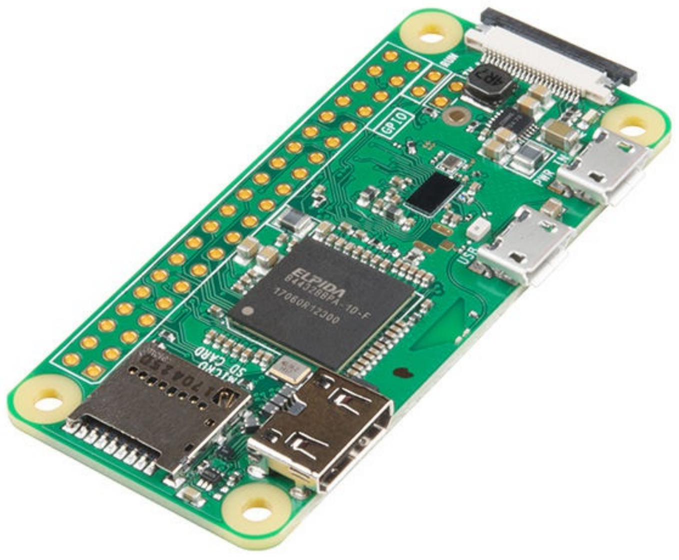
instructables

Raspberry Pi Enterprise Network WiFi Bridge

By: Riley Barrett and Dylan Halland

The goal of this project is to allow an IoT device, such as a Weemo Smart Plug, Amazon Echo, Gaming Console, or any other Wi-Fi enabled device to connect to a WPA_EAP Enterprise Network by using a Raspberry Pi Zero W as a packet forwarding device. Extra configuration steps are required for devices connecting to an enterprise network, and many devices are not compatible at all. By using a Wi-Fi Bridge, any device can easily obtain internet access by connecting to the Pi.

The system can be implemented on either one wireless card or two separate cards depending on the requirements of the user. For systems requiring higher signal strength and faster upload/download speeds, it is best to use a dedicated wireless card to host the access point. However, for systems where the signal strength and bandwidth are less important, or where a more cost effective solution is desired, a single card can be shared by the access point and network connection.



Step 1: Setting Up the Raspberry Pi

Begin by connecting your Pi to a keyboard and monitor (may require an HDMI adapter).

Then, you can begin by typing the command:

```
sudo su
```

This will ensure you have the necessary privileges to modify files on the pi.

Now you will want to install dnsmasq and hostapd using the command:

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

You can now begin to setup the WiFi bridge.

NOTE - The following tutorial will contain information for those using the single on-board wireless card for both the access point and for connecting to the network. It is also possible to configure the system to run on two separate cards. To do this, simply look for the commented out "wlan1" lines in the provided files, and substitute them for the neighboring "ap0" lines.

Step 2: 70-persistent-net.rules

Begin by finding the MAC address of your pi by typing:

```
iw dev
```

Create the following file:

```
nano /etc/udev/rules.d/70-persistent-net.rules
```

and edit it so that it contains the following

```
SUBSYSTEM=="ieee80211", ACTION=="add|change", ATTR{macaddress}=="b8:27:eb:c0:38:40", KERNEL=="phy0", \<br>  RUN+="/sbin/iw phy phy0 interface add ap0 t
ype __ap", \
  RUN+="/bin/ip link set ap0 address b8:27:eb:c0:38:40"
```

This file tells the system to allocate a device for the access point at boot. Here, the MAC address should be replaced with that of your own pi, which you just found.

(Two Wireless Cards) This file is not required when using two wireless cards.

Step 3: Hostapd.conf

Next, you will then edit the hostapd.conf file by entering the following:

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

Modify the file so that it matches the following configuration:

```
<p>ctrl_interface=/var/run/hostapd</p><p>ctrl_interface_group=0</p><p>#interface=ap0</p><p>interface=wlan1</p><p>driver=nl80211</p><p>ssid=testnet</p><p>hw_
mode=g</p><p>channel=6</p><p>wmm_enabled=0</p><p>macaddr_acl=0</p><p>auth_algs=1</p><p>wpa=2</p><p>wpa_passphrase=0123456789</p><p>wpa_key_
mgmt=WPA-PSK</p><p>wpa_pairwise=TKIP CCMP</p><p>rsn_pairwise=CCMP</p>
```

Note that while my channel here is set to 6, you may need to change this value to match the channel that wlan0 is on. On some networks, the channel will automatically be changed for the access point to match wlan0, but this was not my experience on the enterprise network. You can check which channels are currently in use and by which interfaces by typing

```
iwlist channel
```

(Two Wireless Cards) Simply uncomment the line containing wlan1 and comment out the one containing ap0.

Step 4: Dnsmasq.conf

Now you will edit the dnsmasq.conf file:

```
nano /etc/dnsmasq.conf
```

Uncomment or add the following lines:

```
interface=lo,ap0
#interface=lo,wlan1
no-dhcp-interface=lo
bind-interfaces
server=8.8.8.8
domain-needed
bogus-priv
dhcp-range=192.168.2.100,192.168.2.200,12h
```

You can use your own subnet here if you would like, just be sure that you are consistent.

(Two WirelessCcards) Uncomment the line containing wlan1, and comment out the one containing ap0.

Step 5: Interfaces

Next, you will need to modify the interfaces file:

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

```
auto lo
auto ap0
#auto wlan1
auto wlan0
iface lo inet loopback
iface eth0 inet dhcp
allow-hotplug ap0
#allow-hotplug wlan1
iface ap0 inet static
#iface wlan1 inet static
address 192.168.2.1
 netmask 255.255.255.0
 hostapd /etc/hostapd/hostapd.conf

allow-hotplug wlan0
iface wlan0 inet dhcp
 pre-up wpa_supplicant -B -Dwext -i wlan0 -c/etc/wpa_supplicant/wpa_supplicant.conf
 post-down killall -q wpa_supplicant
```

It is worth noting that the wlan0 interface MUST come after whatever interface you are forwarding to it, otherwise the system will not work properly.

(Two Wireless Cards) Uncomment any lines containing wlan1 and comment out any containing ap0.

Step 6: Wpa_supplicant.conf

Now you will modify the wpa_supplicant.conf file found at:

```
nano /etc/wpa_supplicant/wpa_supplicant.conf
```

Some networks are configured differently than others so this part may require some tinkering, here is the wpa_supplicant.conf file that allowed me to connect to the network at Cal Poly:

```
country=US<br>ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
network={
 ssid="SecureMustangWireless"
 scan_ssid=1
 key_mgmt=WPA-EAP
 pairwise=CCMP TKIP
 group=CCMP TKIP
 eap=PEAP
 identity="username@calpoly.edu"
 password="your_password"
 phase1="peapver=0"
 phase2="MSCHAPV2"
}
```

This file is used to configure wlan0 to connect to your enterprise network. Some enterprise networks require a CA Certificate in order to connect. Cal Poly's campus network does not require a certificate, so I have skipped this portion, but you can easily download the proper certificates and add them to your wpa_supplicant file with the line

```
<p>ca_cert="/path/to/cert.pem"
```

Step 7: Hostapdstart Script

The last thing left to do is write a script that starts both interfaces and sets up the packet forwarding when the system boots. Creat a file called hostapdstart by typing:

```
nano /usr/local/bin/hostapdstart
```

Add the following to the file

```
<p>sudo ifdown --force wlan0 && sudo ifdown --force ap0 && sudo ifup ap0 && sudo ifup wlan0</p><p>#sudo ifdown --force wlan0 && sudo ifdown --force wlan1 && sudo i
fup wlan1 && sudo ifup wlan0</p><p>sudo sysctl -w net.ipv4.ip_forward=1</p><p>sudo iptables -t nat -A POSTROUTING -s 192.168.2.0/24 ! -d 192.168.2.0/24 -j MASQ
UERADE</p><p>sudo systemctl restart dnsmasq</p>
```

This script brings down both interfaces, then brings them back up in the proper order, tells the pi that you would like to forward packets from one interface to another, and finally restarts dnsmasq so that the changes will take affect.

(Two Wireless Cards) uncomment line with wlan1 and comment out line with ap0.

Step 8: Rc.local

Finally, we want the system to start when the system boots, so we will modify the rc.local file, which is run at boot:

```
nano /etc/rc.local
```

Simply add the following line to the end of the file:

```
hostapdstart>1&
```

Your file should look something like this:

```
 _IP=$(hostname -I) || true
 if [ "$_IP" ]; then
  printf "My IP address is %s\n" "$_IP"
fi

 hostapdstart>1&

 exit 0
```

Step 9: Reboot

And that's it! Now, assuming you have everything setup correctly, and your WiFi dongle is attached (if you are using one), you simply need to reboot your Raspberry Pi with the command:

reboot

Once your Pi has successfully rebooted, you should be able to see the name of your Access Point on any device (phone, laptop, etc.). Once you connect using your specified password, it should connect you directly to your desired Enterprise network!

Special thanks to the following links for providing us with an idea of how to approach this project:

- https://blog.thewalr.us/2017/09/26/raspberry-pi-ze...
- https://www.raspberrypi.org/forums/viewtopic.php?p...
- https://www.raspberrypi.org/forums/viewtopic.php?f...

Let us know if you have any questions, comments, or suggestions!