adafruit learning system

Setting up a Raspberry Pi as a WiFi access point

Created by lady ada



Last updated on 2015-03-10 04:30:11 PM EDT

Guide Contents

Guide Contents 2
Overview 3
What you'll need 5
Preparation 6
Check Ethernet & Wifi 8
Install software 10
Set up DHCP server 10
Set up wlan0 for static IP 14
Configure Access Point 16
Configure Network Address Translation 18
Update hostapd 20
First test! 22
Finishing up! 24
Extra: Removing WPA-Supplicant 24
Connect and Test 25
More! 29
Compiling hostapd 30

Overview



Would you like to use your Pi as a WiFi router? Or maybe have it as a special filtering access point? Setting up a Pi as an access point (AP) is a bit more advanced than using it as a client, but its still only a half hour of typing to configure. If you want to, this tutorial will make it so the Pi broadcasts a WiFi service and then routes internet traffic to an Ethernet cable. Since its all Linux you can go in and update or configure it however you like.

I used the following pages as a guide to create this tutorial, **please note** many of them will not work completely, but check them out if you are interested!

- http://qcktech.blogspot.com/2012/08/raspberry-pi-as-router.html (http://adafru.it/cfU)
- http://itsacleanmachine.blogspot.com/2013/02/wifi-access-point-with-raspberry-pi.html (http://adafru.it/cfV)
- http://esrlabs.com/android-transporter-for-the-nexus-7-and-the-raspberry-pi/ (http://adafru.it/cfW)
- http://elinux.org/RPI-Wireless-Hotspot (http://adafru.it/cfX)

Currently tested working on Raspbian only

What you'll need

You'll need a few things to run this tutorial:

- Raspberry Pi model B+ (http://adafru.it/1914) (or B)- Ethernet is required
- Ethernet cable (http://adafru.it/730)
- WiFi adapter (http://adafru.it/814) **- Not all WiFi adapters work, we know for sure it works with the ones in the Adafruit shop!**
- SD Card (4GB or greater) with Raspbian on it. You can either DIY it or buy a ready-made Raspbian card (http://adafru.it/1121)
- Power supply for your Pi & a Micro USB cable
- USB Console cable (optional) - this makes it a little easier to debug the system (http://adafru.it/954)
- Case for your Pi (optional) (http://adafru.it/2258)
- A SD or MicroSD card reader (http://adafru.it/939) (optional)

Our Pi B+ starter pack (http://adafru.it/2125) will be all you need and even comes with more fun stuff you can play with

Preparation

This tutorial assumes you have your Pi mostly set up and ready to go.

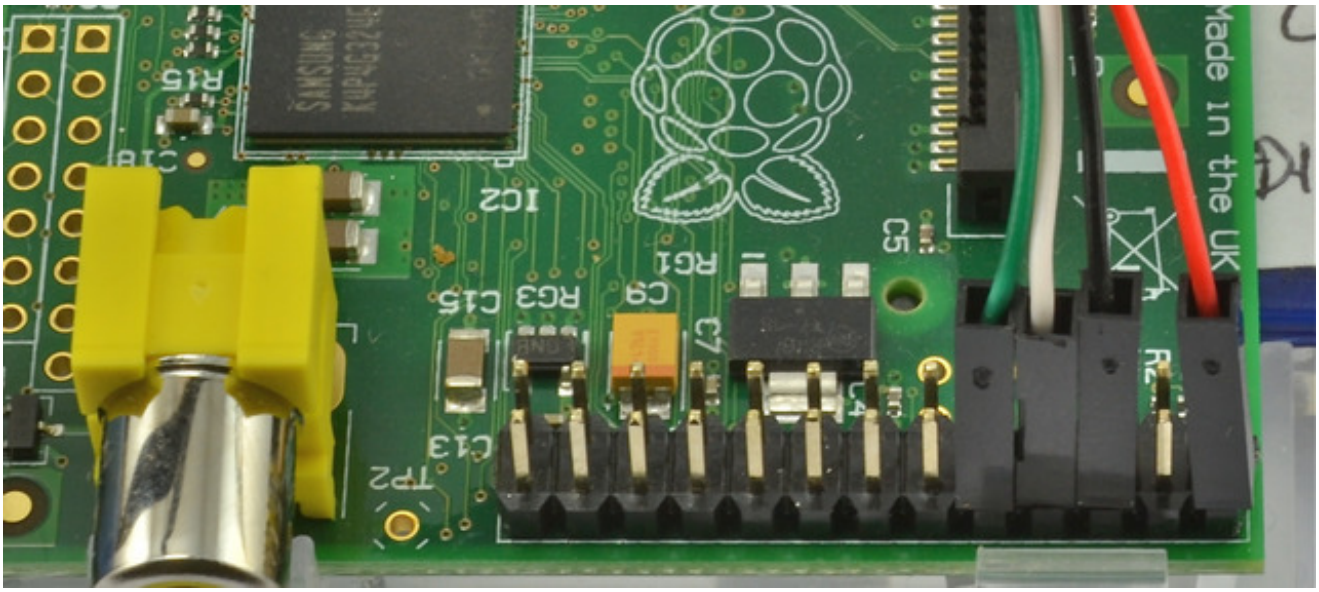
Please follow the tutorials in order to

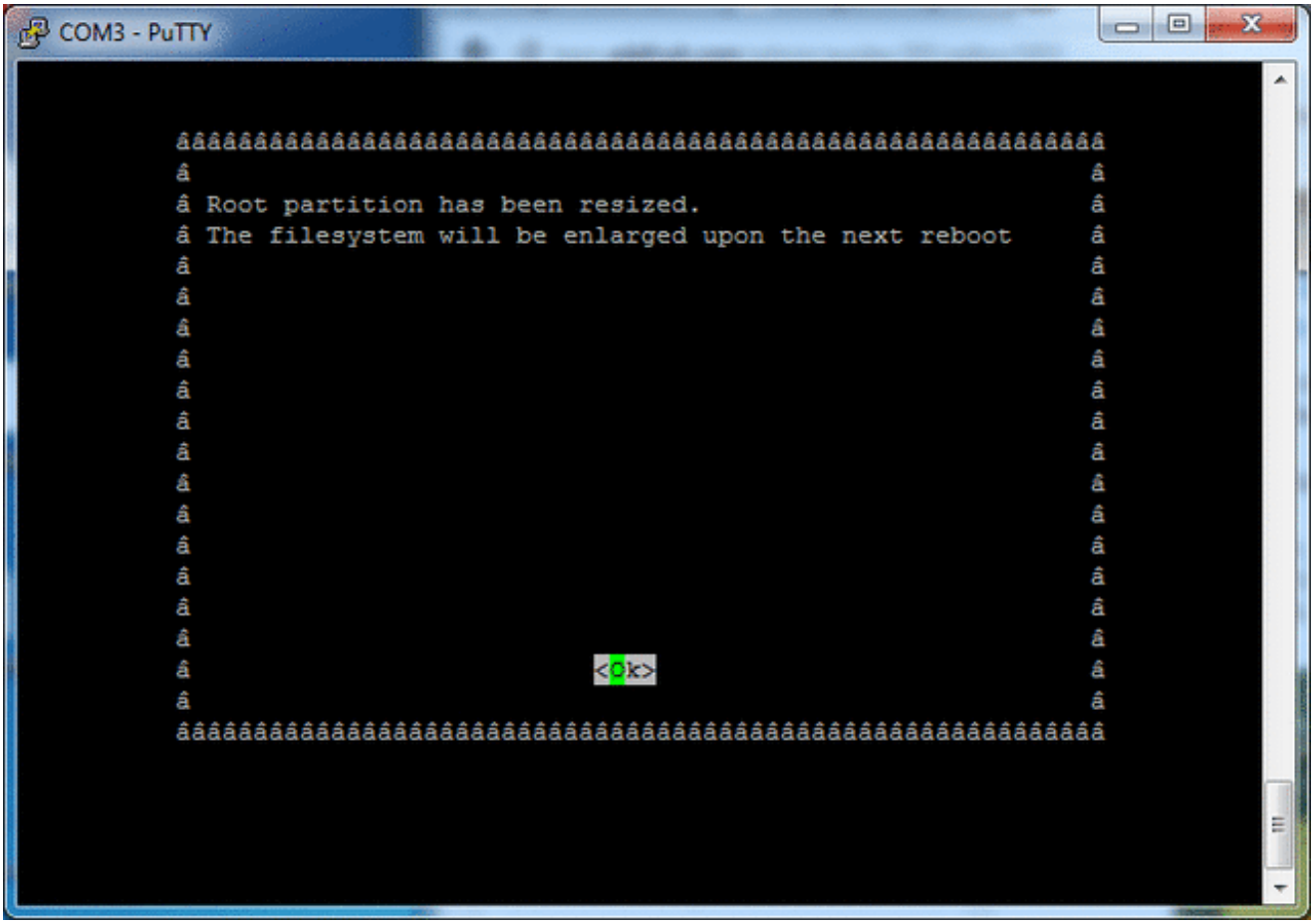
1. Install the OS onto your SD card (http://adafru.it/aWq)
2. Boot the Pi and configure (http://adafru.it/aUa)
   Don't forget to change the default password for the 'pi' acccount!
3. Set up and test the Ethernet and Wifi connection (http://adafru.it/aUB)
4. Connect with a USB console cable (optional)
   (http://adafru.it/aUA)

When done you should have a Pi that is booting Raspbian, you can connect to with a USB console cable and log into the Pi via the command line interface.

It is possible to do this tutorial via **ssh** on the Ethernet port **or** using a console cable.

If using a console cable, even though the diagram on the last step shows powering the Pi via the USB console cable (red wire) we suggest not connecting the red wire and instead powering from the wall adapter. Keep the black, white and green cables connected as is.

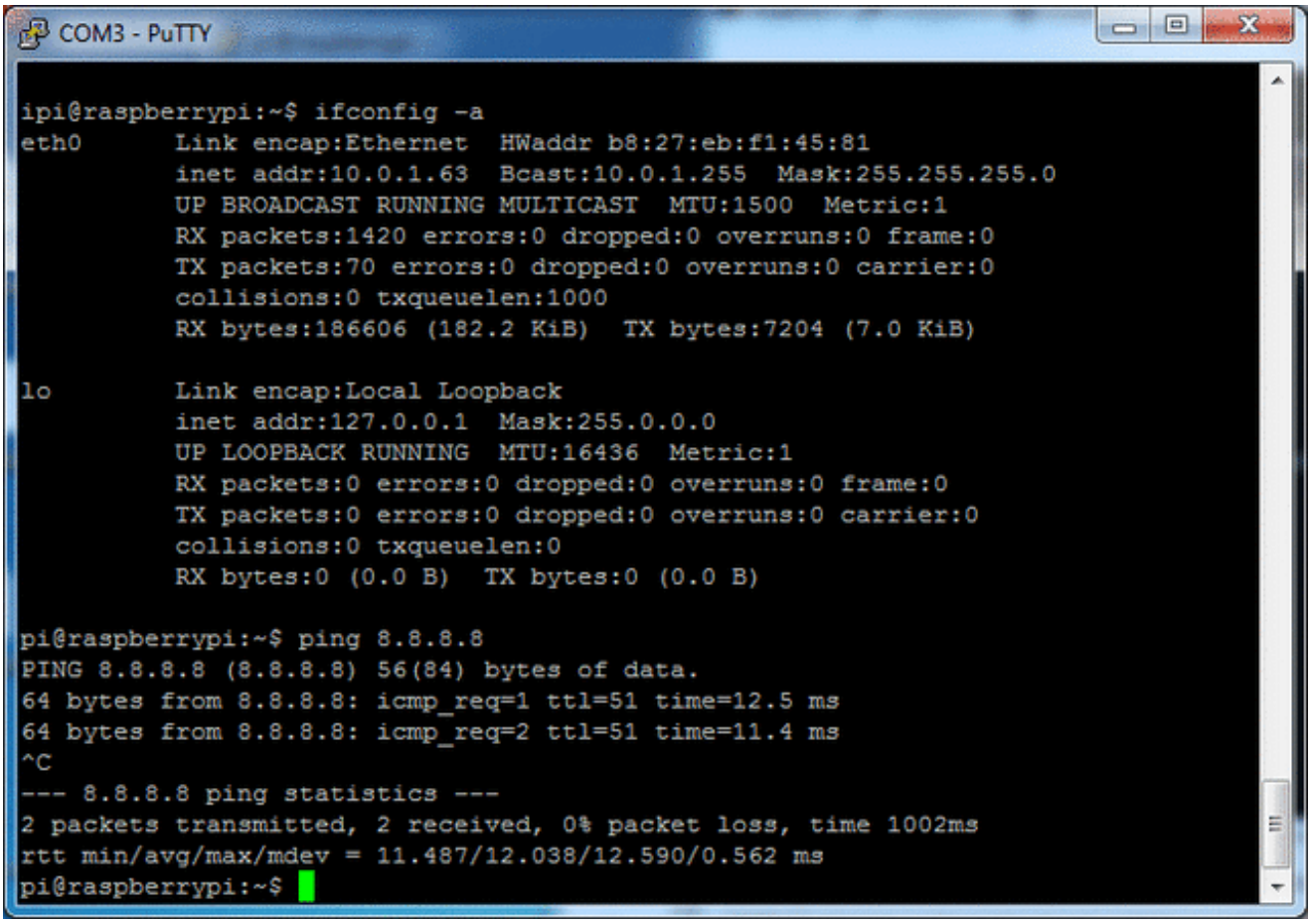




Don't forget to expand the SD card, or you may run out of space!

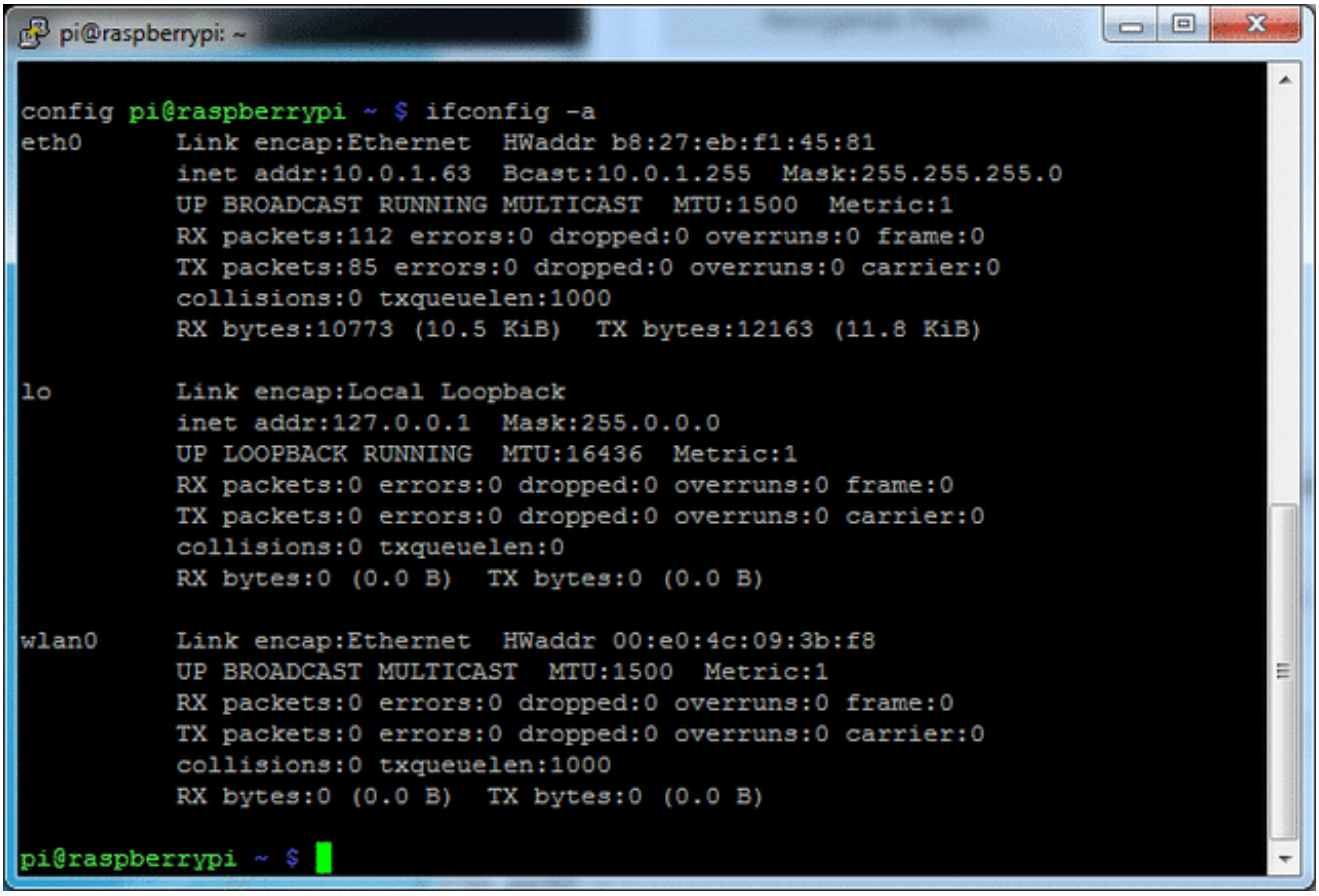
Check Ethernet & Wifi

Before continuing make sure the Ethernet cable is connected in and you can **ping** out from the Pi



You will also want to set up your WiFi dongle. run **sudo shutdown -h now** and then plug in the WiFi module when the Pi is off so you don't cause a power surge.

When it comes back up check with **ifconfig -a** that you see **wlan0** - the WiFi module.

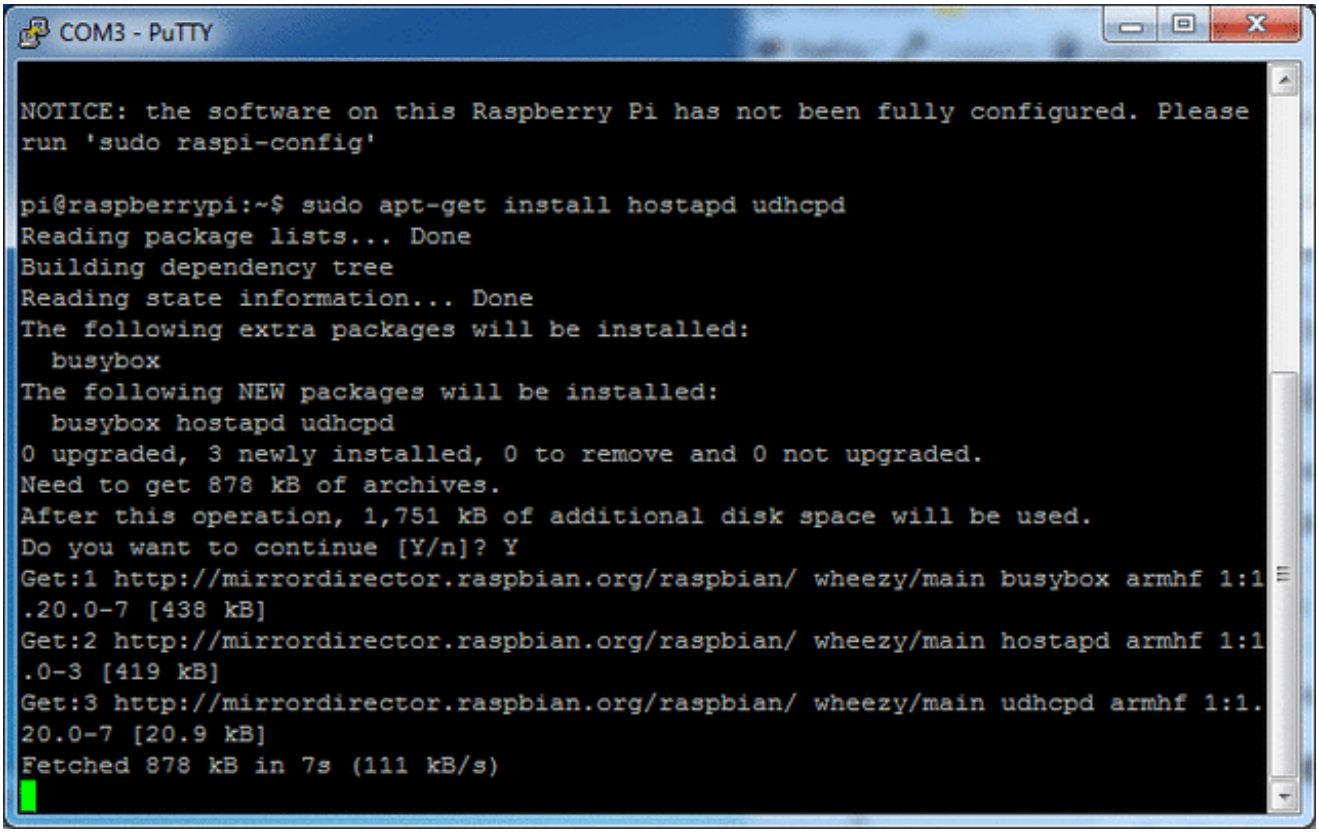

pi@raspberrypi: ~
config pi@raspberrypi ~ $ ifconfig -a
eth0 Link encap:Ethernet HWaddr b8:27:eb:f1:45:81
inet addr:10.0.1.63 Bcast:10.0.1.255 Mask:255.255.255.0
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:112 errors:0 dropped:0 overruns:0 frame:0
TX packets:85 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:10773 (10.5 KiB) TX bytes:12163 (11.8 KiB)
lo Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
UP LOOPBACK RUNNING MTU:16436 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)
wlan0 Link encap:Ethernet HWaddr 00:e0:4c:09:3b:f8
UP BROADCAST MULTICAST MTU:1500 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)
pi@raspberrypi ~ $

Install software

Next up we install the software onto the Pi that will act as the 'hostap' (host access point) **You need internet access for this step so make sure that Ethernet connection is up!**

```
sudo apt-get update
sudo apt-get install hostapd isc-dhcp-server
```

(You may need to **sudo apt-get update** if the Pi can't seem to get to the apt-get repositories)



(text above shows udhcpd but that doesnt work as well as isc-dhcp-server, still, the output should look similar)

Set up DHCP server

Next we will edit /etc/dhcp/dhcpd.conf, a file that sets up our DHCP server - this allows wifi connections to automatically get IP addresses, DNS, etc.

Run this command to edit the file

> **sudo nano /etc/dhcp/dhcpd.conf**

Find the lines that say

```
option domain-name "example.org";
option domain-name-servers ns1.example.org, ns2.example.org;
```

and change them to add a # in the beginning so they say

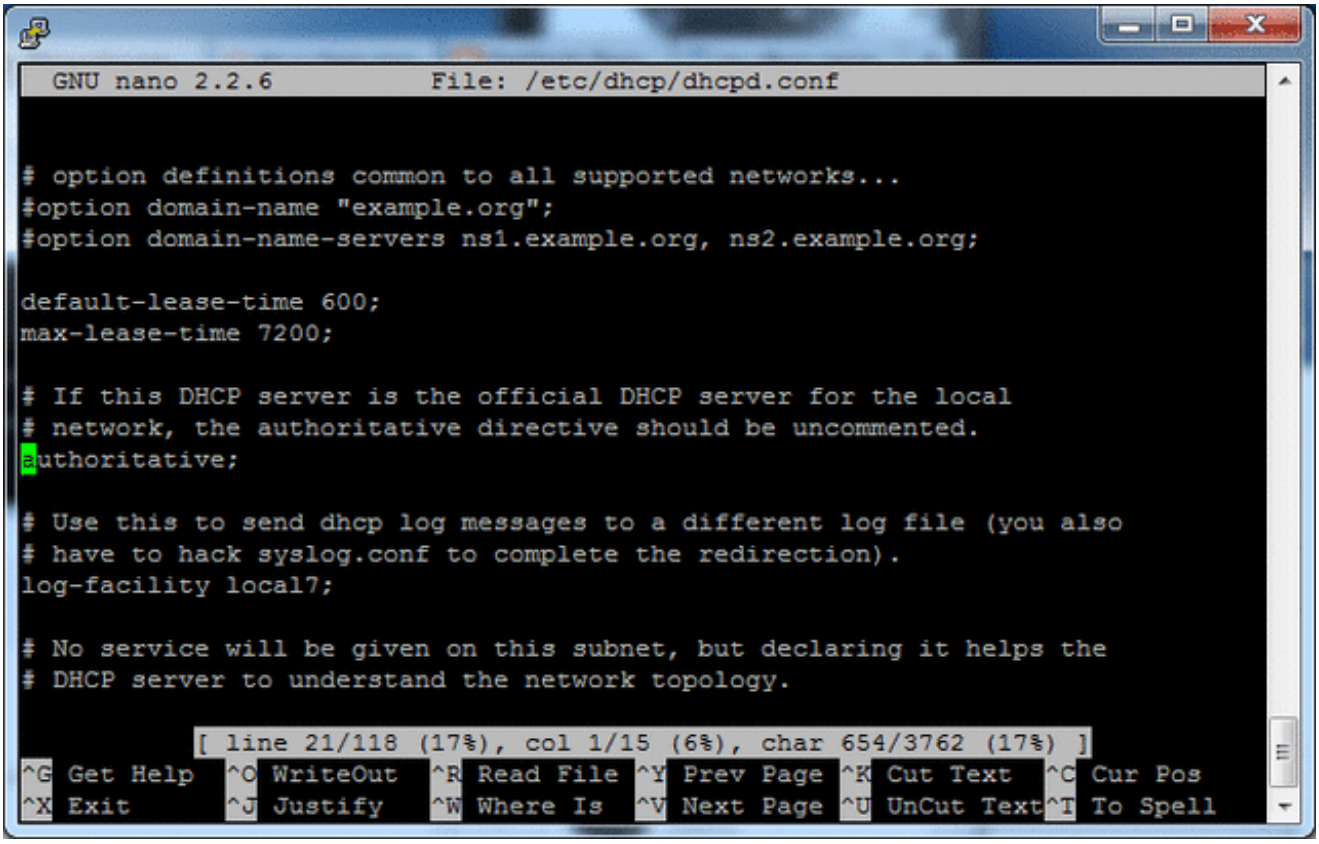
```
#option domain-name "example.org";
#option domain-name-servers ns1.example.org, ns2.example.org;
```

Find the lines that say

```
# If this DHCP server is the official DHCP server for the local
# network, the authoritative directive should be uncommented.
#authoritative;
```

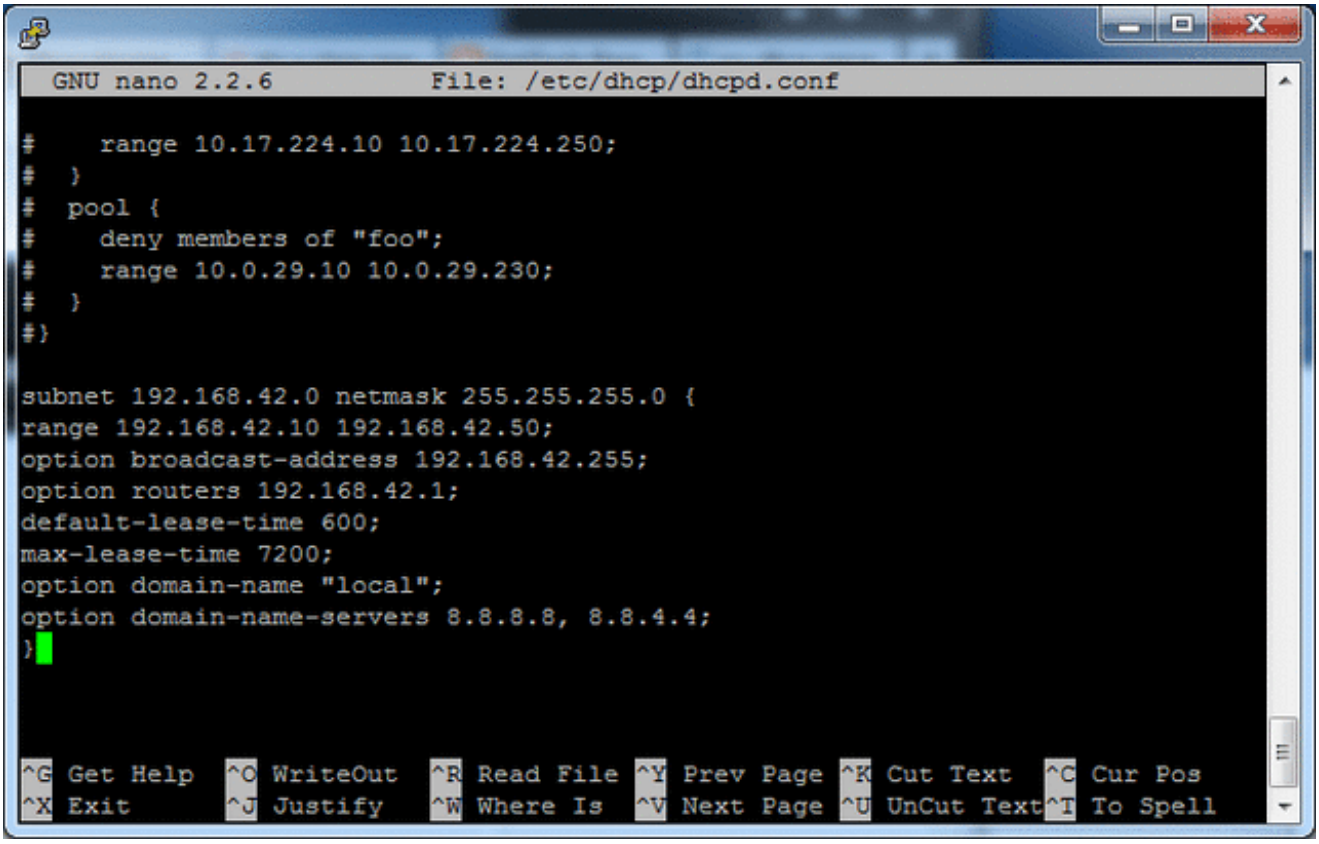
and remove the # so it says

```
# If this DHCP server is the official DHCP server for the local
# network, the authoritative directive should be uncommented.
authoritative;
```



Then scroll down to the bottom and add the following lines

```
subnet 192.168.42.0 netmask 255.255.255.0 {
 range 192.168.42.10 192.168.42.50;
 option broadcast-address 192.168.42.255;
 option routers 192.168.42.1;
 default-lease-time 600;
 max-lease-time 7200;
 option domain-name "local";
 option domain-name-servers 8.8.8.8, 8.8.4.4;
}
```

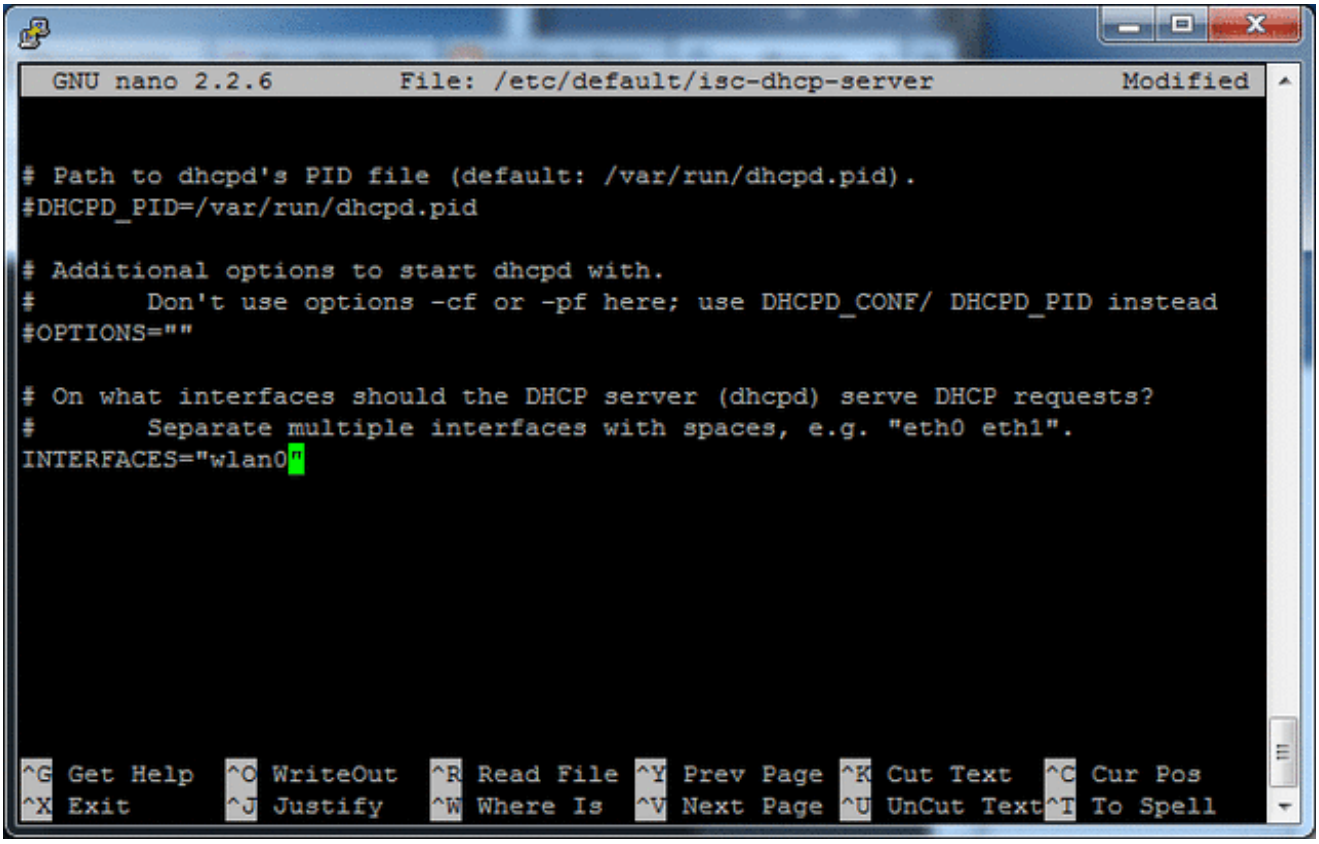


Save the file by typing in **Control-X** then **Y** then **return**

Run

> **sudo nano /etc/default/isc-dhcp-server**

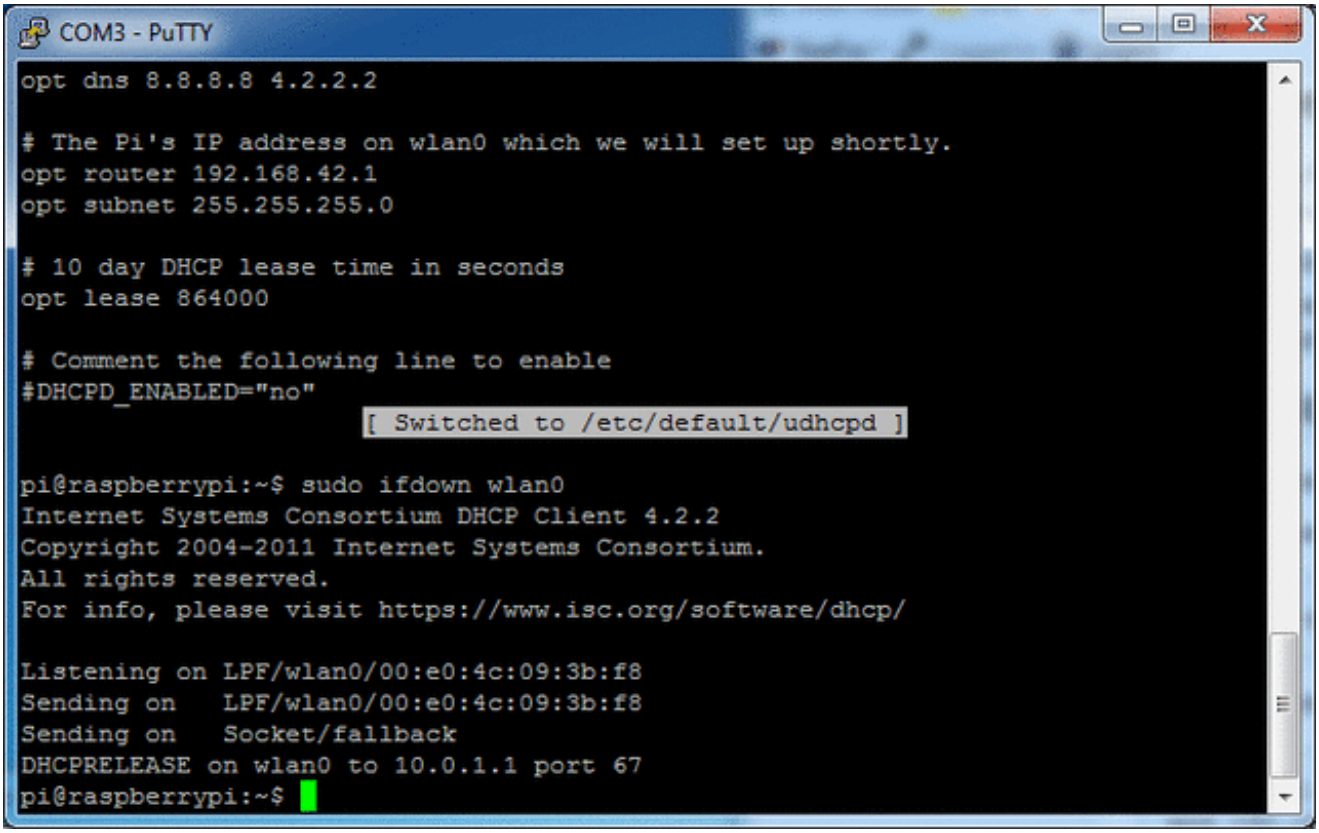
and scroll down to **INTERFACES=""** and update it to say **INTERFACES="wlan0"**



close and save the file

Set up wlan0 for static IP

If you happen to have wlan0 active because you set it up, run **sudo ifdown wlan0**
There's no harm in running it if you're not sure



Next we will set up the **wlan0** connection to be static and incoming. run **sudo nano /etc/network/interfaces** to edit the file

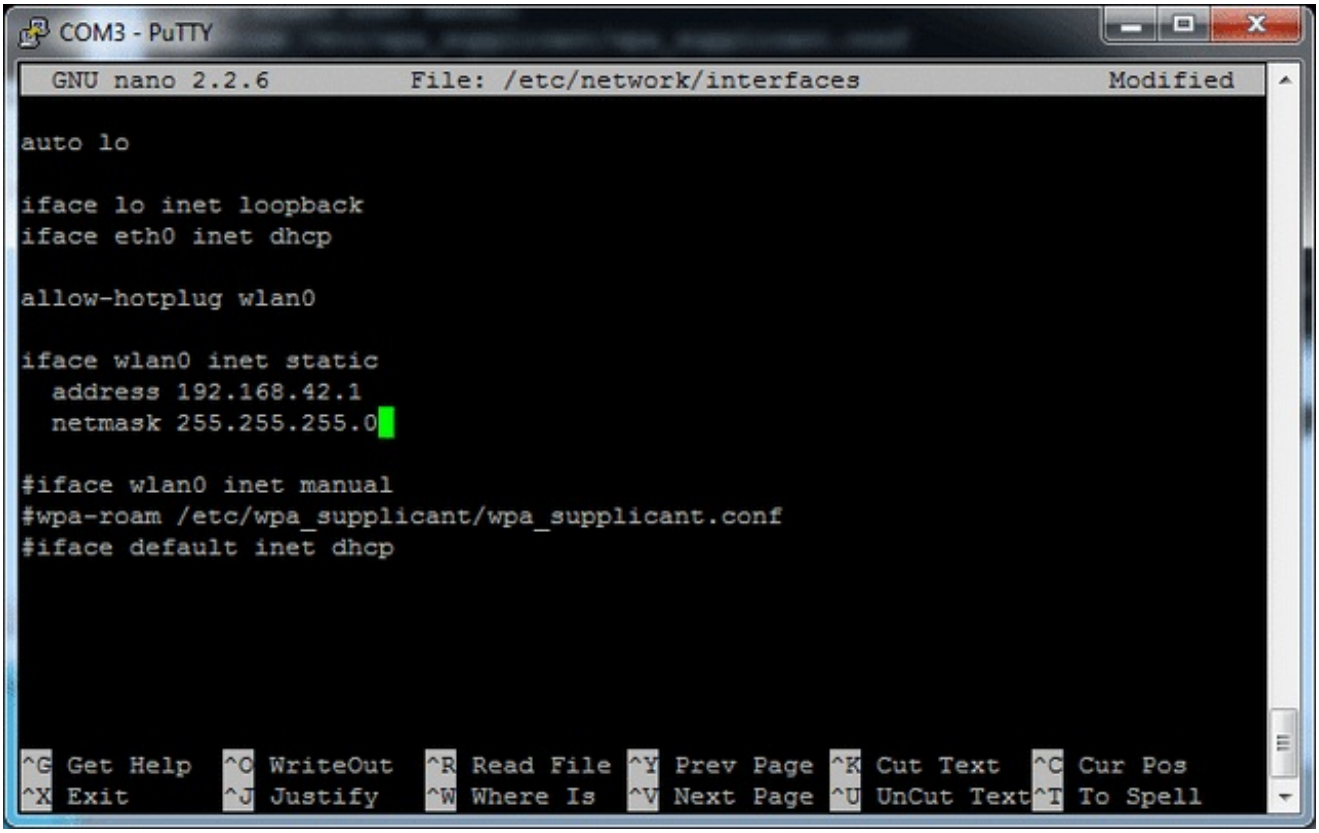
Find the line **auto wlan0** and add a **#** in front of the line, and in front of every line afterwards. If you don't have that line, just make sure it looks like the screenshot below in the end! Basically just remove any old **wlan0** configuration settings, we'll be changing them up

Depending on your existing setup/distribution there might be more or less text and it may vary a little bit

Add the lines

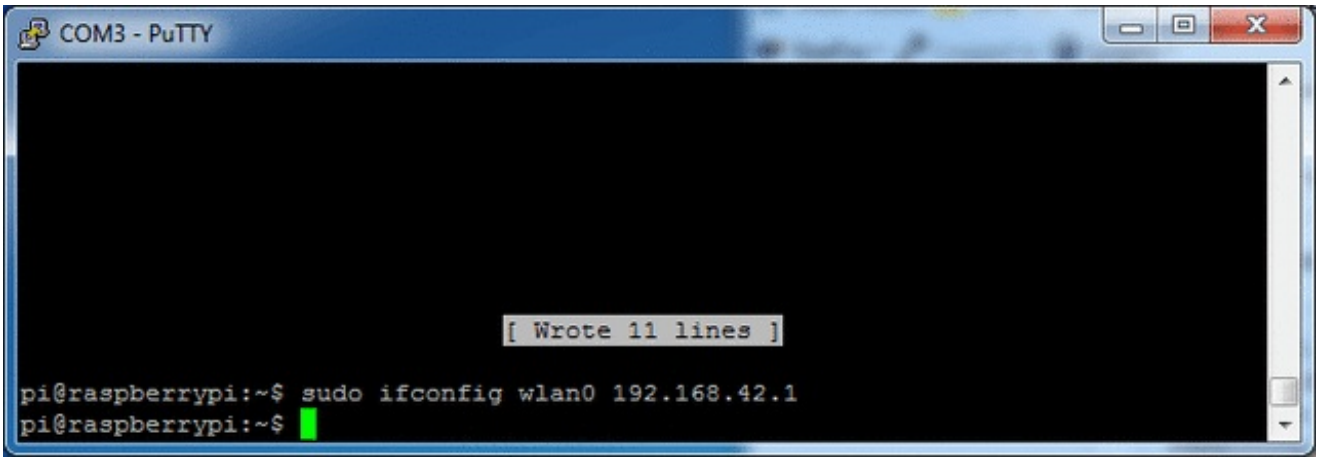
```
iface wlan0 inet static
  address 192.168.42.1
  netmask 255.255.255.0
```

After **allow-hotplug wlan0** - see below for an example of what it should look like. Any other lines afterwards should have a **#** in front to disable them



Save the file (Control-X Y <return>)

Assign a static IP address to the wifi adapter by running
sudo ifconfig wlan0 192.168.42.1



Configure Access Point

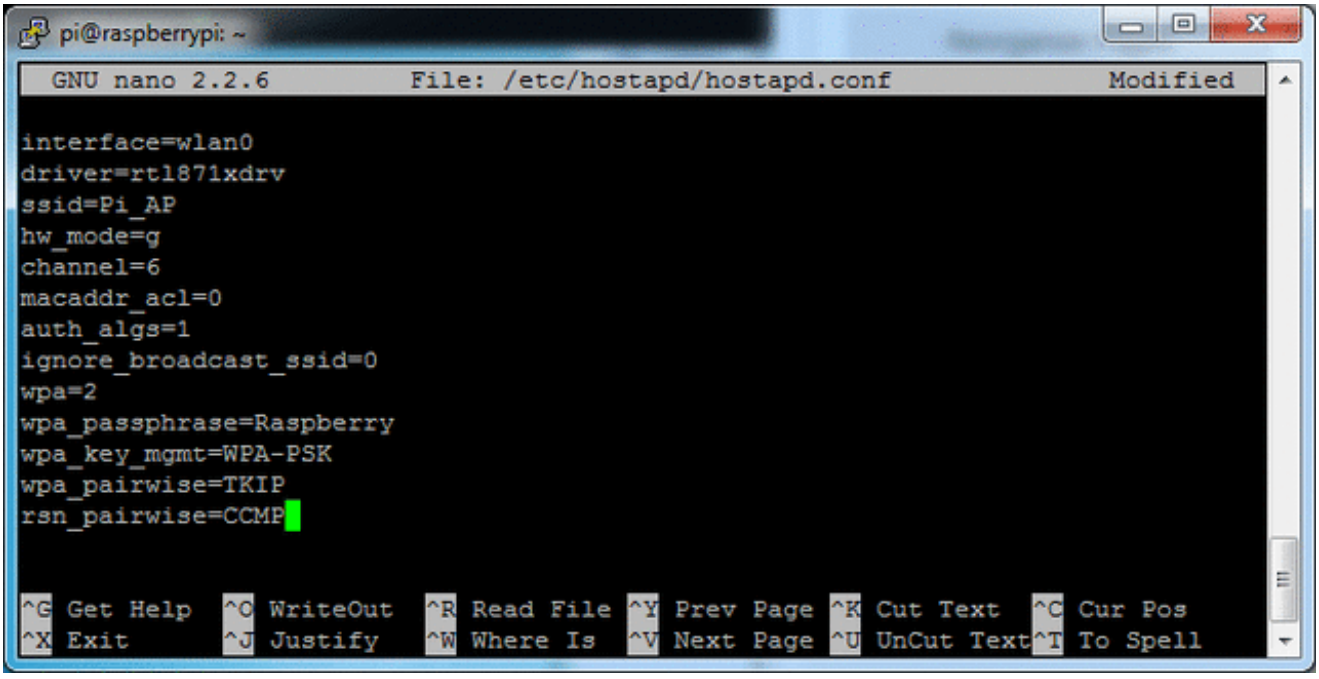
Now we can configure the access point details. We will set up a password-protected network so only people with the password can connect.

Create a new file by running **sudo nano /etc/hostapd/hostapd.conf**

Paste the following in, you can change the text after **ssid=** to another name, that will be the network broadcast name. The password can be changed with the text after **wpa_passphrase=**

```
interface=wlan0
driver=rtl871xdrv
ssid=Pi_AP
hw_mode=g
channel=6
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_passphrase=Raspberry
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
```

If you are not using the Adafruit wifi adapters, you may have to change the **driver=rtl871xdrv** to say **driver=nl80211** or something, we don't have tutorial support for that tho, YMMV!

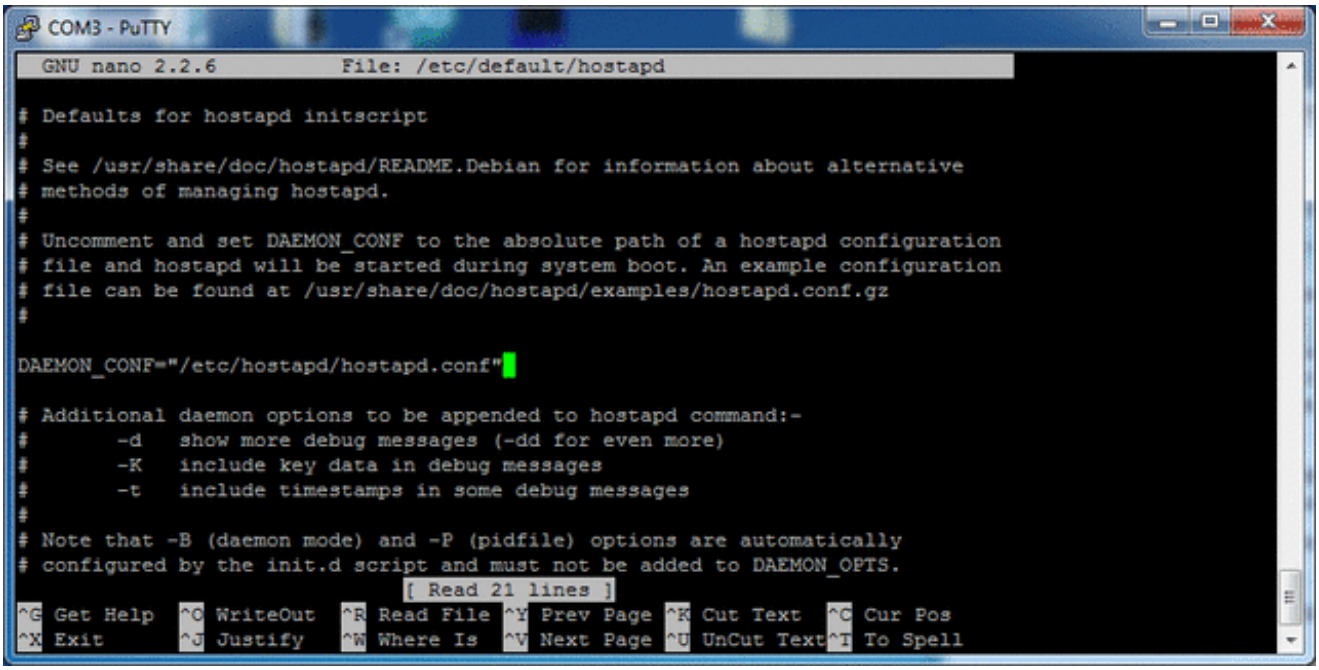


Save as usual. Make sure each line has no extra spaces or tabs at the end or beginning - this file is pretty picky!

Now we will tell the Pi where to find this configuration file. Run **sudo nano /etc/default/hostapd**

Find the line **#DAEMON_CONF=""** and edit it so it says
DAEMON_CONF="/etc/hostapd/hostapd.conf"
Don't forget to remove the **#** in front to activate it!

Then save the file



Configure Network Address Translation

Setting up NAT will allow multiple clients to connect to the WiFi and have all the data 'tunneled' through the single Ethernet IP. (But you should do it even if only one client is going to connect)

Run **sudo nano /etc/sysctl.conf**

Scroll to the bottom and add

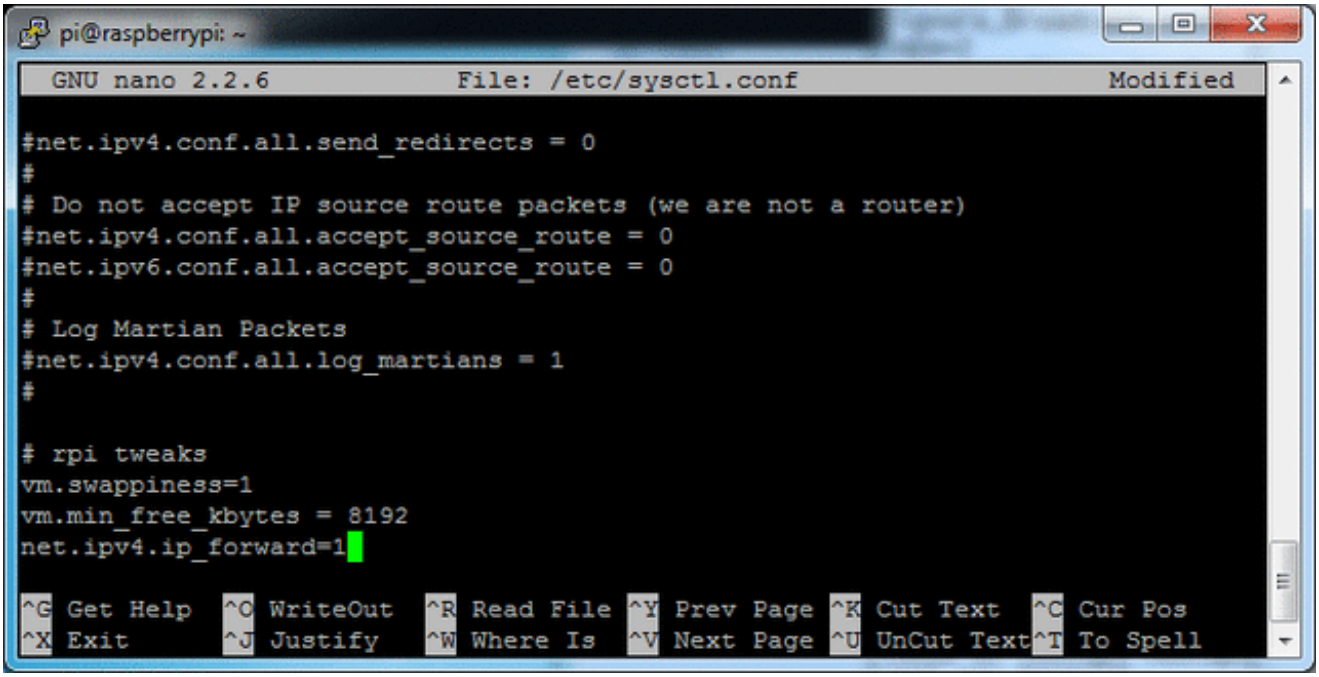
> **net.ipv4.ip_forward=1**

on a new line. Save the file. This will start IP forwarding on boot up

Also run

> **sudo sh -c "echo 1 > /proc/sys/net/ipv4/ip_forward"**

to activate it immediately



Run the following commands to create the network translation between the ethernet port **eth0** and the wifi port **wlan0**

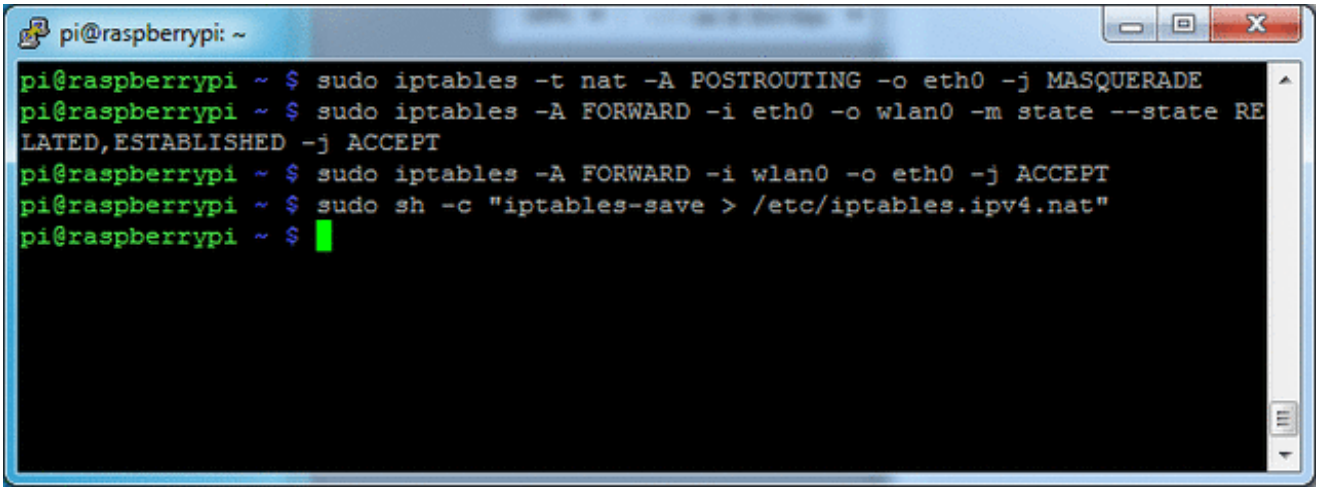
```
sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
sudo iptables -A FORWARD -i eth0 -o wlan0 -m state --state RELATED,ESTABLISHED -j ACCEPT
sudo iptables -A FORWARD -i wlan0 -o eth0 -j ACCEPT
```

You can check to see whats in the tables with

> **sudo iptables -t nat -S**
> **sudo iptables -S**

To make this happen on reboot (so you don't have to type it every time) run

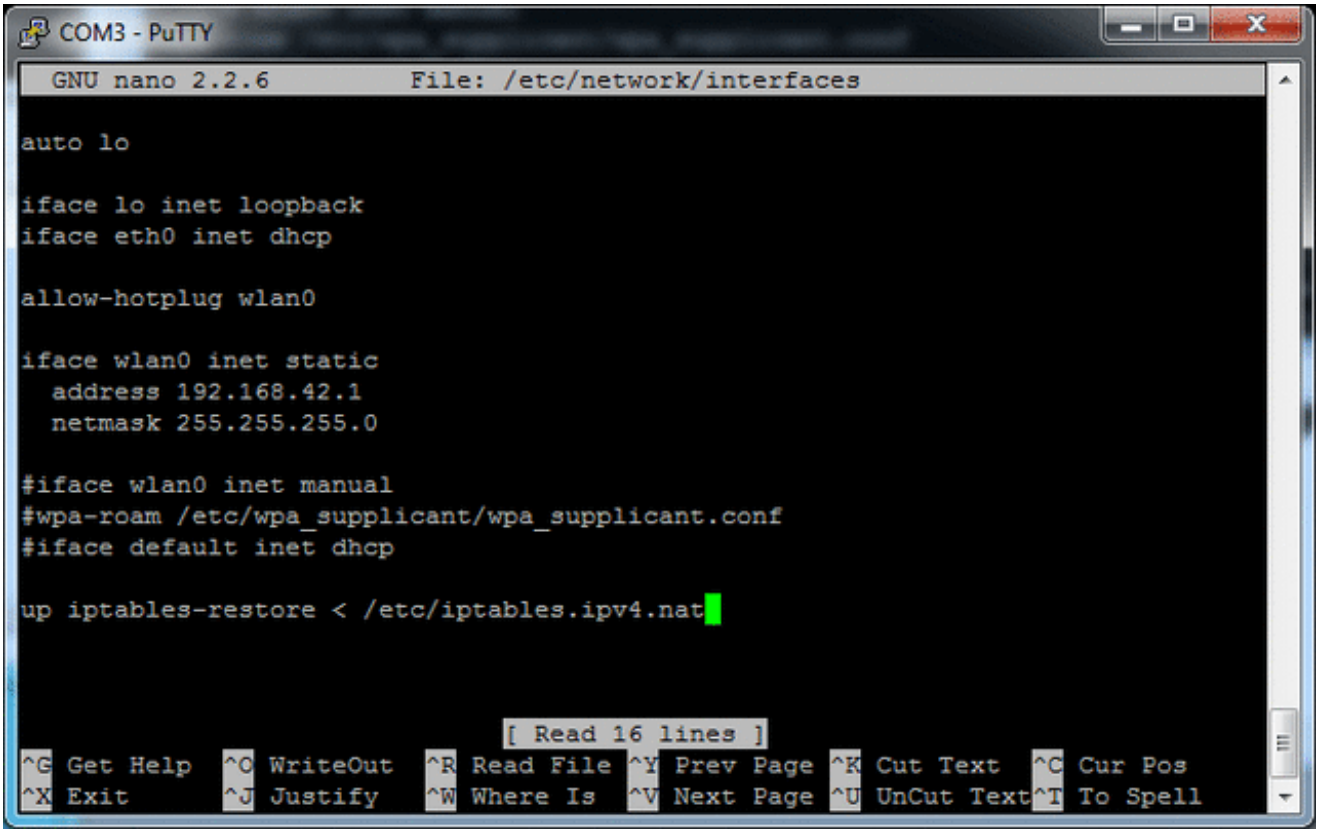
> **sudo sh -c "iptables-save > /etc/iptables.ipv4.nat"**



run **sudo nano /etc/network/interfaces** and add

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

to the very end



Update hostapd

Before we can run the access point software, we have to update it to a version that supports the WiFi adapter.
First get the new version by typing in

```
wget http://adafruit-download.s3.amazonaws.com/adafruit_hostapd_14128.zip
```

to download the new version (check the next section for how to compile your own updated **hostapd**) then

```
unzip adafruit_hostapd_14128.zip
```

to uncompress it. Move the old version out of the way with

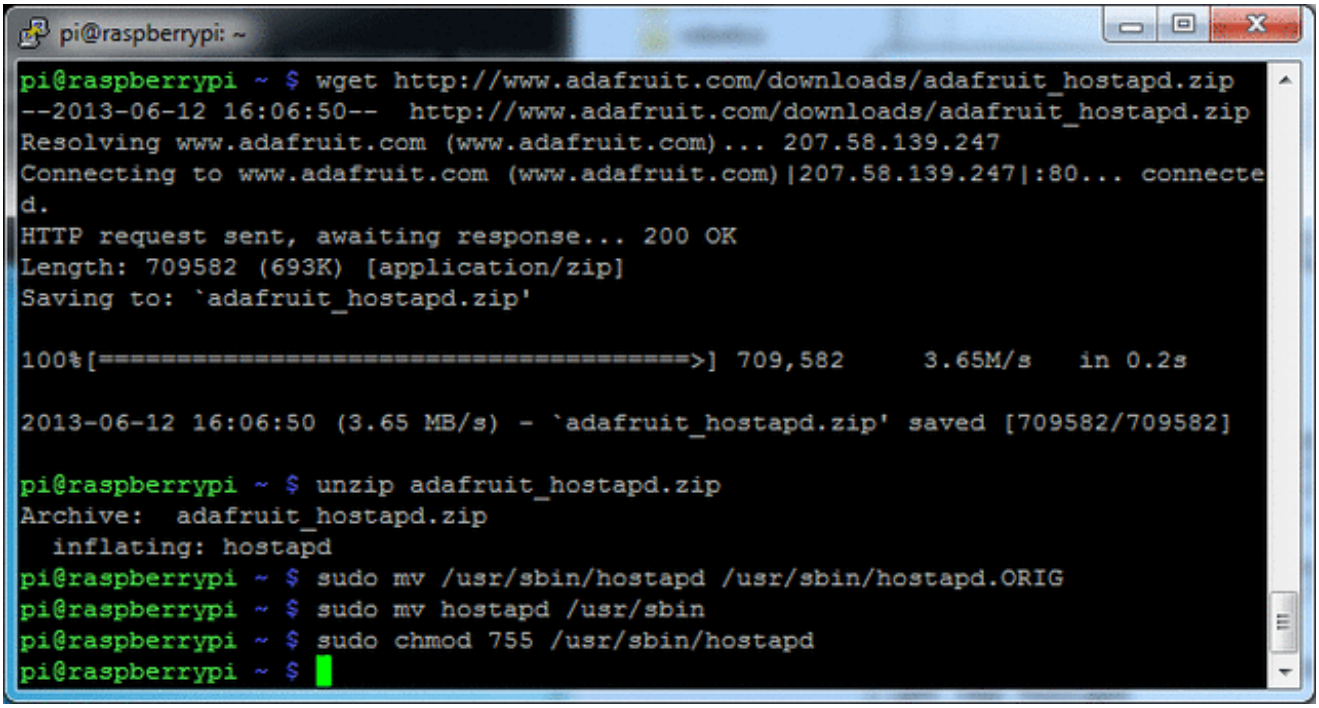
```
sudo mv /usr/sbin/hostapd /usr/sbin/hostapd.ORIG
```

And move the new version back with

```
sudo mv hostapd /usr/sbin
```

set it up so its valid to run with

```
sudo chmod 755 /usr/sbin/hostapd
```

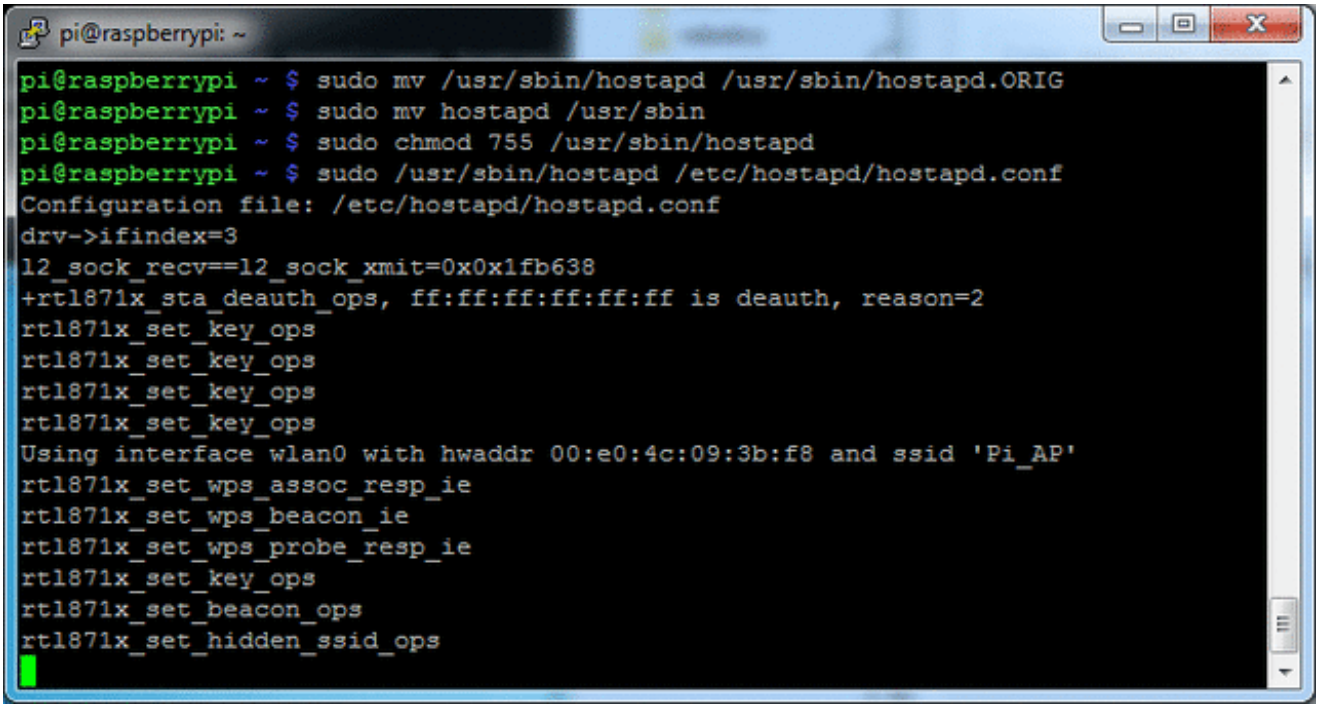


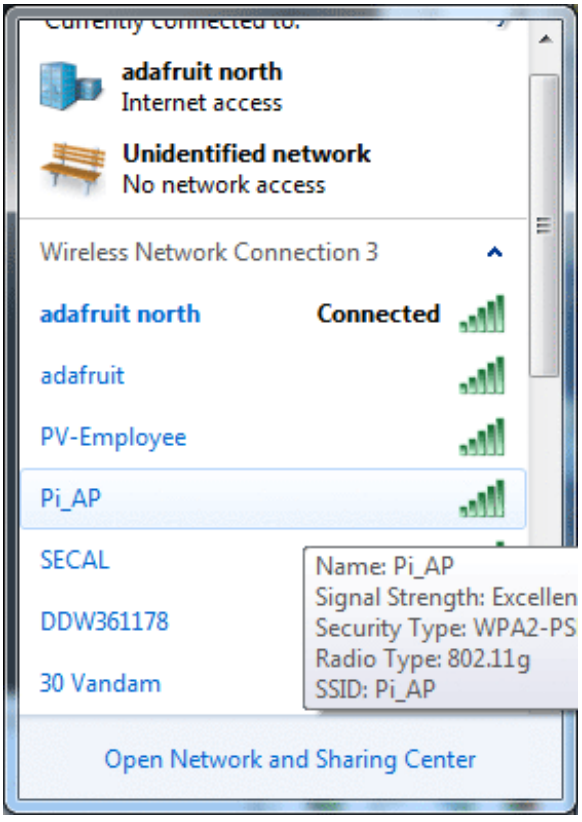
First test!

Finally we can test the access point host! Run

> **sudo /usr/sbin/hostapd /etc/hostapd/hostapd.conf**

To manually run **hostapd** with our configuration file. You should see it set up and use **wlan0** then you can check with another wifi computer that you see your SSID show up. If so, you have successfully set up the access point.





You can try connecting and disconnecting from the Pi_AP with the password you set before (probably `Raspberry` if you copied our hostapd config), debug text will display on the Pi console but you won't be able to connect through to the Ethernet connection yet.
Cancel the test by typing **Control-C** in the Pi console to get back to the Pi command line

Finishing up!

OK now that we know it works, time to set it up as a 'daemon' - a program that will start when the Pi boots.
Run the following commands

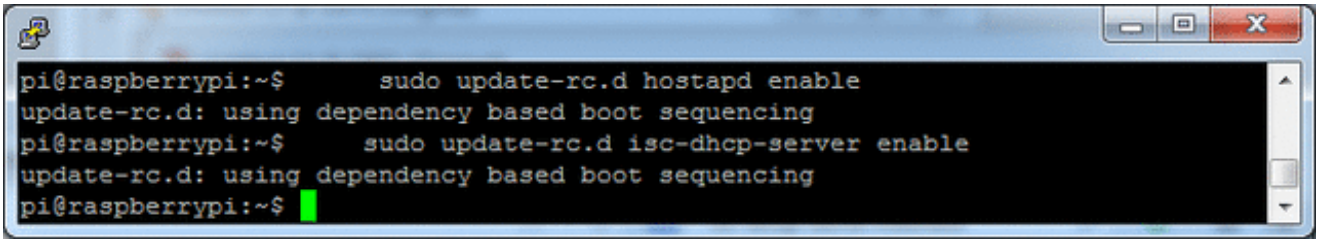
> **sudo service hostapd start**
> **sudo service isc-dhcp-server start**

you can always check the status of the host AP server and the DHCP server with

> **sudo service hostapd status**
> **sudo service isc-dhcp-server status**

To start the daemon services. Verify that they both start successfully (no 'failure' or 'errors')
Then to make it so it runs every time on boot

> **sudo update-rc.d hostapd enable**
> **sudo update-rc.d isc-dhcp-server enable**



Extra: Removing WPA-Supplicant

Depending on your distro, you *may* need to remove WPASupplicant. Do so by running this command:

```
sudo mv /usr/share/dbus-1/system-services/fi.epitest.hostap.WPASupplicant.service ~/
```

and then rebooting (**sudo reboot**)

Connect and Test

Now that we have the software installed on a Pi, it's time to connect to it and test the connection. I'm using a Windows computer but any kind should work fine

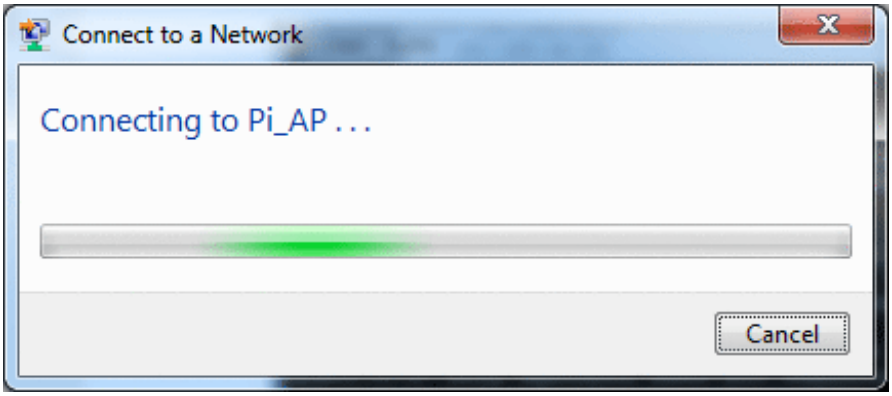
On the Pi, run the command **tail -f /var/log/syslog** to watch the system log data, handy for checking and debugging whats going on!

Connect with another computer to the AP you made in the previous step



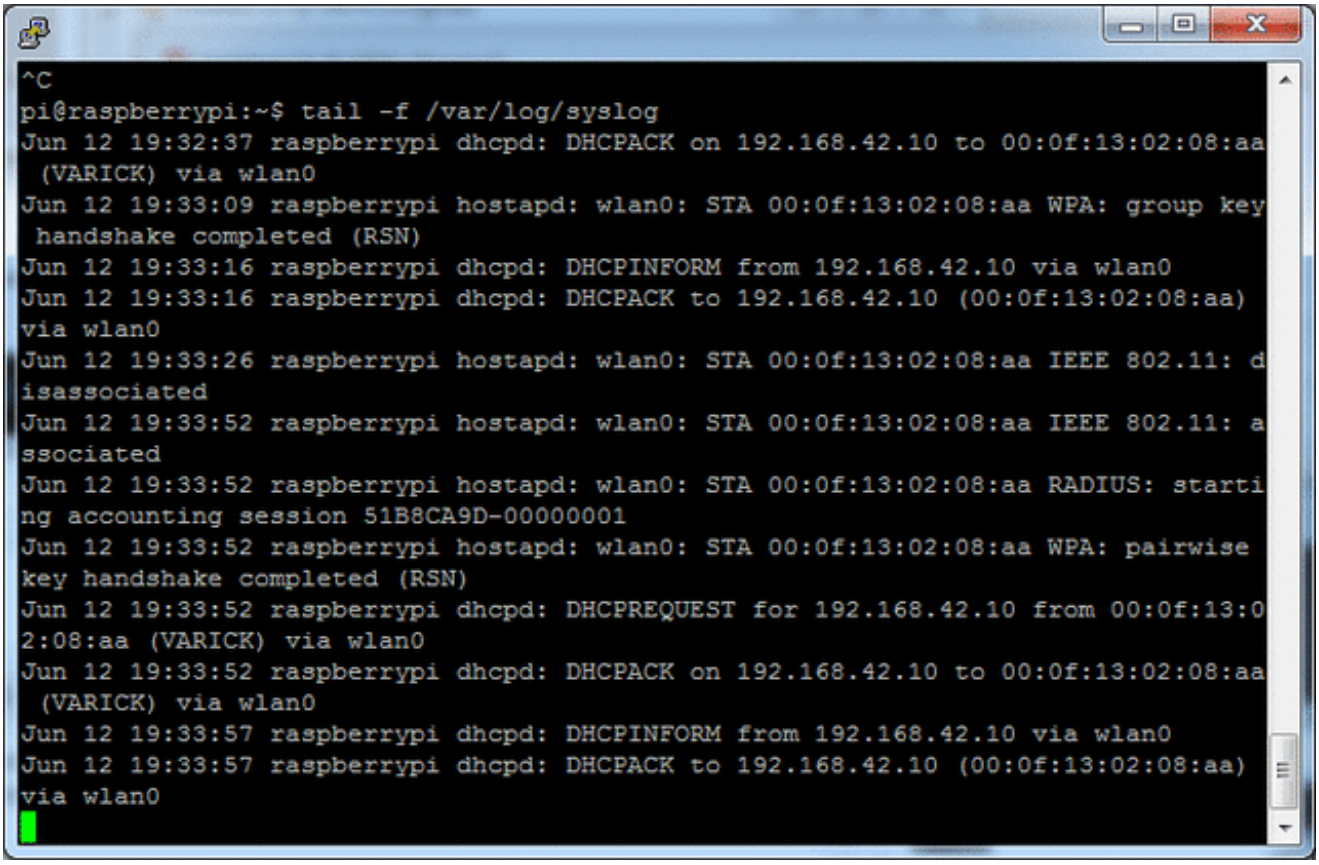
Enter the WPA key you specified in the previous step





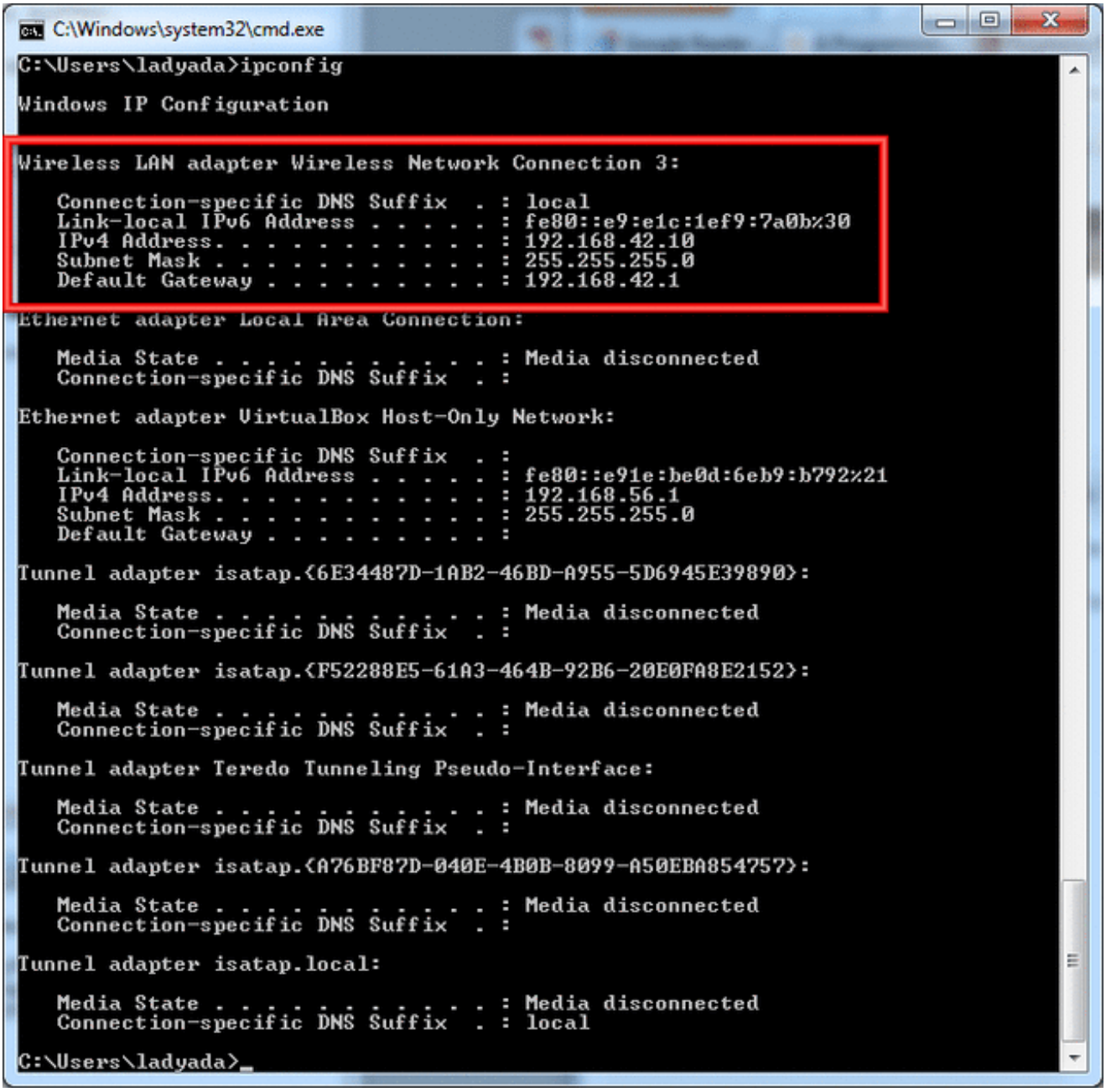
In the Pi syslog you should see stuff like this! It indicates that a client connected, at what time and what IP address was given to them

If you can't connect at all, something is wrong with **hostapd**



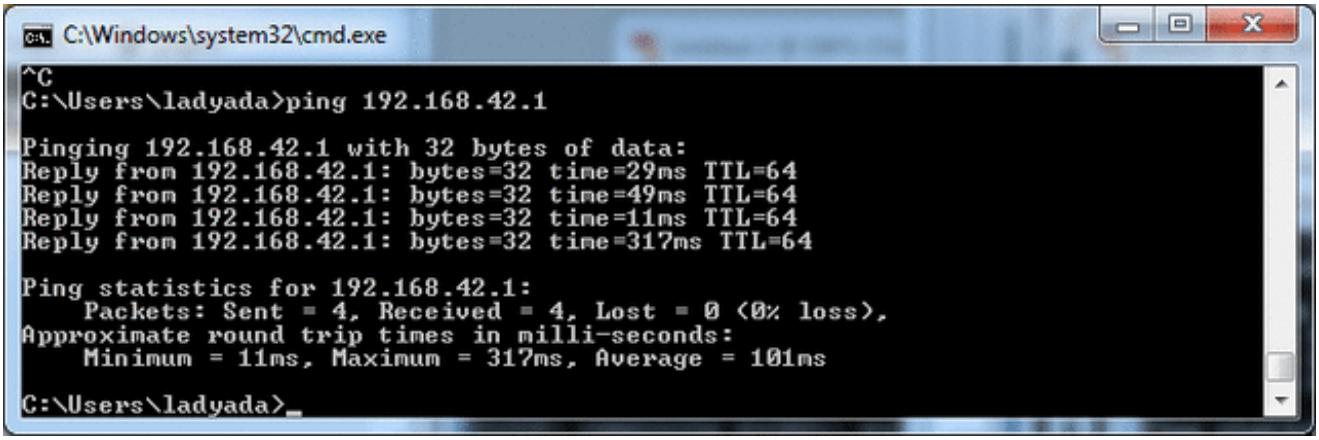
On your computer, open up a **Terminal** (mac/linux) or **Start->Run->cmd** to open up a command line

First check what **ifconfig** (mac/linux) or **ipconfig** (windows) says. You should have IP address in the 192.168.42.10-50 range

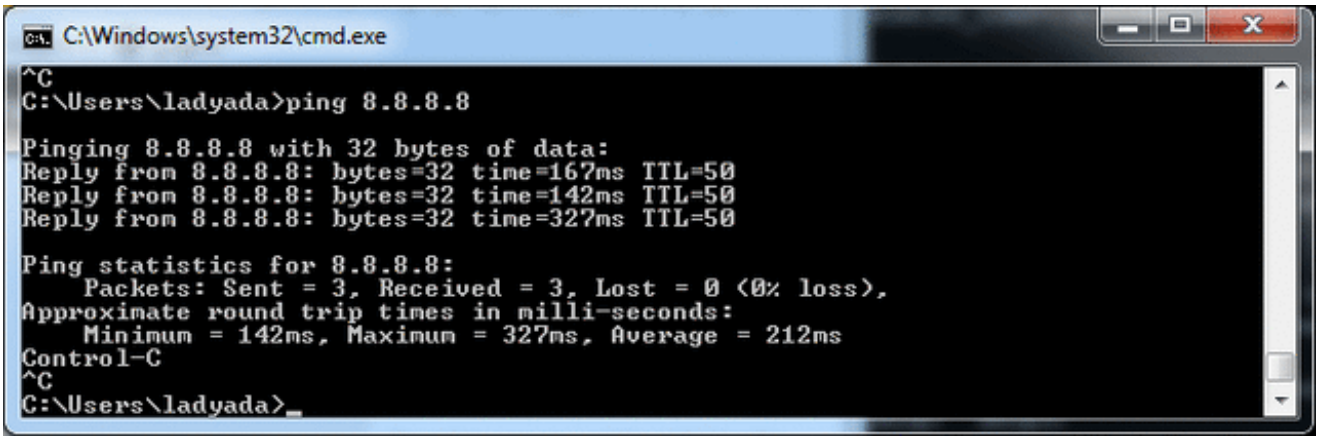


Try pinging the Pi, its address is 192.168.42.1 - on windows it will ping 3 times and quit. On mac/linux press Control-C to quit after a few pings. You should get successful pings as seen below

If that doesn't work, something is wrong with **hostapd** or **dhcpd** (more likely)



Next try pinging 8.8.8.8, if this doesn't work but the previous does, something is wrong with **dhcpd** or the NAT configuration (more likely)



Finally, we'll check that DNS works, try pinging www.mit.edu (http://adafru.it/cfT). If this doesn't work, something is wrong with **dhcpd**

If everything is good so far, try browsing the internet, sending email, etc. You are now using your Pi as a Wifi Router!

More!

Its possible to set up your router for open or WEP access, but we don't cover that here (and it's not as secure!) You might want to search around for tutorials such as this one that cover (http://adafru.it/cDx)**hostapd (http://adafru.it/cDx)** options (http://adafru.it/cDx)

Compiling hostapd

You may have noticed that one step is downloading a copy of hostapd from adafruit.com and swapping it with yours. In case you want to compile your own, here's how (its easy but not necessary if you are OK with using our binary)

1. Go to the Realtek downloads page http://152.104.125.41/downloads/downloadsView.aspx?Langid=1&PNid=21&PFid=48&Level=5&Conn=4&ProdID=27... (http://adafru.it/cfY)
2. Download linux 3.4.4_4749
3. Copy the zip to the SD card using any computer which will place it in the Pi's /boot directory (or somehow get that file onto your Pi)
4. Boot the Pi from the SD card
5. sudo mv /boot/RTL8192xC_USB_linux_v3.4.4_4749.20121105.zip .
6. unzip RTL8192xC_USB_linux_v3.4.4_4749.20121105.zip
7. mv RTL8188C_8192C_USB_linux_v3.4.4_4749.20121105/ rtl
8. cd rtl
9. cd wpa_supplicant_hostapd
10. unzip wpa_supplicant_hostapd-0.8_rtw_20120803.zip
11. cd wpa_supplicant_hostapd-0.8/
12. cd hostapd
13. make
14. *have a sandwich*
15. when done, **hostapd** binary is in the directory