

Linux Networking

Wireless Network Configuration

LinuXpert Systems

WhatsApp: 98841 65649, 90808 01382

E-mail: baskar@linuxpert.in

TABLE OF CONTENTS

1. iw 1

2. iwconfig 15

3. iwlist 22

4. rfkill 34

1. iw

The `iw` command is a modern Linux command-line utility used to configure, monitor, and troubleshoot **wireless network interfaces**. It communicates directly with the Linux kernel through the **nl80211** interface and is the recommended replacement for the older **`iwconfig`** utility.

`iw` can be used to view wireless interfaces, scan for nearby Wi-Fi networks, configure channels and frequencies, check wireless link information, manage interface modes, and perform various wireless networking operations.

Note: `iw` is primarily a low-level wireless configuration tool. For normal Wi-Fi connection management on desktop and server systems, **`nmcli`** or another network manager is generally more convenient.

Syntax

```
iw [OPTIONS] OBJECT COMMAND
```

- **OPTIONS** – Optional global options.
- **OBJECT** – The wireless object to operate on, such as `dev`, `phy`, or `reg`.
- **COMMAND** – The operation to perform on that object.

Common objects

Object	Purpose
`dev`	Work with wireless network interfaces
`phy`	Work with the physical wireless device
`reg`	View or configure wireless regulatory-domain information
`event`	Monitor wireless kernel events
`help`	Display help

1. Check Whether `iw` Is Installed

```
iw --version
```

Example:

```
iw version 6.9
```

If `iw` is not installed on Ubuntu/Debian:

```
sudo apt install iw
```

On Fedora/RHEL:

```
sudo dnf install iw
```

---

2. Display Wireless Interfaces

```
iw dev
```

Example:

```
phy#0
        Interface wlan0
                ifindex 3
                wdev 0x1
                addr 00:11:22:33:44:55
                type managed
```

This provides information about the wireless interface, including:

- Interface name
- Physical device
- MAC address
- Interface type
- Interface index

3. List Wireless Physical Devices

```
iw phy
```

or:

```
iw list
```

`iw list` is particularly useful when determining what capabilities a wireless adapter supports.

It can show:

- Supported interface modes
- Supported frequencies
- Supported channels
- Supported bit rates
- Supported encryption capabilities
- HT/VHT/HE capabilities
- Number of antennas
- TX power capabilities

---

4. Show Wireless Interface Information

```
iw dev wlan0 info
```

Example:

```
Interface wlan0
        ifindex 3
        wdev 0x1
        addr 00:11:22:33:44:55
        type managed
        channel 36 (5180 MHz), width: 80 MHz
```

5. Show Wireless Link Information

```
iw dev wlan0 link
```

Example:

```
Connected to aa:bb:cc:dd:ee:ff
        SSID: Campus-WiFi
        freq: 5180
        RX: 125634 bytes
        TX: 84672 bytes
        signal: -48 dBm
        tx bitrate: 866.7 MBit/s VHT-MCS 9 80MHz short GI
```

This is one of the most useful commands for troubleshooting Wi-Fi connectivity.

Important information includes:

- Connected access point
- SSID
- Frequency
- Signal strength
- RX/TX traffic
- Transmit bitrate
- Wireless mode

---

6. Check Whether the Interface Is Connected

```
iw dev wlan0 link
```

If connected:

```
Connected to aa:bb:cc:dd:ee:ff
        SSID: MyWiFi
```

If not connected:

```
Not connected.
```

7. Scan for Nearby Wi-Fi Networks

Scanning is performed with:

```
sudo iw dev wlan0 scan
```

This can produce a large amount of information.

To display only SSIDs:

```
sudo iw dev wlan0 scan | grep "SSID:"
```

Example:

```
        SSID: Campus-WiFi
        SSID: Guest-WiFi
        SSID: Library-WiFi
```

---

8. Scan for Wi-Fi Networks and Signal Strength

```
sudo iw dev wlan0 scan | grep -E "SSID:|signal:"
```

Example:

```
        signal: -42.00 dBm
        SSID: Campus-WiFi
        signal: -67.00 dBm
        SSID: Guest-WiFi
```

Signal strength is normally expressed in **dBm**.

As a general guide:

Signal	Approximate quality
-30 dBm	Excellent
-50 dBm	Very good
-60 dBm	Good
-70 dBm	Weak
-80 dBm	Very weak

9. Scan a Specific Frequency

```
sudo iw dev wlan0 scan freq 2412
```

For example, 2412 MHz corresponds to Wi-Fi channel 1 in the 2.4 GHz band.

Another example:

```
sudo iw dev wlan0 scan freq 5180
```

---

10. Display Supported Frequencies and Channels

```
iw list
```

Look for:

```
Frequencies:
        * 2412 MHz [1]
        * 2417 MHz [2]
        * 2422 MHz [3]
        ...
        * 5180 MHz [36]
        * 5200 MHz [40]
       ...
```

This is useful for determining whether a wireless adapter supports:

- 2.4 GHz
- 5 GHz
- 6 GHz
- Particular Wi-Fi channels

---

11. Display Current Channel

```
iw dev wlan0 info
```

Example:

```
channel 36 (5180 MHz), width: 80 MHz
```

12. Change the Wireless Channel

First, the interface generally needs to be taken down:

```
sudo ip link set wlan0 down
```

Set the channel:

```
sudo iw dev wlan0 set channel 6
```

Bring the interface back up:

```
sudo ip link set wlan0 up
```

For example:

```
sudo iw dev wlan0 set channel 11
```

Changing the channel manually can interfere with NetworkManager or another network-management service. It is more commonly useful for testing, monitoring, or specialized wireless configurations.

---

13. Display Current Transmit Power

```
iw dev wlan0 info
```

or:

```
sudo iw dev wlan0 get txpower
```

Example:

```
30.00 dBm
```

---

14. Set Transmit Power

For example:

```
sudo iw dev wlan0 set txpower fixed 2000
```

The value is normally specified in mBm, where:

```
100 mBm = 1 dBm
```

Therefore:

```
2000 mBm = 20 dBm
```

Another example:

```
sudo iw dev wlan0 set txpower fixed 1500
```

This corresponds to:

```
15 dBm
```

Regulatory restrictions may limit the actual transmit power.

---

15. Display Regulatory Domain

```
iw reg get
```

Example:

```
global
country IN: DFS-...
        (2402 - 2482 @ 40), ...
        (5170 - 5250 @ 80), ...
```

The regulatory domain determines which frequencies, channels, transmit powers, and DFS rules apply in a particular country or region.

---

16. Set the Regulatory Domain

For India:

```
sudo iw reg set IN
```

Then verify:

```
iw reg get
```

In normal systems, the regulatory domain should generally be managed automatically by the kernel/driver and system configuration. Manually setting it should be done only when you understand the regulatory implications.

---

17. Display Wireless Interface Type

```
iw dev wlan0 info
```

Example:

```
type managed
```

Common interface types include:

Type	Purpose
`managed`	Normal Wi-Fi client
`AP`	Access point
`monitor`	Wireless monitoring
`adhoc`	Ad-hoc networking
`mesh point`	Mesh networking
`P2P-client`	Wi-Fi Direct client
`P2P-GO`	Wi-Fi Direct group owner

18. Change Interface to Monitor Mode

First:

```
sudo ip link set wlan0 down
```

Change mode:

```
sudo iw dev wlan0 set type monitor
```

Bring it up:

```
sudo ip link set wlan0 up
```

Verify:

```
iw dev wlan0 info
```

You should see:

```
type monitor
```

To return to normal client mode:

```
sudo ip link set wlan0 down
```

```
sudo iw dev wlan0 set type managed
```

```
sudo ip link set wlan0 up
```

19. Display Wireless Station Statistics

```
sudo iw dev wlan0 station dump
```

This displays information about wireless stations associated with an access point.

Typical information includes:

- MAC address
- Signal strength
- RX bytes
- TX bytes
- RX packets
- TX packets
- Bitrate
- Connection time
- Retransmissions
- Packet loss-related statistics

---

20. Display a Specific Station

```
sudo iw dev wlan0 station get AA:BB:CC:DD:EE:FF
```

Example:

```
sudo iw dev wlan0 station get 00:11:22:33:44:55
```

This is particularly useful when the wireless interface is operating as an access point.

---

21. Display Wireless Events

```
iw event
```

This monitors wireless events generated by the kernel.

It can help troubleshoot events such as:

- Interface changes
- Authentication

- Association
- Disassociation
- Channel changes
- Regulatory changes

---

22. Monitor Wireless Events Continuously

```
sudo iw event -t
```

The `-t` option adds timestamps to events.

This can be useful when investigating intermittent wireless problems.

---

23. Display Help

General help:

```
iw help
```

Help for `dev`:

```
iw dev help
```

Help for a particular operation:

```
iw dev wlan0 set help
```

---

24. Find the Wireless Interface Name

On modern Linux systems, the interface might not be called `wlan0`.

Use:

```
iw dev
```

For example:

```
phy#0
        Interface wlp2s0
                ifindex 3
                type managed
```

Here the wireless interface is:

```
wlp2s0
```

You can then use:

```
iw dev wlp2s0 link
```

---

25. `iw` (vs) `iwconfig`

`iw` is the modern replacement for many functions historically performed by **`iwconfig`**.

Task	Old command	Modern command
Show wireless information	`iwconfig`	`iw dev wlan0 info`
Show link status	`iwconfig`	`iw dev wlan0 link`
Scan networks	`iwlist wlan0 scan`	`iw dev wlan0 scan`
Set channel	`iwconfig wlan0 channel 6`	`iw dev wlan0 set channel 6`
Set TX power	`iwconfig wlan0 txpower`	`iw dev wlan0 set txpower ...`
Set mode	`iwconfig wlan0 mode monitor`	`iw dev wlan0 set type monitor`

The older **Wireless Extensions** interface used by **`iwconfig`** has largely been superseded by **nl80211**, which is used by **`iw`**.

---

26. `iw` (vs) `nmcli`

These commands serve somewhat different purposes.

Feature	iw	nmcli
Low-level Wi-Fi control	Excellent	Limited
Scan Wi-Fi	Yes	Yes
View signal	Yes	Yes
Connect to Wi-Fi	Not normally	Excellent
Configure IP address	No	Yes
Configure DNS	No	Yes
Manage NetworkManager	No	Yes
Monitor mode	Yes	Limited
Wireless driver capabilities	Excellent	Limited
Regulatory information	Yes	Limited

For example, `iw` can inspect the wireless link:

```
iw dev wlan0 link
```

while `nmcli` can manage a Wi-Fi connection:

```
nmcli device wifi connect "Campus-WiFi" password "password"
```

Thus, `iw` **is primarily a wireless kernel/interface tool, whereas** `nmcli` **is a complete network-management tool.**

---

27. Useful `iw` Commands at a Glance

Command	Purpose
`iw dev`	List wireless interfaces
`iw list`	Display wireless hardware capabilities
`iw dev wlan0 info`	Display interface information
`iw dev wlan0 link`	Display current Wi-Fi connection
`iw dev wlan0 scan`	Scan for Wi-Fi networks
`iw dev wlan0 scan | grep SSID`	List discovered SSIDs
`iw dev wlan0 get txpower`	Display transmit power
`iw dev wlan0 set txpower fixed 2000`	Set transmit power
`iw dev wlan0 set channel 6`	Change channel
`iw dev wlan0 station dump`	Display connected stations
`iw dev wlan0 station get MAC`	Display a particular station
`iw reg get`	Display regulatory information
`iw reg set IN`	Set India regulatory domain
`iw event`	Monitor wireless events
`iw dev wlan0 set type monitor`	Enable monitor mode
`iw dev wlan0 set type managed`	Return to managed mode
`iw help`	Display command help

Important distinction

For Linux networking, it is useful to position the wireless commands in this order:

```
iw
  ├── Wireless hardware/interface inspection
  ├── Wireless scanning
  ├── Channel/frequency management
  ├── Signal/link monitoring
  ├── Regulatory-domain management
  ├── Monitor-mode configuration
  └── Low-level wireless troubleshooting
```

Whereas:

```
nmcli
  ├── Wi-Fi connection management
  ├── SSID/password configuration
  ├── IP configuration
  ├── DNS configuration
  ├── Routing
  └── NetworkManager administration
```

This makes `iw` **an important command for students learning the lower layers of Linux wireless networking**, particularly when they need to understand the relationship between the wireless adapter, kernel, driver, `nl80211`, channels, frequencies, and the network-management layer.

2. iwconfig

The **iwconfig** command is a Linux wireless networking utility used to configure and display information about **wireless network interfaces**. It is part of the older **Wireless Tools for Linux** package and works with wireless extensions supported by the kernel and wireless drivers.

It can be used to view and configure parameters such as the wireless network name (ESSID), operating mode, frequency, channel, transmit power, sensitivity, bit rate, encryption settings, and power management.

Note:

`iwconfig` is a legacy wireless networking command. On modern Linux systems, **`iw`** is generally preferred for low-level Wi-Fi configuration, while **`nmcli`** is preferred when NetworkManager manages the connection.

Syntax

```
iwconfig [interface]
```

To configure a wireless interface:

```
iwconfig interface OPTION
```

- **interface** – The wireless network interface, such as `wlan0` or `wlp2s0`.
- **OPTION** – The wireless parameter to display or configure.

Common Examples

1. Display wireless interface information

```
iwconfig
```

Example:

```
wlp2s0    IEEE 802.11  ESSID:"CampusWiFi"
          Mode:Managed  Frequency:2.437 GHz
          Access Point: AA:BB:CC:DD:EE:FF
          Bit Rate=54 Mb/s
          Tx-Power=20 dBm
          Link Quality=70/70
          Signal level=-40 dBm
```

This displays information about all wireless interfaces.

2. Display information for a specific interface

```
iwconfig wlp2s0
```

This displays the wireless configuration and status of `wlp2s0`.

---

3. Set the wireless interface ESSID

```
sudo iwconfig wlp2s0 essid "CampusWiFi"
```

This sets the ESSID (wireless network name) for the interface.

---

4. Set wireless operating mode

```
sudo iwconfig wlp2s0 mode managed
```

Common modes include:

```
managed
ad-hoc
master
monitor
auto
```

For example:

```
sudo iwconfig wlp2s0 mode managed
```

sets the interface to normal client/station mode.

---

5. Set the wireless frequency

```
sudo iwconfig wlp2s0 freq 2.437G
```

This attempts to configure the interface to **2.437 GHz**, corresponding to channel 6 in the 2.4-GHz Wi-Fi band.

6. Set the wireless channel

```
sudo iwconfig wlp2s0 channel 6
```

This configures the wireless interface to channel 6.

You can also use:

```
sudo iwconfig wlp2s0 channel 11
```

---

7. Set the wireless bit rate

```
sudo iwconfig wlp2s0 rate 54M
```

This requests a bit rate of **54 Mb/s**.

To allow automatic rate selection:

```
sudo iwconfig wlp2s0 rate auto
```

---

8. Configure transmit power

```
sudo iwconfig wlp2s0 txpower 20dBm
```

This attempts to set the transmit power to 20 dBm.

To enable automatic transmit-power selection:

```
sudo iwconfig wlp2s0 txpower auto
```

---

9. Configure power management

Enable power management:

```
sudo iwconfig wlp2s0 power on
```

Disable it:

```
sudo iwconfig wlp2s0 power off
```

Display the current setting:

```
iwconfig wlp2s0
```

10. Set wireless sensitivity

```
sudo iwconfig wlp2s0 sens 3
```

The meaning and supported range depend on the wireless driver.

---

11. Set the access point MAC address

```
sudo iwconfig wlp2s0 ap AA:BB:CC:DD:EE:FF
```

This specifies the access point to associate with, when supported by the driver.

---

12. Configure wireless encryption

Older wireless drivers supported commands such as:

```
sudo iwconfig wlp2s0 key s:password
```

or:

```
sudo iwconfig wlp2s0 key 1234567890
```

However, **WEP configuration using `iwconfig` is obsolete and insecure**. Modern WPA/WPA2/WPA3 networks should normally be configured through **NetworkManager (`nmcli`)**, **`wpa_supplicant`**, or another modern Wi-Fi management system.

Important `iwconfig` Options

Option	Purpose	Example
`essid`	Set wireless network name	`iwconfig wlp2s0 essid "CampusWiFi"`
`mode`	Set operating mode	`iwconfig wlp2s0 mode managed`
`freq`	Set frequency	`iwconfig wlp2s0 freq 2.437G`
`channel`	Set channel	`iwconfig wlp2s0 channel 6`
`rate`	Set bit rate	`iwconfig wlp2s0 rate 54M`
`txpower`	Set transmit power	`iwconfig wlp2s0 txpower 20dBm`
`sens`	Set sensitivity	`iwconfig wlp2s0 sens 3`
`power`	Configure power management	`iwconfig wlp2s0 power on`
`ap`	Specify access point	`iwconfig wlp2s0 ap  AA:BB:CC:DD:EE:FF`
`key`	Configure legacy WEP key	`iwconfig wlp2s0 key ...`

Finding Wireless Interfaces

Before using `iwconfig`, identify the wireless interface:

```
ip link
```

or:

```
iw dev
```

Typical wireless interface names include:

```
wlan0
wlan1
wlp2s0
wlp3s0
```

You can then run:

```
iwconfig wlp2s0
```

Checking Whether an Interface Is Wireless

Running:

```
iwconfig
```

will normally list wireless interfaces and their wireless parameters. An Ethernet interface may appear with:

```
no wireless extensions.
```

For example:

```
enp3s0    no wireless extensions.

wlp2s0    IEEE 802.11  ESSID:"CampusWiFi"
          Mode:Managed
          ...
```

`iwconfig` (vs) `iw` (vs) `nmcli`

Command	Primary Purpose	Modern Status
`iwconfig`	Legacy wireless configuration	Legacy
`iw`	Low-level 802.11/Wi-Fi configuration	Modern
`nmcli`	NetworkManager-based network configuration	Modern and recommended when NetworkManager is used
`ip`	Interface/IP/routing configuration	Modern

For example, the older:

```
iwconfig wlp2s0
```

can generally be supplemented by:

```
iw dev wlp2s0 info
```

and:

```
iw dev wlp2s0 link
```

for more modern wireless information.

Practical Troubleshooting Examples

Check whether Wi-Fi is detected:

```
iwconfig
```

Check the wireless interface:

```
iwconfig wlp2s0
```

Check whether it is associated:

```
iwconfig wlp2s0
```

or, preferably:

```
iw dev wlp2s0 link
```

Check available wireless networks using the modern tool:

```
sudo iw dev wlp2s0 scan
```

Check NetworkManager's view:

```
nmcli device wifi list
```

Important Limitation

`iwconfig` **does not itself provide a complete modern Wi-Fi connection-management system**.

For WPA/WPA2/WPA3 authentication, DHCP, saved wireless profiles, roaming, and enterprise Wi-Fi, use **NetworkManager/nmcli** or another appropriate modern network-management framework.

For Linux command reference, a useful way to position the commands is:

```
iwconfig  →   Legacy wireless interface configuration
iw        →   Modern low-level Wi-Fi configuration
nmcli     →   Modern complete network connection management
ip        →   IP address, interface and routing configuration
```

3. iwlist

The **`iwlist`** command is a Linux wireless networking utility used to display detailed information about wireless interfaces. It is part of the **Wireless Tools for Linux** package and works with the older **Wireless Extensions (WEXT)** interface.

`iwlist` is primarily used to **scan for nearby Wi-Fi networks, display supported wireless capabilities, show available frequencies/channels, and inspect wireless parameters**.

Note:

`iwlist` is considered a legacy wireless networking command. Modern Linux systems generally use **`iw`** and **`nmcli`** instead. However, **`iwlist`** remains useful for understanding traditional Linux wireless configuration and for working with older wireless drivers.

Syntax

```
iwlist [interface] [command]
```

or:

```
iwlist [interface] [command] [options]
```

- **interface** – Wireless network interface such as `wlan0` or `wlp2s0`.
- **command** – Information to display, such as `scan`, `frequency`, `channel`, or `rate`.

Common `iwlist` Commands

Command	Purpose
`iwlist wlan0 scan`	Scan for nearby wireless networks
`iwlist wlan0 frequency`	Display supported frequencies/channels
`iwlist wlan0 channel`	Display wireless channels
`iwlist wlan0 rate`	Display supported bit rates
`iwlist wlan0 bitrate`	Display current/supported bit rates
`iwlist wlan0 txpower`	Display transmit-power information
`iwlist wlan0 power`	Display power-management information
`iwlist wlan0 retry`	Display retry-limit information
`iwlist wlan0 ap`	Display associated access point information
`iwlist wlan0 event`	Display wireless event information
`iwlist wlan0 keys`	Display available encryption-key information
`iwlist wlan0 auth`	Display authentication parameters
`iwlist wlan0 modulation`	Display modulation information
`iwlist wlan0 version`	Display Wireless Tools version

1. Identify the Wireless Interface

Before using `iwlist`, determine the name of the wireless interface.

```
ip link show
```

Example:

```
1: lo: <LOOPBACK,UP,LOWER_UP> ...
2: enp3s0: <BROADCAST,MULTICAST,UP,LOWER_UP> ...
3: wlp2s0: <BROADCAST,MULTICAST> ...
```

Here, **`wlp2s0`** is the wireless interface.

You can also use:

```
iwconfig
```

Example:

```
wlp2s0    IEEE 802.11  ESSID:"CampusWiFi"
          Mode:Managed  Frequency:2.437 GHz
          Access Point: AA:BB:CC:DD:EE:FF
```

---

2. Scan for Nearby Wi-Fi Networks

The most commonly used `iwlist` operation is `scan`.

```
sudo iwlist wlp2s0 scan
```

Example output:

```
wlp2s0    Scan completed :
          Cell 01 - Address: 00:11:22:33:44:55
                    Channel:6
                    Frequency:2.437 GHz
                    Quality=70/70
                    Signal level=-40 dBm
                    Encryption key:on
                    ESSID:"CampusWiFi"

          Cell 02 - Address: AA:BB:CC:DD:EE:FF
                    Channel:11
                    Frequency:2.462 GHz
                    Quality=45/70
                    Signal level=-65 dBm
                    Encryption key:on
                    ESSID:"GuestWiFi"
```

The scan provides useful information such as:

- Access Point MAC address
- Channel
- Frequency
- Signal quality
- Signal level
- Encryption status
- ESSID/network name

Scan using interface name

```
sudo iwlist wlan0 scan
```

or:

```
sudo iwlist wlp2s0 scan
```

---

3. Display Wireless Frequencies

To display the frequencies supported by the wireless adapter:

```
iwlist wlp2s0 frequency
```

Example:

```
wlp2s0    13 channels in total; available frequencies :
          Channel 01 : 2.412 GHz
          Channel 02 : 2.417 GHz
          Channel 03 : 2.422 GHz
          Channel 04 : 2.427 GHz
          Channel 05 : 2.432 GHz
          Channel 06 : 2.437 GHz
          Channel 07 : 2.442 GHz
          Channel 08 : 2.447 GHz
          Channel 09 : 2.452 GHz
          Channel 10 : 2.457 GHz
          Channel 11 : 2.462 GHz
          Channel 12 : 2.467 GHz
          Channel 13 : 2.472 GHz
```

This is useful for determining which frequency bands and channels an adapter supports.

4. Display Wireless Channels

The `channel` operation displays the available wireless channels.

```
iwlist wlp2s0 channel
```

Example:

```
wlp2s0    13 channels in total; available frequencies :
          Channel 01 : 2.412 GHz
          Channel 02 : 2.417 GHz
          Channel 03 : 2.422 GHz
          ...
          Channel 13 : 2.472 GHz
          Current Frequency:2.437 GHz (Channel 6)
```

The current channel can therefore be identified from the output.

---

5. Display Supported Bit Rates

Use:

```
iwlist wlp2s0 rate
```

Example:

```
wlp2s0    4 available bit-rates :
          1 Mb/s
          2 Mb/s
          5.5 Mb/s
          11 Mb/s
```

On some older wireless adapters, additional rates may be displayed:

```
6 Mb/s
9 Mb/s
12 Mb/s
18 Mb/s
24 Mb/s
36 Mb/s
48 Mb/s
54 Mb/s
```

This information describes the data rates reported through the wireless driver.

6. Display Bit-Rate Information

The `bitrate` command can be used to examine wireless bit-rate information.

```
iwlist wlp2s0 bitrate
```

Example:

```
wlp2s0    unknown bit-rate information.
          Current Bit Rate=54 Mb/s
```

On supported hardware, the output may contain more detailed information.

---

7. Display Transmit Power

The wireless adapter's transmit-power information can be displayed with:

```
iwlist wlp2s0 txpower
```

Example:

```
wlp2s0    5 available transmit-powers :
          5 dBm
          10 dBm
          15 dBm
          20 dBm
          23 dBm
          Current Tx-Power=20 dBm
```

Transmit power affects the wireless transmission range, subject to hardware and regulatory restrictions.

---

8. Display Power Management Information

Use:

```
iwlist wlp2s0 power
```

Example:

```
wlp2s0    Current mode:off
```

Power management can reduce power consumption, particularly on laptops.

9. Display Retry Information

The `retry` command displays wireless retry parameters.

```
iwlist wlp2s0 retry
```

Example:

```
wlp2s0    Fixed limit; min limit:8
          Current mode:limit
          Current limit:8
```

Retry parameters determine how many times transmission attempts may be retried under certain conditions.

---

10. Display Access Point Information

The `ap` operation can display information about the access point associated with the interface.

```
iwlist wlp2s0 ap
```

Example:

```
wlp2s0    Associated with 00:11:22:33:44:55
```

If the interface is not associated:

```
wlp2s0    No Access Point
```

---

11. Display Wireless Events

Use:

```
iwlist wlp2s0 event
```

Wireless events are notifications generated by the wireless subsystem.

For example, events may indicate:

- Association
- Disassociation
- Connection changes
- Wireless parameter changes

This option is particularly useful for troubleshooting older wireless drivers.

12. Display Encryption-Key Information

The `keys` operation can display information about wireless encryption keys.

```
sudo iwlist wlp2s0 keys
```

Example:

```
wlp2s0    2 key sizes :
          40bits
          104bits
```

The command does **not normally display the actual Wi-Fi password in readable form**.

---

13. Display Authentication Information

Use:

```
iwlist wlp2s0 auth
```

This displays authentication-related parameters supported by the wireless driver.

Example:

```
wlp2s0    Authentication information :
          Authentication algorithms :
              Open System
              Shared Key
```

The exact output depends on the driver.

---

14. Display Modulation Information

Some older wireless adapters support:

```
iwlist wlp2s0 modulation
```

Example:

```
wlp2s0    2 available modulations :
          CCK
          OFDM
```

This operation is mainly relevant to older wireless hardware and drivers.

15. Display Wireless Tools Version

Use:

```
iwlist --version
```

Example:

```
iwlist Wireless-Tools version 30
```

You can also try:

```
iwlist version
```

---

16. Scan and Extract ESSIDs

A complete scan can generate a large amount of information:

```
sudo iwlist wlp2s0 scan
```

To display only ESSID lines:

```
sudo iwlist wlp2s0 scan | grep ESSID
```

Example:

```
          ESSID:"CampusWiFi"
          ESSID:"GuestWiFi"
          ESSID:"Library"
          ESSID:"Engineering"
```

---

17. Extract Signal Levels

```
sudo iwlist wlp2s0 scan | grep "Signal level"
```

Example:

```
          Signal level=-42 dBm
          Signal level=-58 dBm
          Signal level=-71 dBm
```

Generally, a signal closer to **0 dBm** represents a stronger received signal.

A rough interpretation is:

Signal	Approximate quality
-30 dBm	Excellent
-50 dBm	Very good
-60 dBm	Good
-70 dBm	Weak
-80 dBm	Very weak
-90 dBm	Extremely weak

These values are approximate and should not be treated as universal quality thresholds.

---

18. Extract Wireless Channels

```
sudo iwlist wlp2s0 scan | grep Channel
```

Example:

```
          Channel:1
          Channel:6
          Channel:11
```

This can be useful when investigating channel congestion.

---

19. Extract Access Point MAC Addresses

```
sudo iwlist wlp2s0 scan | grep Address
```

Example:

```
Cell 01 - Address: 00:11:22:33:44:55
Cell 02 - Address: AA:BB:CC:DD:EE:FF
Cell 03 - Address: 12:34:56:78:9A:BC
```

The address is the **BSSID**, normally the MAC address of the wireless access point.

20. Check Whether the Interface Supports Scanning

Run:

```
iwlist wlp2s0 scan
```

If scanning is unsupported, you may see an error similar to:

```
wlp2s0    Interface doesn't support scanning.
```

Possible reasons include:

- Driver does not support Wireless Extensions.
- Wireless interface is disabled.
- Interface is not available.
- Hardware does not support the requested operation.
- Modern driver exposes capabilities through `nl80211` rather than WEXT.

---

21. Bring the Interface Up Before Scanning

If the interface is down:

```
sudo ip link set wlp2s0 up
```

Then:

```
sudo iwlist wlp2s0 scan
```

Check its state with:

```
ip link show wlp2s0
```

---

22. Using `iwlist` with Multiple Wireless Interfaces

First identify the interfaces:

```
iw dev
```

Example:

```
phy#0
    Interface wlp2s0
        type managed
```

```
phy#1
    Interface wlan1
        type managed
```

You can then scan a particular interface:

```
sudo iwlist wlan1 scan
```

---

23. `iwlist` (vs) `iw`

`iwlist` belongs to the older **Wireless Tools** ecosystem, while **`iw`** uses the modern **nl80211** wireless interface.

Feature	iwlist	iw
Wireless API	Wireless Extensions (WEXT)	nl80211
Status	Legacy	Modern
Wi-Fi scanning	Yes	Yes
Frequency information	Yes	Yes
Channel information	Yes	Yes
Modern Wi-Fi features	Limited	Much better
Modern driver support	Limited	Recommended
Typical use today	Legacy systems/learning	Current Linux systems

For example, the traditional command:

```
sudo iwlist wlp2s0 scan
```

would normally be replaced by:

```
sudo iw dev wlp2s0 scan
```

---

24. `iwlist` (vs) `nmcli`

`nmcli` is another important modern Linux networking utility.

Command	Primary purpose
`iwlist`	Legacy wireless information/scanning
`iw`	Low-level modern Wi-Fi configuration and monitoring
`nmcli`	NetworkManager configuration and management
`iwconfig`	Legacy wireless configuration
`ip`	General network interface/IP/routing configuration

Important Note

`iwlist` **does not configure a Wi-Fi connection in the same way that** `nmcli` **does**. Its most important educational use is to inspect wireless capabilities and scan for networks.

For example:

```
sudo iwlist wlp2s0 scan
```

is primarily an **information-gathering command**, whereas:

```
nmcli device wifi connect "CampusWiFi" password "mypassword"
```

is a **network-connection/configuration command**.

Summary

The most important **`iwlist`** commands to remember are:

```
iwlist wlan0 scan

iwlist wlan0 frequency

iwlist wlan0 channel

iwlist wlan0 rate

iwlist wlan0 bitrate

iwlist wlan0 txpower

iwlist wlan0 power

iwlist wlan0 retry

iwlist wlan0 ap

iwlist wlan0 event

iwlist wlan0 keys

iwlist wlan0 auth

iwlist wlan0 modulation
```

In short: `iwlist` is primarily a **legacy wireless information and scanning utility**, whereas `iw` **is the modern replacement for low-level wireless operations and** `nmcli` **is generally preferred for NetworkManager-based Wi-Fi configuration.**

4. rfkill

The **rfkill** command is a Linux utility used to **enable, disable, and monitor wireless communication devices** such as Wi-Fi, Bluetooth, WWAN (mobile broadband), and other radio transmitters. It is particularly useful for controlling whether wireless hardware is **blocked (disabled)** or **unblocked (enabled)**.

The command interfaces with the Linux kernel's **RFKill subsystem**, which provides a standardized mechanism for controlling radio transmitters.

`rfkill` is commonly used for **wireless troubleshooting, power management, airplane-mode functionality, and enabling/disabling Wi-Fi or Bluetooth hardware**.

Syntax

```
rfkill [OPTIONS] COMMAND
```

Common commands include:

```
rfkill list

rfkill block DEVICE

rfkill unblock DEVICE

rfkill event

rfkill help
```

---

1. Display all wireless devices

```
rfkill list
```

Example output:

```
0: phy0: Wireless LAN
        Soft blocked: no
        Hard blocked: no
1: hci0: Bluetooth
        Soft blocked: no
        Hard blocked: no
```

The output provides:

- **Device number** – Identifier assigned by rfkill.
- **Device type** – Wireless LAN, Bluetooth, WWAN, etc.
- **Soft blocked** – Software has disabled the radio.
- **Hard blocked** – Hardware or firmware has disabled the radio.

---

2. Display Wi-Fi status

```
rfkill list wifi
```

Example:

```
0: phy0: Wireless LAN
        Soft blocked: no
        Hard blocked: no
```

If you see:

```
Soft blocked: yes
```

Wi-Fi has been disabled by software.

If you see:

```
Hard blocked: yes
```

Wi-Fi has been disabled by a physical switch, BIOS/UEFI setting, firmware, or hardware mechanism.

---

3. Display Bluetooth status

```
rfkill list bluetooth
```

Example:

```
1: hci0: Bluetooth
        Soft blocked: no
        Hard blocked: no
```

This allows you to determine whether Bluetooth is currently blocked.

4. Block Wi-Fi

```
sudo rfkill block wifi
```

This disables the Wi-Fi radio through software.

Check the status:

```
rfkill list wifi
```

Output may show:

```
Soft blocked: yes
Hard blocked: no
```

---

5. Unblock Wi-Fi

```
sudo rfkill unblock wifi
```

Check the status:

```
rfkill list wifi
```

Expected output:

```
Soft blocked: no
Hard blocked: no
```

This allows the Wi-Fi radio to be used again.

---

6. Block Bluetooth

```
sudo rfkill block bluetooth
```

Check:

```
rfkill list bluetooth
```

Example:

```
1: hci0: Bluetooth
        Soft blocked: yes
        Hard blocked: no
```

7. Unblock Bluetooth

```
sudo rfkill unblock bluetooth
```

Check:

```
rfkill list bluetooth
```

---

8. Block a specific device

Instead of specifying the device type, you can use its rfkill ID.

First:

```
rfkill list
```

Example:

```
0: phy0: Wireless LAN
        Soft blocked: no
        Hard blocked: no
1: hci0: Bluetooth
        Soft blocked: no
        Hard blocked: no
```

To block device `0`:

```
sudo rfkill block 0
```

To unblock it:

```
sudo rfkill unblock 0
```

---

9. Block all wireless radios

```
sudo rfkill block all
```

This can disable all radio devices managed by RFKill, including Wi-Fi and Bluetooth.

Check:

```
rfkill list
```

For example:

```
0: phy0: Wireless LAN
        Soft blocked: yes
        Hard blocked: no
1: hci0: Bluetooth
        Soft blocked: yes
        Hard blocked: no
```

---

10. Unblock all wireless radios

```
sudo rfkill unblock all
```

Verify:

```
rfkill list
```

---

11. Monitor RFKill events

The `event` command can be used to monitor changes in RFKill state.

```
rfkill event
```

The command waits for RFKill events.

For example, when a wireless device is blocked or unblocked, information about the event may be displayed.

Press:

```
Ctrl+C
```

to stop monitoring.

This is useful for troubleshooting **airplane-mode switches, laptop hotkeys, and hardware wireless switches**.

12. Check whether Wi-Fi is blocked

A quick check can be performed with:

```
rfkill list wifi
```

If the output contains:

```
Soft blocked: yes
```

run:

```
sudo rfkill unblock wifi
```

If it contains:

```
Hard blocked: yes
```

unblocking with **`rfkill`** may not solve the problem because the radio is being disabled at the hardware/firmware level.

---

13. Soft block vs. hard block

One of the most important concepts when using `rfkill` is the distinction between **soft blocking** and **hard blocking**.

Soft block

```
Soft blocked: yes
```

The wireless device has been disabled by software.

For example:

```
sudo rfkill block wifi
```

can produce:

```
Soft blocked: yes
```

A soft block can normally be removed with:

```
sudo rfkill unblock wifi
```

Hard block

```
Hard blocked: yes
```

The wireless device has been disabled by something outside the normal software RFKill control.

Possible causes include:

- Physical wireless switch
- Laptop Fn key
- Airplane-mode switch
- BIOS/UEFI configuration
- Firmware
- Hardware-level power control

Running:

```
sudo rfkill unblock wifi
```

may **not** remove a hard block.

---

14. Find all blocked wireless devices

```
rfkill list
```

Look for:

```
Soft blocked: yes
```

or:

```
Hard blocked: yes
```

For example:

```
0: phy0: Wireless LAN
        Soft blocked: yes
        Hard blocked: no

1: hci0: Bluetooth
        Soft blocked: no
        Hard blocked: no
```

Here, Wi-Fi is software-blocked while Bluetooth is available.

15. Troubleshoot Wi-Fi that cannot be enabled

Suppose Wi-Fi is not working.

First check:

```
rfkill list
```

If you see:

```
Soft blocked: yes
```

run:

```
sudo rfkill unblock wifi
```

Then check:

```
rfkill list wifi
```

If you see:

```
Soft blocked: no
Hard blocked: no
```

RFKill is no longer preventing Wi-Fi operation.

You can then investigate the interface with:

```
ip link
```

or:

```
iw dev
```

---

16. Troubleshoot a hard-blocked Wi-Fi device

If:

```
rfkill list wifi
```

shows:

```
Soft blocked: no
Hard blocked: yes
```

try the following:

1. Check the laptop's physical wireless switch.
2. Try the appropriate **Fn + wireless key**.
3. Check airplane-mode settings.
4. Check BIOS/UEFI wireless settings.
5. Reboot the system.
6. Check whether the wireless device is recognized by the kernel.

You can inspect the hardware with:

```
lspci | grep -i wireless
```

For USB Wi-Fi adapters:

```
lsusb
```

---

17. Using rfkill with NetworkManager

On systems using NetworkManager, `rfkill` works at a lower level than NetworkManager.

For example:

```
sudo rfkill block wifi
```

may cause NetworkManager to report that Wi-Fi is unavailable.

You can check NetworkManager's state with:

```
nmcli radio
```

Example:

```
WIFI-HW  WIFI     WWAN-HW  WWAN
enabled  enabled  enabled  enabled
```

You can enable Wi-Fi through NetworkManager with:

```
nmcli radio wifi on
```

However, if RFKill reports:

```
Hard blocked: yes
```

NetworkManager generally cannot override that hardware-level block.

18. Compare rfkill with nmcli

Both commands can affect wireless operation, but they serve different purposes.

Command	Primary purpose
`rfkill`	Control radio blocking
`nmcli`	Configure and manage network connections
`iw`	Configure/manage modern Wi-Fi interfaces
`iwconfig`	Older wireless configuration utility
`ip`	Configure network interfaces and IP networking

For example:

```
sudo rfkill block wifi
```

means:

Disable the Wi-Fi radio.

Whereas:

```
nmcli radio wifi off
```

means:

Tell NetworkManager to turn Wi-Fi off.

---

19. Display rfkill help

```
rfkill help
```

You can also use:

```
rfkill --help
```

For version information:

```
rfkill --version
```

Important rfkill commands

Command	Purpose
`rfkill list`	List all RFKill-controlled devices
`rfkill list wifi`	Show Wi-Fi RFKill status
`rfkill list bluetooth`	Show Bluetooth RFKill status
`sudo rfkill block wifi`	Software-block Wi-Fi
`sudo rfkill unblock wifi`	Software-unblock Wi-Fi
`sudo rfkill block bluetooth`	Software-block Bluetooth
`sudo rfkill unblock bluetooth`	Software-unblock Bluetooth
`sudo rfkill block all`	Block all RFKill devices
`sudo rfkill unblock all`	Unblock all RFKill devices
`rfkill event`	Monitor RFKill events
`rfkill help`	Display command help
`rfkill --version`	Display version information

Summary

`rfkill` is a **radio-control utility**, rather than a complete network-configuration tool. Its primary purpose is to determine whether wireless devices such as **Wi-Fi and Bluetooth are blocked**, and to control software-level blocking.

A useful troubleshooting sequence for Wi-Fi is:

```
rfkill list
ip link
iw dev
nmcli device status
```

If `rfkill` reports:

```
Soft blocked: yes
```

use:

```
sudo rfkill unblock wifi
```

If it reports:

```
Hard blocked: yes
```

investigate the **hardware switch, Fn key, BIOS/UEFI, airplane mode, or firmware** rather than repeatedly issuing `rfkill unblock`.