Tutorial 1 - Peer WiFi

Following tutorial was done with Raspbian 10 Buster and hostapd 2.7

Recommendation:

DO NOT use your main system disk (sdCard), instead, use a fresh system
or some "learn/lab/test" system to makes theses tutorials/experiments.
After completing all set of 3 tutorials you should be able to have
a good basic understanding of the whole matter, and should be able
to know the best way to use the knowledge...

Text Editor:
Use the text editor of your preference, like
Geany (GUI based) or Nano (terminal based), etc.

A1.1.0 WiFi between 2 peers (or more).

A1.1.1 Install hostapd software

```
    sudo apt install hostapd
```

Temporarilly shutdown hostapd service

```
    sudo service hostapd stop
```

Unmask (unlock) hostapd service

```
sudo systemctl unmask hostapd.service
sudo systemctl enable hostapd.service
```

A1.1.2 Create/Modify hostapd configuration file (/etc/hostapd/hostapd.conf)

```
sudo  nano  /etc/hostapd/hostapd.conf
```

Copy and Paste the following text

```
interface=wlan0
driver=nl80211
hw_mode=g
channel=6
wmm_enabled=0
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
ssid=YOUR-SSID
wpa_passphrase=YOUR-PASSPHRASE-PASSWORD
```

A1.1.3 Edit the file /etc/hostapd, modify the variable "DAEMON_CONF".

Find the line with "#DAEMON_CONF=",

Remove any "#" sign at the beginning of the line, if it exist.

add the following value:

```
DAEMON_CONF="/etc/hostapd/hostapd.conf"
```

A1.1.4 Edit the file /etc/dhcpcd.conf, give the WIFI an static IP Address

---

```
    sudo  nano  /etc/dhcpcd.conf
```

Add the following lines at the end of the file.

```
interface wlan0
static ip_address=192.168.50.1/24
nohook wpa_supplicant
```

A1.1.5 Start/Restart hostapd service

```
    sudo service hostapd start
```

A1.1.6 Reboot the system

Reboot the system, so that you can use the installed software.

Congratulations, at this point you have a WIFI ACCESS POINT working.

Be aware that the other peer (guest) need to connect and setup manually the IP Address.

The software hostapd described here, DOES NOT provide IP Addresses to the peers that

will be connecting, we will look at it later in other separated tutorial...

Now you can go ahead and connect another peer(guests) compute to this peer(host)!

At this point, peers can connect with each other and run any kind of software
that uses the network link to exchange data. We have an little example
showing how to exchange files between peers using, for example, webbrowser.

B1.2.0 Add Internet Routing to the peers.

B1.2.1 Edit /etc/sysctl.conf, allow IPv4 FORWARD

Find the line "#net.ipv4.ip_forward=1", then, remove the sharp (#) signal,
at the start of the line. The sharp (#) signal means that this instruction
is not active, the computer does not read lines starting with sharp (#)...

The line then becomes:

```
net.ipv4.ip_forward=1
```

This step will require a reboot.
This step will require a reboot.
The next time the system start it will be able to route traffic (forward)
between peers and the Internet.

B1.2.2 Run the following iptables command line instructions

```
sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
sudo iptables -A FORWARD -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
sudo iptables -A FORWARD -i wlan0 -o eth0 -j ACCEPT
```

Change the "eth0" and "wlan0" and use the names of your actual interfaces.

Use the command "ip a" to display the actual names of your interfaces.

Every time the computer REBOOT, you will need to run these lines!

Congratulations, now you have an equivalent to home router device (part of).

The key point is that you basically did 2 things:

- Created a WIFI ACCESS POINT (also called AP).

- Made the ACCESS POINT share the Internet.

About RaspberryPI Zero with Internet Sharing

RPIZero Internet Sharing was done using an USB-to-Ethernet Adapter,

thus creating eth0 hardware on the RPIZero.

The wlan0 was the onboard WiFi Adapter.

Debian/Ubuntu x86

I did test installing hostapd and net sharing on Debian and Ubuntu in x86 platforms, everything works as described here with very little difference.

I will document these differences so you also can run these steps in a x86 machine as peer(host).