Part 28
-
Raspberry Pi as a Wireless Access Point

Version: 2018-03-18

Raspberry Pi as a wireless access point

The Raspberry Pi can do a lot, especially now that the new Raspberry Pi comes with wireless capabilities already on board. It can take the place of a ton of different (and more expensive) devices – including a router. If you turn your Raspberry Pi into a wireless access point, you can make it act as a router. It's not the most powerful thing in the world, but it does work, and the project is a lot of fun.

All we're really doing is installing a couple packages that give the Pi the ability to do router-like things like assign IP addresses to devices that connect to it.

Step 1: Update Raspbian

```
sudo apt-get update
sudo apt-get upgrade
```

If you get an upgrade, It's a good idea to reboot with **sudo reboot**.

Step 2: Install hostapd and dnsmasq
These are the two programs we're going to use to make your Raspberry Pi into a wireless access point. To get them, just type these lines into the terminal:

```
sudo apt-get install hostapd
sudo apt-get install dnsmasq
```

Both times, you'll have to hit y to continue. `hostapd` is the package that lets us create a wireless hotspot using a Raspberry Pi, and `dnsmasq` is an easy-to-use DHCP and DNS server. We're going to edit the programs' configuration files in a moment, so let's turn the programs off before we start tinkering:

```
sudo systemctl stop hostapd
sudo systemctl stop dnsmasq
```

Step 3: Configure a static IP for the wlan0 interface
For our purposes here, I'm assuming that we're using the standard home network IP addresses, like 192.168.###.###. Given that assumption, let's assign the IP address **192.168.0.10** to the wlan0 interface by editing the dhcpcd configuration file. Start editing with this command:

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

Now that you're in the file, add the following lines at the end:

```
interface wlan0
static ip_address=192.168.0.10/24
denyinterfaces eth0
denyinterfaces wlan0
```

The last two lines are needed in order to make our bridge work –- but more on that in Step 8. After that, press **Ctrl+X**, then **Y**, then **Enter** to save the file and exit the editor.

Step 4: Configure the DHCP server
We're going to use `dnsmasq` as our DHCP server. The idea of a DHCP server is to dynamically distribute network configuration parameters, such as IP addresses, for interfaces and services. `dnsmasq`'s default configuration file contains a lot of unnecessary information, so it's easier for us to start from scratch. Let's rename the default configuration file and write a new one:

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

You'll be editing a new file now, and with the old one renamed, this is the config file that dnsmasq will use. Type these lines into your new configuration file:

```
interface=wlan0
  dhcp-range=192.168.0.11,192.168.0.30,255.255.255.0,24h
```

The lines we added mean that we're going to provide IP addresses between 192.168.0.11 and 192.168.0.30 for the wlan0 interface.

Step 5: Configure the access point host software
Another config file! This time, we're messing with the hostapd config file. Open 'er up:

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

This should create a brand new file. Type in this:

```
interface=wlan0
bridge=br0
hw_mode=g
channel=7
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=NETWORK
wpa_passphrase=PASSWORD
```

Note that where I have "NETWORK" and "PASSWORD," you should come up with your own names. This is how you'll join the Pi's network from other devices. We still have to show the system the location of the configuration file:

```
sudo nano /etc/default/hostapd
```

In this file, track down the line that says #DAEMON_CONF="" – delete that **#** and put the path to our config file in the quotes, so that it looks like this:

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

The # keeps the line from being read as code, so you're basically bringing this line to life here while giving it the right path to our config file.

Step 6: Set up traffic forwarding
The idea here is that when you connect to your Pi, it will forward the traffic over your Ethernet cable. So we're going to have wlan0 forward via Ethernet cable to your modem. This involves editing yet another config file:

```
sudo nano /etc/sysctl.conf
```

Now find this line:

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

…and delete the "#" – leaving the rest, so it just reads:

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

Step 7: Add a new iptables rule
Next, we're going to add IP masquerading for outbound traffic on eth0 using iptables:

```
sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

…and save the new iptables rule:

```
sudo sh -c "iptables-save > /etc/iptables.ipv4.nat"
```

To load the rule on boot, we need to edit the file /etc/rc.local and add the following line just above the line exit 0:

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

Step 8: Enable internet connection
Now the Raspberry Pi is acting as an access point to which other devices can connect. However, those devices can't use the Pi to access the internet just yet. To make the possible, we need to build a bridge that will pass all traffic between the wlan0 and eth0 interfaces. To build the bridge, let's install one more package:

```
sudo apt-get install bridge-utils
```

We're ready to add a new bridge (called br0):

```
sudo brctl addbr br0
```

Next, we'll connect the eth0 interface to our bridge:

```
sudo brctl addif br0 eth0
```

Finally, let's edit the interfaces file:

```
sudo nano /etc/network/interfaces
```

…and add the following lines at the end of the file:

```
auto br0
iface br0 inet manual
bridge_ports eth0 wlan0
```

Step 9: Reboot
Now that we're ready, let's reboot with

```
sudo reboot
```

Now your Pi should be working as a wireless access point. Try it out by hopping on another device and looking for the network name you used back in step 5.

Make an Access Point of your Pi

1. Prepare the raspberry pi

   We need a lot of software to get the wireless access point working. First, make sure your Raspberry Pi is up to date

   ```
   sudo apt-get update
   sudo apt-get upgrade
   ```

   Then install

   ```
   sudo apt-get install hostapd bridge-utils
   ```

   You'll need to turn off some of the new services you've just installed

   ```
   sudo systemctl stop hostapd
   ```

2. Bridge the devices

   This works best with an Ethernet connection, but you should be able to use a wireless connection – with some caveats. We need to bridge the connection between the internet (eth0) and the wireless network (wlan0). Edit the DHCP file by first using

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

   and add

   ```
   denyinterfaces wlan0
   ```

   and

   ```
   denyinterfaces eth0
   ```

   to the bottom of the file. Save and exit, and then create a bridge with

   ```
   sudo brctl addbr br0
   ```

   Connect the network ports with

   ```
   sudo brctl addif br0 eth0 wlan0
   ```

3. Bridge information

   Open the network interfaces file with

   ```
   sudo nano /etc/network/interfaces
   ```

   then change the wlan info to manual.

   ```
   allow-hotplug wlan0
   iface wlan0 inet manual
   ```

   With that done, add the information for the bridge to the file.

   ```
   # Bridge setup
   auto br0
   iface br0 inet dhcp
   bridge_ports eth0 wlan0
   ```

4. Create the access point

   Now we need to edit the hostapd file to allow another computer to connect to the Raspberry Pi. Open the file with

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

   and add the following, putting in your own network name and password

   ```
   interface=wlan0
   bridge=br0
   ssid=[Network name]
   hw_mode=g
   channel=7
   wmm_enabled=0
   macaddr_acl=0
   auth_algs=1
   ignore_broadcast_ssid=0
   wpa=2
   wpa_passphrase=[Password]
   wpa_key_mgmt=WPA-PSK
   wpa_pairwise=TKIP
   rsn_pairwise=CCMP
   ```

5. Final tweaks

   Save the conf file and open another file with

   ```
   sudo nano /etc/default/hostapd
   ```

   In this file we can tell the system where to find the configuration file we edited. Find the line

   ```
   #DAEMON_CONF
   ```

   and replace it with

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

   Save, exit, and then reboot the Raspberry Pi.
   You should now be able to connect to it from another device.