# Wireless Access Point With a Raspberry Pi 3

## Introduction

This HOWTO shows you how to replace the WiFi element of your home network setup with a Raspberry Pi running Slackware. Other HOWTOs explain how to setup DHCP and so on for your wireless devices, but this tries to keep things really simple and just give you an ethernet→Wifi bridge. This is very close to the config I ended up with when I configured a TP-LINK ADSL router to do the same thing, only this way I get to patch stuff a lot more easily.

Although I've originally written this HOWTO for a RPi 3, I've since tried it on an RPi2 and it works just as well. Interface wlan1 will obviously change to wlan0 because the Pi2 has no on-board WiFi however.

## Hardware Requirements

- A Raspberry Pi 3 running Slackware 14.2
- A powered USB hub with 2A Power supply (to ensure the Wifi adapter gets enough juice)
- Alfa Network AWUS036NHA - USB WiFi Adapter with chipset ar9271l
- Ethernet cable

The powered Hub was purchased from the_pi_hut (http://www.ebaystores.co.uk/thepihut), although pretty much any decent powered hub should be OK. The pi_hut one claimed to have the feature that it didn't push power up the USB connection to the Pi, which sounded like it may be useful.


Whatever you get, Make sure it has a beefy power supply, e.g. 2A. USB WiFi dongles, particularly the ones with external antennas can be thirsty and a 2A power adapter is unlikely to cost much more than a 1A one.


The Wifi Adapter choice is important. I did some research and went for the model that I believed would give maximum flexibility and compatibility with Linux. The Alpha Network one seemed to do that. It works with Kali (assuring Linux compatibility), but that is by no means the only thing you should look for. You also want to see the output of:

```
# iw list
```

Which for this model gives:

```
Supported interface modes:
         * IBSS
         * managed
         * AP
         * AP/VLAN
         * monitor
         * mesh point
         * P2P-client
         * P2P-GO
         * outside context of a BSS
```

Obviously the 'AP' bit is the bit we need, but IMHO the more things appearing in this list, the more likely your adapter is to be well tested and widely used.

My adapter was purchased from Amazon: https://www.amazon.co.uk/dp/B004Y6MIXS, however I am not connected with this seller. The adapter looks like this:


## Adapter detection


I advise to not connect and disconnect the hub from the Pi while it's running. Instead, power it up with hub connected (and powered) and then connect and disconnect the Wifi adapter as needed.

Ensure that iwconfig shows two adapters wlan0 and wlan1. wlan0 is the built-in one on the Pi. If dmesg isn't showing the Alpha one has been plugged in, then just un-plug it and plug it in again. Bring the Adapter up with:

```
# ifconfig wlan1 up
```

This should result in the blue LED coming on.

```
# ifconfig wlan1 down
```

This should result in the blue LED going off again. If this happens, it seems your adapter is being recognised.

While you're here, have a quick scan to see which channels might be in use:

```
# iwlist wlan1 scanning | grep Channel
```

This will indicate the channels that you **don't** want to use. You can configure automatic channel selection but this guide will not cover that.

# Bridge Configuration

We will configure the AP to get it's own IP address on the ethernet side. Edit /etc/rc.d/rc.inet1.conf. Comment out the section with your eth0 network config, it probably looks a bit like this (it may use a static IP address instead):

```
# Config information for eth0:
IPADDR[0]=""
NETMASK[0]=""
USE_DHCP[0]="yes"
DHCP_HOSTNAME[0]="myrpi"
```

(comment out the four lines).

Now replace that lot by commenting **in** the section starting:

```
# Example of how to configure a bridge:
```

Ensure the following appears so a bridge will be created (adjust DHCP_HOSTNAME to suit):

```
IFNAME[0]="br0"
BRNICS[0]="eth0"
IPADDR[0]=""
NETMASK[0]=""
USE_DHCP[0]="yes"
DHCP_HOSTNAME[0]="wap"
```

This sets up a bridge, with eth0 attached to it, and tells Slackware to give the new br0 interface an IP address via DHCP. Reboot after you've done that and check that your ethernet networking is still in working order.

# Hostapd Install

The only AP software for Linux that I'm aware of is hostapd. Get it from Slackbuilds and build it directly on your Pi (you did install the 'D' disk set didn't you?)

```
# wget https://slackbuilds.org/slackbuilds/14.2/network/hostapd.tar.gz
# tar xvf hostapd.tar.gz
# cd hostapd
# wget https://w1.fi/releases/hostapd-2.6.tar.gz
# ./hostapd.SlackBuild
```

```
# installpkg /tmp/hostapd-2.6-arm-2_SBo.tgz
```

Next you need to edit /etc/hostapd/hostapd.conf to set stuff up. Backup the contents of hostapd.conf to another file as it contains useful comments, but replace the contents completely so it contains only this:

```
interface=wlan1
bridge=br0
logger_syslog=-1
logger_syslog_level=2
ctrl_interface=/var/run/hostapd
ssid=biffsnetwork
wpa=2
wpa_passphrase=somethingreallysecret!
hw_mode=g
channel=7
auth_algs=1
```

What this does: The interface wlan1, we've seen before. The bridge br0 will get wlan1 interface added to it when hostapd has fired up. The syslog logging lines ensure you get something in /var/log/messages to tell you what's going on. The ssid, wpa and wpa_passphrase settings should be rather obvious, the hw_mode you can just leave as 'g' unless you know what you're doing, channel is set to something you didn't see when you ran `iwlist wlan1 scanning` earlier on.

That minimal config should get hostapd working fine, at least it did for me.

Finally, you just need to start hostapd, either by adding /etc/rc.d/rc.hostapd to /etc/rc.d/rc.local or adding an extra entry to /etc/rc.d/rc.inet2 (my preference)

```
if [ -x /etc/rc.d/rc.hostapd ]; then
  /etc/rc.d/rc.hostapd start
fi
```

Don't forget to:

```
# chmod 755 /etc/rc.d/rc.hostapd
```

# QR Codes

No home WiFi network setup is complete without a QR code that you can wave at all your mates when they are round your house wanting to connect. You can create one of these from your WiFi settings using the **qrencode** program, available from Slackbuilds, or alternatively, get the build from Alien

Bob for a slightly later version.

Type the following to create a PNG file with the code, which you can print, copy to your mobile, put on a local webpage or whatever.

```
# qrencode -t PNG -o wifi-setup.png
"WIFI:S:MYSSID;T:WPA2;P:MYPASSWD;H:true;"
```

The above assumes WPA2, MYSSID should be replaced with your SSID, MYPASSWD replaced with your password. For my Android phone I installed a piece of software from the Play store called Barcode Scanner by the ZXing Team. It works very well for reading these codes.

At a pinch, the following bash script should pull your SSID and passphrase out of the hostapd.conf file, and automatically use them to create a new QR code, so you can quickly keep it up to date after any changes.

```
#!/bin/sh

SSID=$(cat /etc/hostapd/hostapd.conf | grep ^ssid= | cut -d= -f2)
PASS=$(cat /etc/hostapd/hostapd.conf | grep ^wpa_passphrase= | cut -d= -f2)

qrencode -t PNG -o hostapd.png "WIFI:S:$SSID;T:WPA2;P:$PASS;H:true;"
```

## Conclusion

That's it! You should now be able connect to this AP, and (assuming you are running a DHCP server on your ethernet) it will serve your wireless devices, and everything will be able to talk to everything else. Your desktop will be able to ping your mobile and so on.

# Sources

- Originally written by User bifferos

howtos, ethernet, wifi, wap, raspberry, arm, author bifferos