[UPDATED!] USING YOUR NEW RASPBERRY PI 3 AS A WIFI ACCESS POINT WITH HOSTAPD

04 MARCH 2016 on wifi, raspberrypi, hostapd, softap

There's a new Raspberry Pi. This is exciting. It also has on-board WiFi. This makes it doubly exciting!

One of my first thoughts was, can I use it as a SoftAP for some ESP8266 sensor nodes? As it turns out, you can, and it's not that difficult, as the BCM43438 chip is supported by the open-source `brcmfmac` driver!

PACKAGES

The first step is to install the required packages: `sudo apt-get install dnsmasq hostapd`

I'll go into a little detail about the two:

- **hostapd** - This is the package that allows you to use the built in WiFi as an access point
- **dnsmasq** - This is a combined DHCP and DNS server that's very easy to configure

If you want something a little more 'heavyweight', you can use the `isc-dhcp-server` and `bind9` packages for DHCP and DNS respectively, but for our purposes, `dnsmasq` works just fine.

CONFIGURE YOUR INTERFACES

The first thing you'll need to do is to configure your `wlan0` interface with a static IP.

If you're connected to the Pi via WiFi, connect via ethernet/serial/keyboard first.

In newer Raspian versions, interface configuration is handled by `dhcpcd` by default. We need to tell it to ignore `wlan0`, as we will be configuring it with a static IP address elsewhere. So open up the `dhcpcd` configuration file with `sudo nano /etc/dhcpcd.conf` and add the following line to the bottom of the file:

```
denyinterfaces wlan0
```

Note: This must be ABOVE any `interface` lines you may have added!

Now we need to configure our static IP. To do this open up the interface configuration file with `sudo nano /etc/network/interfaces` and edit the `wlan0` section so that it looks like this:

```
allow-hotplug wlan0
iface wlan0 inet static
    address 172.24.1.1
    netmask 255.255.255.0
    network 172.24.1.0
    broadcast 172.24.1.255
#    wpa-conf /etc/wpa_supplicant/wpa_supplicant.conf
```

Restart `dhcpcd` with `sudo service dhcpcd restart` and then reload the configuration for `wlan0` with `sudo ifdown wlan0; sudo ifup wlan0`.

CONFIGURE HOSTAPD

Next, we need to configure `hostapd`. Create a new configuration file with `sudo nano /etc/hostapd/hostapd.conf` with the following contents:

```
# This is the name of the WiFi interface we configured above
interface=wlan0

# Use the nl80211 driver with the brcmfmac driver
driver=nl80211

# This is the name of the network
ssid=Pi3-AP

# Use the 2.4GHz band
hw_mode=g
```

```
# Use channel 6
channel=6

# Enable 802.11n
ieee80211n=1

# Enable WMM
wmm_enabled=1

# Enable 40MHz channels with 20ns guard interval
ht_capab=[HT40][SHORT-GI-20][DSSS_CCK-40]

# Accept all MAC addresses
macaddr_acl=0

# Use WPA authentication
auth_algs=1

# Require clients to know the network name
ignore_broadcast_ssid=0

# Use WPA2
wpa=2

# Use a pre-shared key
wpa_key_mgmt=WPA-PSK

# The network passphrase
wpa_passphrase=raspberry

# Use AES, instead of TKIP
rsn_pairwise=CCMP
```

We can check if it's working at this stage by running `sudo /usr/sbin/hostapd /etc/hostapd/hostapd.conf`. If it's all gone well thus far, you should be able to see to the network **Pi3-AP**! If you try connecting to it, you will see some output from the Pi, but you won't receive and IP address until we set up dnsmasq in the next step. Use **Ctrl+C** to stop it.

We aren't quite done yet, because we also need to tell hostapd where to look for the config file when it starts up on boot. Open up the default configuration file with `sudo nano /etc/default/hostapd` and find the line `#DAEMON_CONF=""` and replace it with `DAEMON_CONF="/etc/hostapd/hostapd.conf"`.

CONFIGURE DNSMASQ

The shipped `dnsmasq` config file contains a wealth of information on how to use it, but the majority of it is largely redundant for our purposes. I'd advise moving it (rather than deleting it), and creating a new one with

```
sudo mv /etc/dnsmasq.conf /etc/dnsmasq.conf.orig
sudo nano /etc/dnsmasq.conf
```

Paste the following into the new file:

```
interface=wlan0      # Use interface wlan0
listen-address=172.24.1.1 # Explicitly specify the address to listen on
bind-interfaces      # Bind to the interface to make sure we aren't sending
things elsewhere
server=8.8.8.8       # Forward DNS requests to Google DNS
domain-needed        # Don't forward short names
bogus-priv           # Never forward addresses in the non-routed address
spaces.
dhcp-range=172.24.1.50,172.24.1.150,12h # Assign IP addresses between
172.24.1.50 and 172.24.1.150 with a 12 hour lease time
```

SET UP IPV4 FORWARDING

One of the last things that we need to do before we send traffic anywhere is to enable packet forwarding.

To do this, open up the `sysctl.conf` file with `sudo nano /etc/sysctl.conf`, and remove the `#` from the beginning of the line containing `net.ipv4.ip_forward=1`. This will enable it on the next reboot, but because we are impatient, activate it immediately with :
`sudo sh -c "echo 1 > /proc/sys/net/ipv4/ip_forward"`

We also need to share our Pi's internet connection to our devices connected over WiFi by the configuring a NAT between our `wlan0` interface and our `eth0` interface. We can do this using the following commands:

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

However, we need these rules to be applied every time we reboot the Pi, so run `sudo sh -c "iptables-save > /etc/iptables.ipv4.nat"` to save the rules to the file `/etc/iptables.ipv4.nat`. Now we need to run this after each reboot, so open the `rc.local` file with `sudo nano /etc/rc.local` and just above the line `exit 0`, add the following line:

```
iptables-restore < /etc/iptables.ipv4.nat
```

WE'RE ALMOST THERE!

Now we just need to start our services:

```
sudo service hostapd start
sudo service dnsmasq start
```

And that's it! You should now be able to connect to the internet through your Pi, via the on-board WiFi!

To double check we have got everything configured correctly, reboot with `sudo`

`reboot`.

EDIT: *Thanks to Justin for helping iron out some of the errors in this post!*

EDIT2: *Thanks to Ashok for several performance related enhancements!*

EDIT3: *Thanks to Lasse for some amendments to the dnsmasq configuration!*

EDIT4: *Fixed race condition between dhcpcd and dnsmasq, wlano is no longer configured by dhcpcd.*

PHIL MARTIN

Read more posts by this author.

READ THIS NEXT

RASPBERRY PI 3 UART BAUD RATE WORKAROUND

If you've tried to use the UART on the GPIO header of the new Raspberry Pi 3, you may...

YOU MIGHT ENJOY

GPS CLOCK

Due to my workplace's lack of enthusiasm for precision time keeping (and also the fact that the clock is...