL4b ~]# ping 192.168.1.5
PING 192.168.1.5 (192.168.1.5) 56(84) bytes of data.
64 bytes from 192.168.1.5: icmp_seq=0 ttl=64 time=1004 ms
64 bytes from 192.168.1.5: icmp_seq=1 ttl=64 time=1.19 ms
64 bytes from 192.168.1.5: icmp_seq=2 ttl=64 time=0.494 ms
64 bytes from 192.168.1.5: icmp_seq=3 ttl=64 time=0.419 ms
--- 192.168.1.5 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3009ms
rtt min/avg/max/mdev = 0.419/251.574/1004.186/434.520 ms, pipe 2
[root@RHEL4b ~]#
```

- ***optional: ethtool***

To display or change network card settings, use **ethtool**. The results depend on the capabilities of your network card. The example shows a network that auto-negotiates it's bandwidth.

```
root@laika:~# ethtool eth0
Settings for eth0:
Supported ports: [ TP ]
Supported link modes: 10baseT/Half 10baseT/Full
100baseT/Half 100baseT/Full
1000baseT/Full
Supports auto-negotiation: Yes
Advertised link modes: 10baseT/Half 10baseT/Full
100baseT/Half 100baseT/Full
1000baseT/Full
Advertised auto-negotiation: Yes
Speed: 1000Mb/s
Duplex: Full
Port: Twisted Pair
PHYAD: 0
Transceiver: internal
Auto-negotiation: on
Supports Wake-on: pumbg
Wake-on: g
Current message level: 0x00000033 (51)
Link detected: yes
```

This example shows how to use ethtool to switch the bandwidth from 1000Mbit to 100Mbit and back. Note that some time passes before the nic is back to 1000Mbit.

```
root@laika:~# ethtool eth0 | grep Speed
Speed: 1000Mb/s
root@laika:~# ethtool -s eth0 speed 100
root@laika:~# ethtool eth0 | grep Speed
Speed: 100Mb/s
root@laika:~# ethtool -s eth0 speed 1000
root@laika:~# ethtool eth0 | grep Speed
Speed: 1000Mb/s
```

Network Sniffing - A network administrator should be able to use a sniffer like **wireshark** or **tcpdump** to troubleshoot network problems.

- ***tcpdump***

Sniffing on the command line can be done with **tcpdump**. Here are some examples. Using the **tcpdump host $ip** command displays all traffic with one host (192.168.1.38 in this example).

```
root@ubuntu910:~# tcpdump host 192.168.1.38
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
```

Capturing only ssh (tcp port 22) traffic can be done with **tcpdump tcp port $port**. This screenshot is cropped to 76 characters for readability in the pdf.

```
root@deb503:~# tcpdump tcp port 22
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 96 bytes
14:22:20.716313 IP deb503.local.37973 > rhel53.local.ssh: P 666050963:66605
14:22:20.719936 IP rhel53.local.ssh > deb503.local.37973: P 1:49(48) ack 48
14:22:20.720922 IP rhel53.local.ssh > deb503.local.37973: P 49:113(64) ack
14:22:20.721321 IP rhel53.local.ssh > deb503.local.37973: P 113:161(48) ack
14:22:20.721820 IP deb503.local.37973 > rhel53.local.ssh: . ack 161 win 200
14:22:20.722492 IP rhel53.local.ssh > deb503.local.37973: P 161:225(64) ack
14:22:20.760602 IP deb503.local.37973 > rhel53.local.ssh: . ack 225 win 200
14:22:23.108106 IP deb503.local.54424 > ubuntu910.local.ssh: P 467252637:46
14:22:23.116804 IP ubuntu910.local.ssh > deb503.local.54424: P 1:81(80) ack
14:22:23.116844 IP deb503.local.54424 > ubuntu910.local.ssh: . ack 81 win 2
^C
10 packets captured
10 packets received by filter
0 packets dropped by kernel
```

Same as above, but write the output to a file with the **tcpdump -w $filename** command.

```
root@ubuntu910:~# tcpdump -w sshdump.tcpdump tcp port 22
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
^C
17 packets captured
17 packets received by filter
0 packets dropped by kernel
```

With **tcpdump -r $filename** the file created above can be displayed.

```
root@ubuntu910:~# tcpdump -r sshdump.tcpdump
```

Many more examples can be found in the manual page of **tcpdump**.

- ***Wireshark***

1. *Installing Wireshark*

This example shows how to install **wireshark** on **.deb** based distributions (including Debian, Mint, Xubuntu, and others).

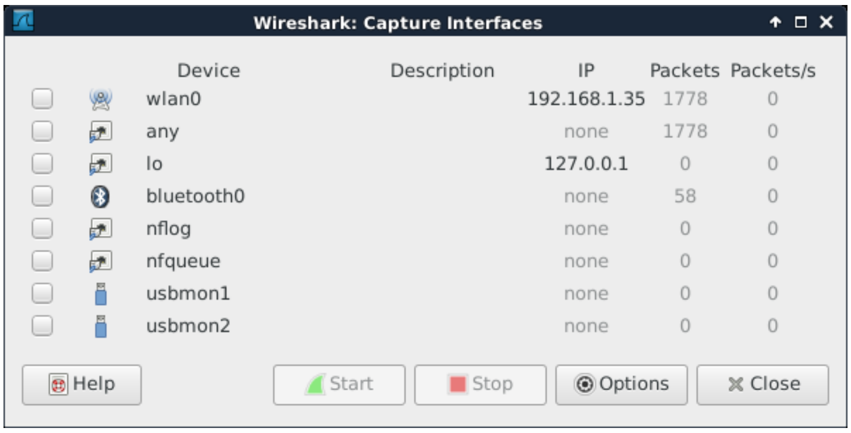
```
root@debian8:~# apt-get install wireshark
Reading package lists... Done
Building dependency tree
Reading state information... Done
... (output truncated)
```

On **.rpm** based distributions like CentOS, RHEL and Fedora you can use **yum** to install **wireshark**.

```
[root@centos7 ~]# yum install wireshark
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
... (output truncated)
```

2. *Selecting interface*

When we start **wireshark** for the first time, we will need to select an interface. We will see a dialog box that looks similar to this one.



Wireshark: Capture Interfaces

Device	Description	IP	Packets	Packets/s
wlan0		192.168.1.35	1778	0
any		none	1778	0
lo		127.0.0.1	0	0
bluetooth0		none	58	0
nflog		none	0	0
nfqueue		none	0	0
usbmon1		none	0	0
usbmon2		none	0	0

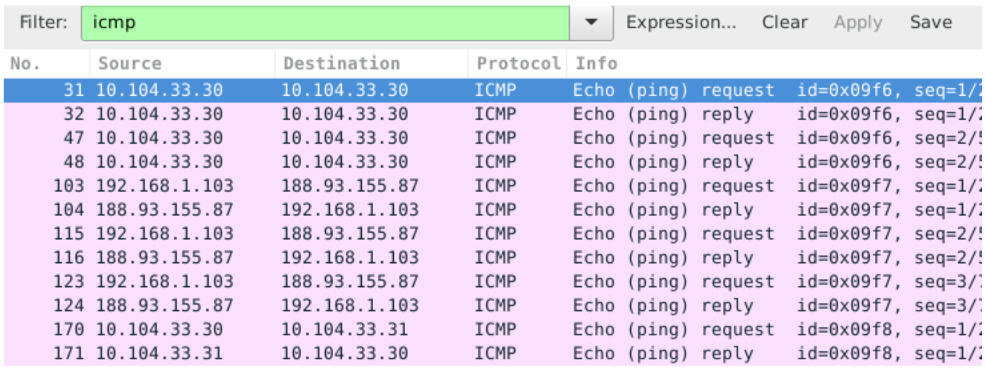
It is possible that there are no interfaces available because some distributions only allow root to sniff the network. We may need to use **sudo wireshark**. Or We can follow the general advice to sniff using **tcpdump** or any other tool, and save the capture to a file. Any saved capture can be analyzed using **wireshark** at a later time.

3. *Minimize traffic*

Sniffing a network can generate many thousands of packets in a very short time. This can be overwhelming. Try to mitigate by isolating our sniffer on the network. Preferably sniff an isolated virtual network interface over which you control all traffic.

4. *Sniffing ping*

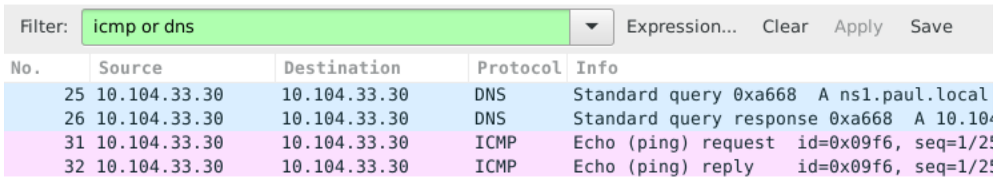
In total more than 200 packets were sniffed from the network. Things become clearer when we enter **icmp** in the filter field and press the **apply** button.



```
Filter: icmp                                 Expression...  Clear  Apply  Save

No.   Source          Destination     Protocol Info
   31 10.104.33.30    10.104.33.30    ICMP     Echo (ping) request  id=0x09f6, seq=1/
   32 10.104.33.30    10.104.33.30    ICMP     Echo (ping) reply    id=0x09f6, seq=1/
   47 10.104.33.30    10.104.33.30    ICMP     Echo (ping) request  id=0x09f6, seq=2/
   48 10.104.33.30    10.104.33.30    ICMP     Echo (ping) reply    id=0x09f6, seq=2/
  103 192.168.1.103   188.93.155.87   ICMP     Echo (ping) request  id=0x09f7, seq=1/
  104 188.93.155.87   192.168.1.103   ICMP     Echo (ping) reply    id=0x09f7, seq=1/
  115 192.168.1.103   188.93.155.87   ICMP     Echo (ping) request  id=0x09f7, seq=2/
  116 188.93.155.87   192.168.1.103   ICMP     Echo (ping) reply    id=0x09f7, seq=2/
  123 192.168.1.103   188.93.155.87   ICMP     Echo (ping) request  id=0x09f7, seq=3/
  124 188.93.155.87   192.168.1.103   ICMP     Echo (ping) reply    id=0x09f7, seq=3/
  170 10.104.33.30    10.104.33.31    ICMP     Echo (ping) request  id=0x09f8, seq=1/
  171 10.104.33.31    10.104.33.30    ICMP     Echo (ping) reply    id=0x09f8, seq=1/
```

5. *Sniffing ping and dns*

Using the same capture as before, but now with a different **filter**. We want to see both **dns** and **icmp** traffic, so we enter both in the filter field. We put **dns or icmp** in the filter to achieve this. Putting **dns and icmp** would render nothing because there is no packet that matches both protocols.

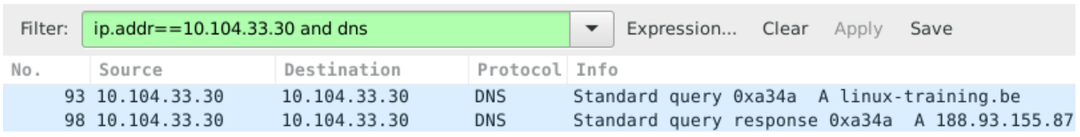


```
Filter: icmp or dns                          Expression...  Clear  Apply  Save

No.   Source          Destination     Protocol Info
   25 10.104.33.30    10.104.33.30    DNS      Standard query 0xa668  A ns1.paul.local
   26 10.104.33.30    10.104.33.30    DNS      Standard query response 0xa668  A 10.10
   31 10.104.33.30    10.104.33.30    ICMP     Echo (ping) request  id=0x09f6, seq=1/2
   32 10.104.33.30    10.104.33.30    ICMP     Echo (ping) reply    id=0x09f6, seq=1/2
```

In the screenshot above you can see that packets 25 and 26 both have 10.104.33.30 as **source** and **destination** ip address. That is because the dns client is the same computer as the dns server. The same is true for packets 31 and 32, since the machine is actually pinging itself.

6. *specific ip address*

This is a screenshot that filters for **dns** packets that contain a certain **ip address**. The filter in use is **ip.addr==10.104.33.30 and dns**. The **and** directive forces each displayed packet to match both conditions.

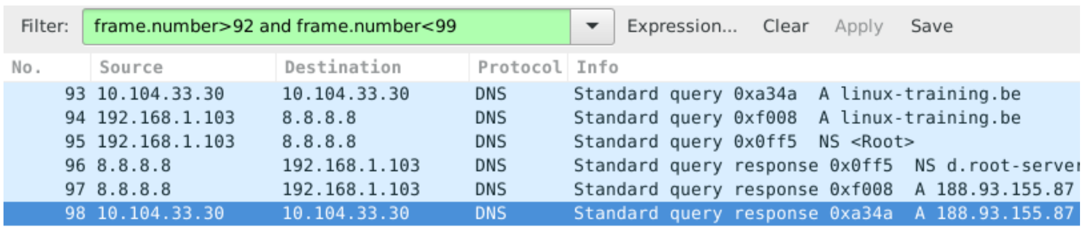


```
Filter: ip.addr==10.104.33.30 and dns        Expression...  Clear  Apply  Save

No.   Source          Destination     Protocol Info
   93 10.104.33.30    10.104.33.30    DNS      Standard query 0xa34a  A linux-training.be
   98 10.104.33.30    10.104.33.30    DNS      Standard query response 0xa34a  A 188.93.155.87
```

Packet 93 is the **dns query** for the A record of linux-training.be. Packet 98 is the response from the **dns server**.

7. *filtering by frame*

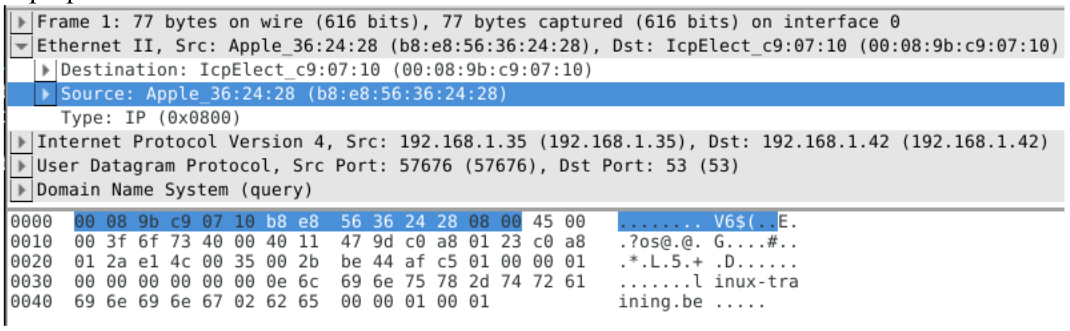
The correct technical term for a **packet** as sniffed is a **frame** (because we sniff on layer two). So to display packets with certain numbers, we use **frame.number** in the filter.



```
Filter: frame.number>92 and frame.number<99  Expression...  Clear  Apply  Save

No.   Source          Destination     Protocol Info
   93 10.104.33.30    10.104.33.30    DNS      Standard query 0xa34a  A linux-training.be
   94 192.168.1.103   8.8.8.8         DNS      Standard query 0xf008  A linux-training.be
   95 192.168.1.103   8.8.8.8         DNS      Standard query 0x0ff5  NS <Root>
   96 8.8.8.8         192.168.1.103   DNS      Standard query response 0x0ff5  NS d.root-server
   97 8.8.8.8         192.168.1.103   DNS      Standard query response 0xf008  A 188.93.155.87
   98 10.104.33.30    10.104.33.30    DNS      Standard query response 0xa34a  A 188.93.155.87
```

8. *looking inside packets*

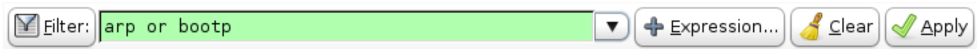
The middle pane can be expanded. When selecting a line in this pane, you can see the corresponding bytes in the frame in the bottom panel. This screenshot shows the middle pane with the source address of my laptop selected.



```
Frame 1: 77 bytes on wire (616 bits), 77 bytes captured (616 bits) on interface 0
Ethernet II, Src: Apple_36:24:28 (b8:e8:56:36:24:28), Dst: IcpElect_c9:07:10 (00:08:9b:c9:07:10)
  Destination: IcpElect_c9:07:10 (00:08:9b:c9:07:10)
  Source: Apple_36:24:28 (b8:e8:56:36:24:28)
  Type: IP (0x0800)
Internet Protocol Version 4, Src: 192.168.1.35 (192.168.1.35), Dst: 192.168.1.42 (192.168.1.42)
User Datagram Protocol, Src Port: 57676 (57676), Dst Port: 53 (53)
Domain Name System (query)

0000  00 08 9b c9 07 10 b8 e8  56 36 24 28 08 00 45 00   ........ V6$(..E.
0010  00 3f 6f 73 40 00 40 11  47 9d c0 a8 01 23 c0 a8   .?os@.@. G....#..
0020  01 2a e1 4c 00 35 00 2b  be 44 af c5 01 00 00 01   .*.L.5.+ .D......
0030  00 00 00 00 00 00 0e 6c  69 6e 75 78 2d 74 72 61   .......l inux-tra
0040  69 6e 69 6e 67 02 62 65  00 00 01 00 01            ining.be .....
```

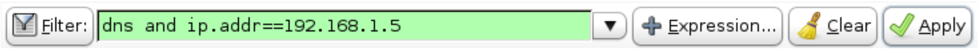
9. *other filter examples*

You can combine two protocols with a logical **or** between them. The example below shows how to filter only **arp** and **bootp** (or **dhcp**) packets.



```
Filter: arp or bootp
```

This example shows how to filter for **dns** traffic containing a certain **ip address**.



```
Filter: dns and ip.addr==192.168.1.5
```

For more details about Wireshark: https://www.wireshark.org/docs/wsug_html/

- ## Port

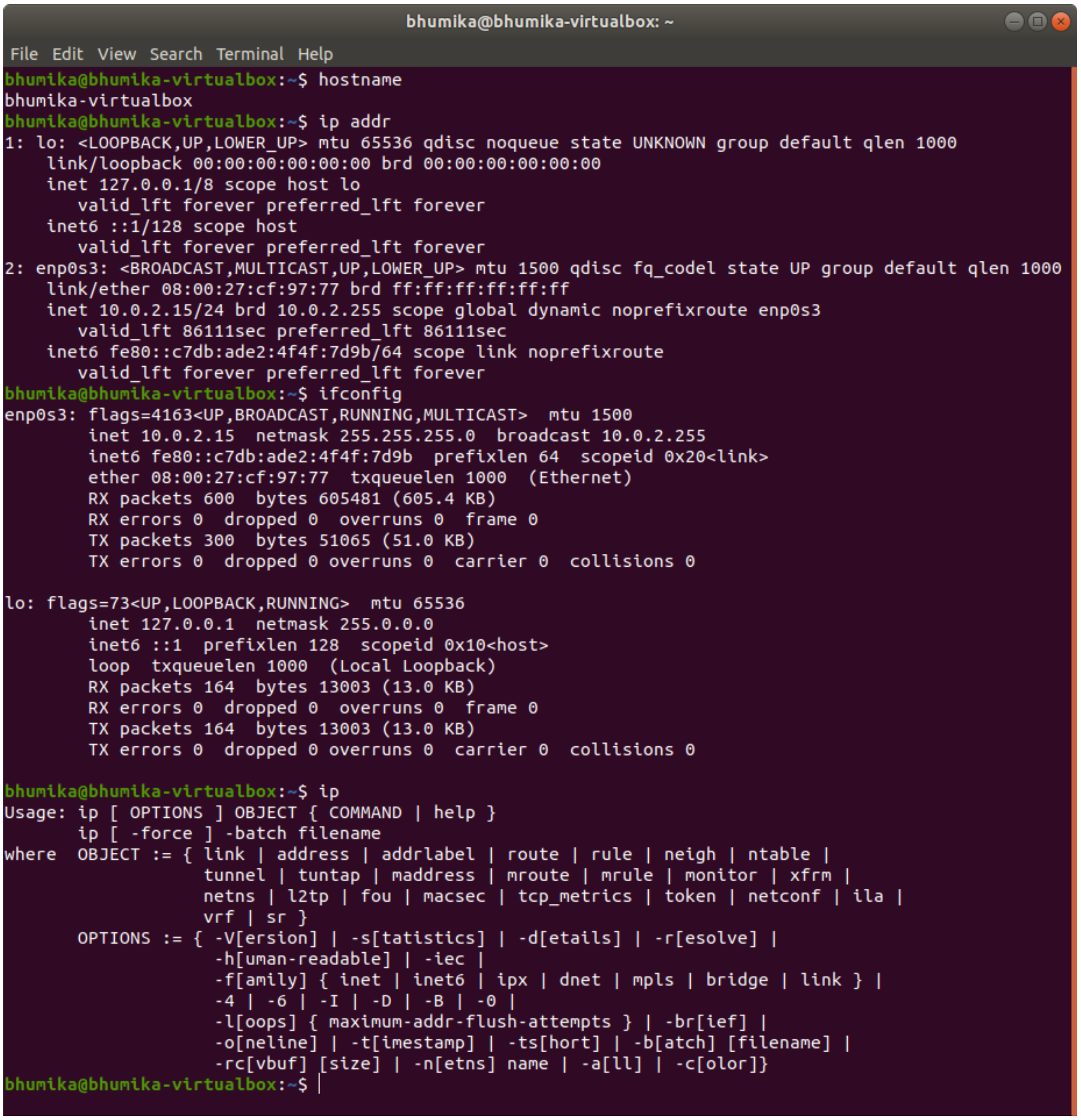
To filter packets based on the desired service or port, use the port filter. For example, capture packets related to a web (HTTP) service by using this command:

```
$ sudo tcpdump -i any -c5 -nn port 80
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on any, link-type LINUX_SLL (Linux cooked), capture size 262144 bytes
09:58:28.790548 IP 192.168.122.98.39330 > 54.204.39.132.80:
Flags [S], seq 1745665159, win 29200, options [mss 1460,sackOK,TS
val 122599140 ecr 0,nop,wscale 7], length 0
09:58:28.834026 IP 54.204.39.132.80 > 192.168.122.98.39330:
Flags [S.], seq 4063583040, ack 1745665160, win 28960, options [mss 1460,sackOK,TS
val 522775728 ecr 122599140,nop,wscale 9], length 0
09:58:28.834093 IP 192.168.122.98.39330 > 54.204.39.132.80: Flags [.], ack 1,
win 229, options [nop,nop,TS val 122599183 ecr 522775728], length 0
09:58:28.834588 IP 192.168.122.98.39330 > 54.204.39.132.80: Flags [P.], seq 1:113,
ack 1, win 229, options [nop,nop,TS val 122599184 ecr 522775728], length 112: HTTP:
GET / HTTP/1.1
09:58:28.878445 IP 54.204.39.132.80 > 192.168.122.98.39330: Flags [.], ack 113,
win 57, options [nop,nop,TS val 522775739 ecr 122599184], length 0
5 packets captured
5 packets received by filter
0 packets dropped by kernel
```

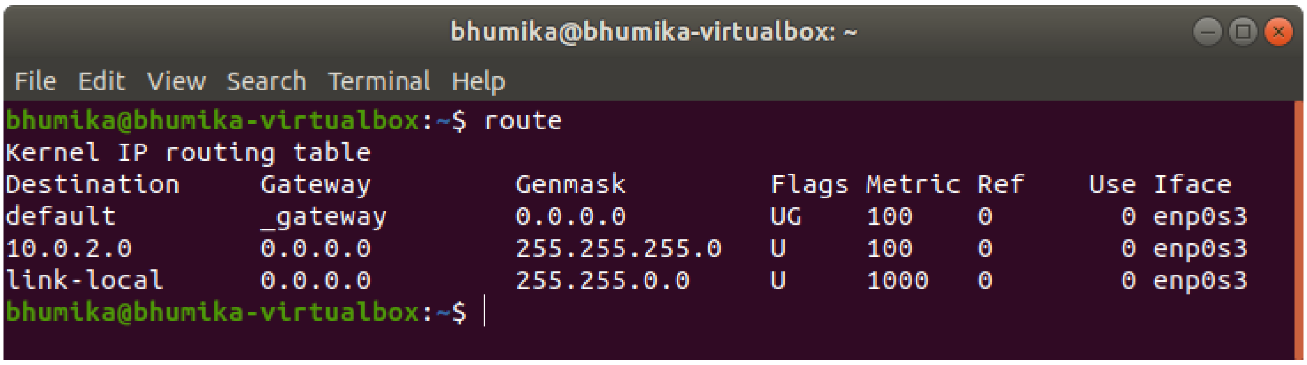
For more networking commands:

- https://www.tecmint.com/linux-networking-commands/
- https://www.computernetworkingnotes.com/networking-tutorials/basic-networking-commands-explained-with-examples.html

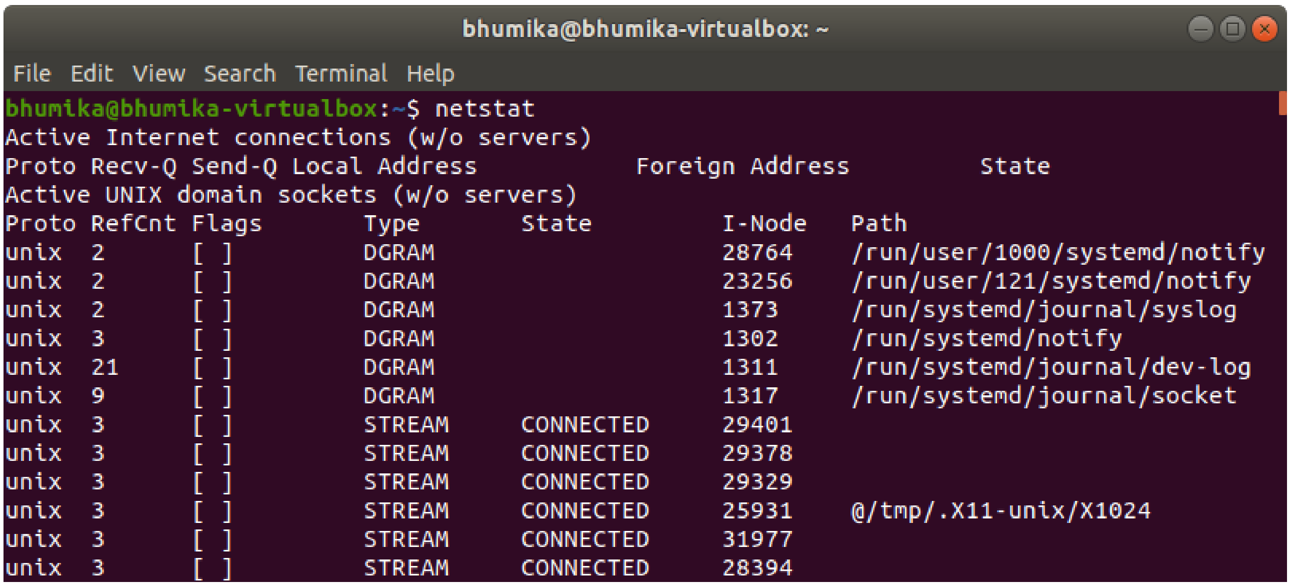
- ***Screenshot of some network related tools and commands:***


bhumika@bhumika-virtualbox: ~
File Edit View Search Terminal Help

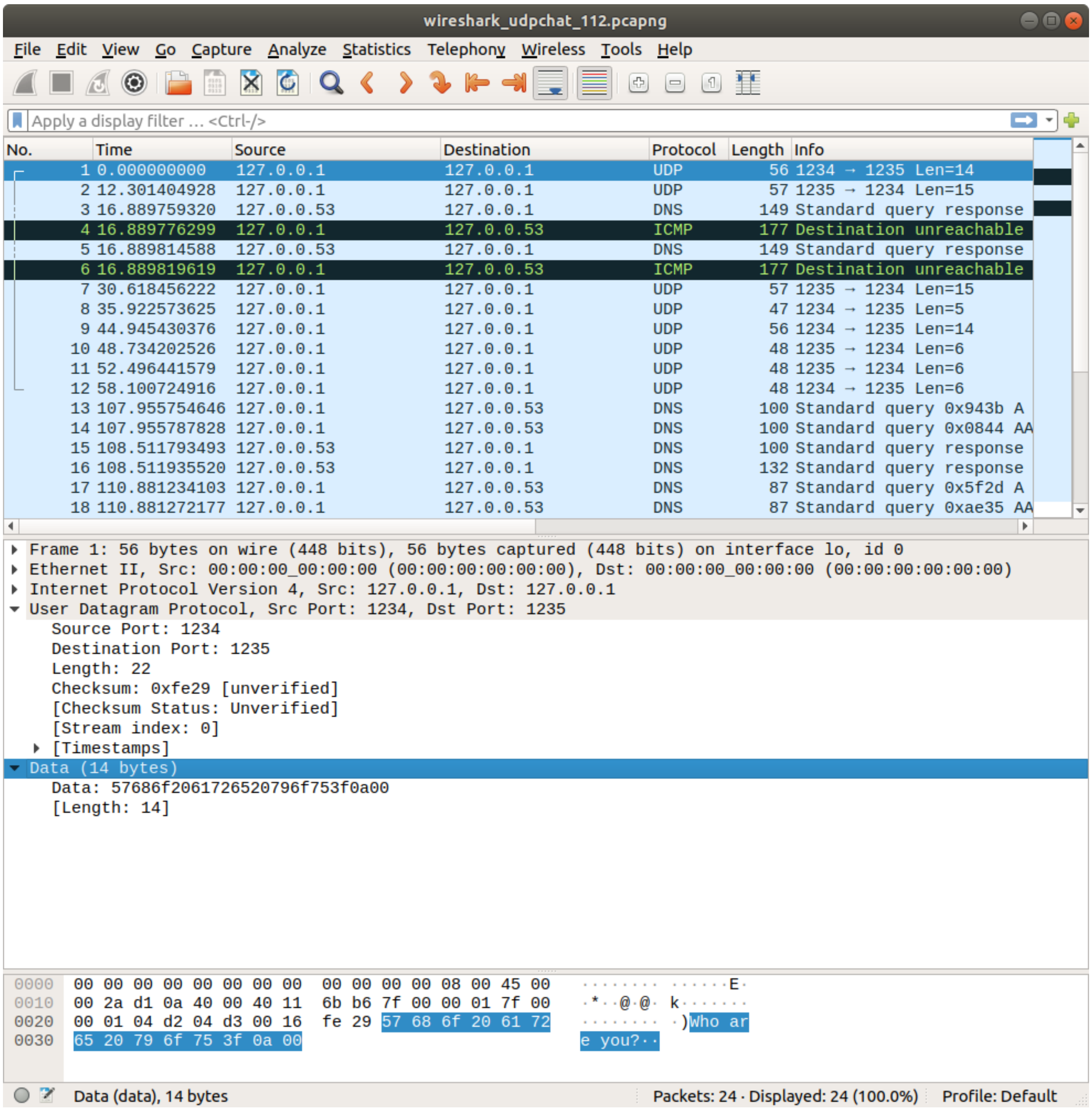
```
bhumika@bhumika-virtualbox:~$ hostname
bhumika-virtualbox
bhumika@bhumika-virtualbox:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:cf:97:77 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic noprefixroute enp0s3
       valid_lft 86111sec preferred_lft 86111sec
    inet6 fe80::c7db:ade2:4f4f:7d9b/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
bhumika@bhumika-virtualbox:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::c7db:ade2:4f4f:7d9b  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:cf:97:77  txqueuelen 1000  (Ethernet)
        RX packets 600  bytes 605481 (605.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 300  bytes 51065 (51.0 KB)
        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 164  bytes 13003 (13.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 164  bytes 13003 (13.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

bhumika@bhumika-virtualbox:~$ ip
Usage: ip [ OPTIONS ] OBJECT { COMMAND | help }
       ip [ -force ] -batch filename
where  OBJECT := { link | address | addrlabel | route | rule | neigh | ntable |
                   tunnel | tuntap | maddress | mroute | mrule | monitor | xfrm |
                   netns | l2tp | fou | macsec | tcp_metrics | token | netconf | ila |
                   vrf | sr }
       OPTIONS := { -V[ersion] | -s[tatistics] | -d[etails] | -r[esolve] |
                    -h[uman-readable] | -iec |
                    -f[amily] { inet | inet6 | ipx | dnet | mpls | bridge | link } |
                    -4 | -6 | -I | -D | -B | -0 |
                    -l[oops] { maximum-addr-flush-attempts } | -br[ief] |
                    -o[neline] | -t[imestamp] | -ts[hort] | -b[atch] [filename] |
                    -rc[vbuf] [size] | -n[etns] name | -a[ll] | -c[olor]}
bhumika@bhumika-virtualbox:~$ 
```


bhumika@bhumika-virtualbox: ~
File Edit View Search Terminal Help

```
bhumika@bhumika-virtualbox:~$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         _gateway        0.0.0.0         UG    100    0        0 enp0s3
10.0.2.0        0.0.0.0         255.255.255.0   U     100    0        0 enp0s3
link-local      0.0.0.0         255.255.0.0     U     1000   0        0 enp0s3
bhumika@bhumika-virtualbox:~$ 
```



```
bhumika@bhumika-virtualbox:~$ netstat
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  2      [ ]         DGRAM                    28764    /run/user/1000/systemd/notify
unix  2      [ ]         DGRAM                    23256    /run/user/121/systemd/notify
unix  2      [ ]         DGRAM                    1373     /run/systemd/journal/syslog
unix  3      [ ]         DGRAM                    1302     /run/systemd/notify
unix  21     [ ]         DGRAM                    1311     /run/systemd/journal/dev-log
unix  9      [ ]         DGRAM                    1317     /run/systemd/journal/socket
unix  3      [ ]         STREAM     CONNECTED     29401
unix  3      [ ]         STREAM     CONNECTED     29378
unix  3      [ ]         STREAM     CONNECTED     29329
unix  3      [ ]         STREAM     CONNECTED     25931    @/tmp/.X11-unix/X1024
unix  3      [ ]         STREAM     CONNECTED     31977
unix  3      [ ]         STREAM     CONNECTED     28394
```



wireshark_udpchat_112.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

```
No.  Time           Source      Destination  Protocol Length Info
   1 0.000000000    127.0.0.1   127.0.0.1    UDP        56 1234 → 1235 Len=14
   2 12.301404928   127.0.0.1   127.0.0.1    UDP        57 1235 → 1234 Len=15
   3 16.889759320   127.0.0.53  127.0.0.1    DNS       149 Standard query response
   4 16.889776299   127.0.0.1   127.0.0.53   ICMP      177 Destination unreachable
   5 16.889814588   127.0.0.53  127.0.0.1    DNS       149 Standard query response
   6 16.889819619   127.0.0.1   127.0.0.53   ICMP      177 Destination unreachable
   7 30.618456222   127.0.0.1   127.0.0.1    UDP        57 1235 → 1234 Len=15
   8 35.922573625   127.0.0.1   127.0.0.1    UDP        47 1234 → 1235 Len=5
   9 44.945430376   127.0.0.1   127.0.0.1    UDP        56 1234 → 1235 Len=14
  10 48.734202526   127.0.0.1   127.0.0.1    UDP        48 1235 → 1234 Len=6
  11 52.496441579   127.0.0.1   127.0.0.1    UDP        48 1235 → 1234 Len=6
  12 58.100724916   127.0.0.1   127.0.0.1    UDP        48 1234 → 1235 Len=6
  13 107.955754646  127.0.0.1   127.0.0.53   DNS       100 Standard query 0x943b A
  14 107.955787828  127.0.0.1   127.0.0.53   DNS       100 Standard query 0x0844 AA
  15 108.511793493  127.0.0.53  127.0.0.1    DNS       100 Standard query response
  16 108.511935520  127.0.0.53  127.0.0.1    DNS       132 Standard query response
  17 110.881234103  127.0.0.1   127.0.0.53   DNS        87 Standard query 0x5f2d A
  18 110.881272177  127.0.0.1   127.0.0.53   DNS        87 Standard query 0xae35 AA
```

```
▸ Frame 1: 56 bytes on wire (448 bits), 56 bytes captured (448 bits) on interface lo, id 0
▸ Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
▸ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
▾ User Datagram Protocol, Src Port: 1234, Dst Port: 1235
    Source Port: 1234
    Destination Port: 1235
    Length: 22
    Checksum: 0xfe29 [unverified]
    [Checksum Status: Unverified]
    [Stream index: 0]
  ▸ [Timestamps]
▾ Data (14 bytes)
    Data: 57686f2061726520796f753f0a00
    [Length: 14]
```

```
0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 00   ········ ······E·
0010  00 2a d1 0a 40 00 40 11  6b b6 7f 00 00 01 7f 00   ·*··@·@· k·······
0020  00 01 04 d2 04 d3 00 16  fe 29 57 68 6f 20 61 72   ········ ·)Who ar
0030  65 20 79 6f 75 3f 0a 00                            e you?··
```

Data (data), 14 bytes   Packets: 24 · Displayed: 24 (100.0%)   Profile: Default