



## 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) (<http://adafru.it/1914>) (or B)- Ethernet is required
- [Ethernet cable](http://adafru.it/730) (<http://adafru.it/730>)
- [WiFi adapter](http://adafru.it/814) (<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) (<http://adafru.it/1121>)
- Power supply for your Pi & a Micro USB cable
- [USB Console cable](http://adafru.it/954) (optional) - [this makes it a little easier to debug the system](http://adafru.it/954) (<http://adafru.it/954>)
- [Case for your Pi](http://adafru.it/2258) (optional) (<http://adafru.it/2258>)
- [A SD or MicroSD card reader](http://adafru.it/939) (<http://adafru.it/939>) (optional)

Our [Pi B+ starter pack](http://adafru.it/2125) (<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.

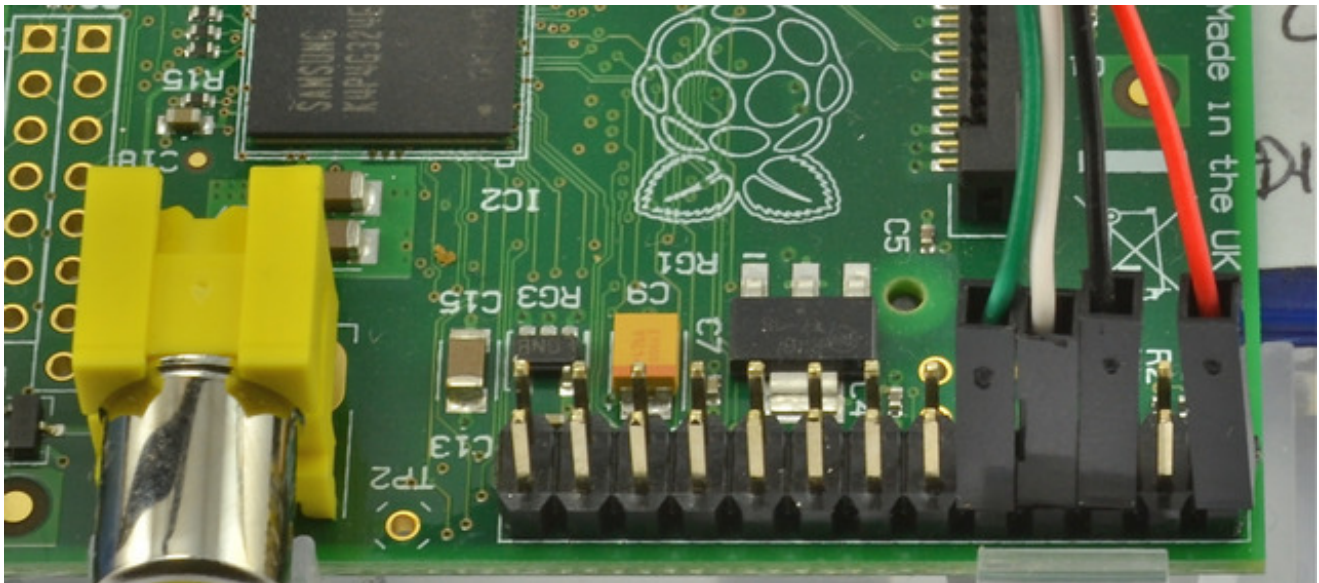
Please follow the tutorials in order to

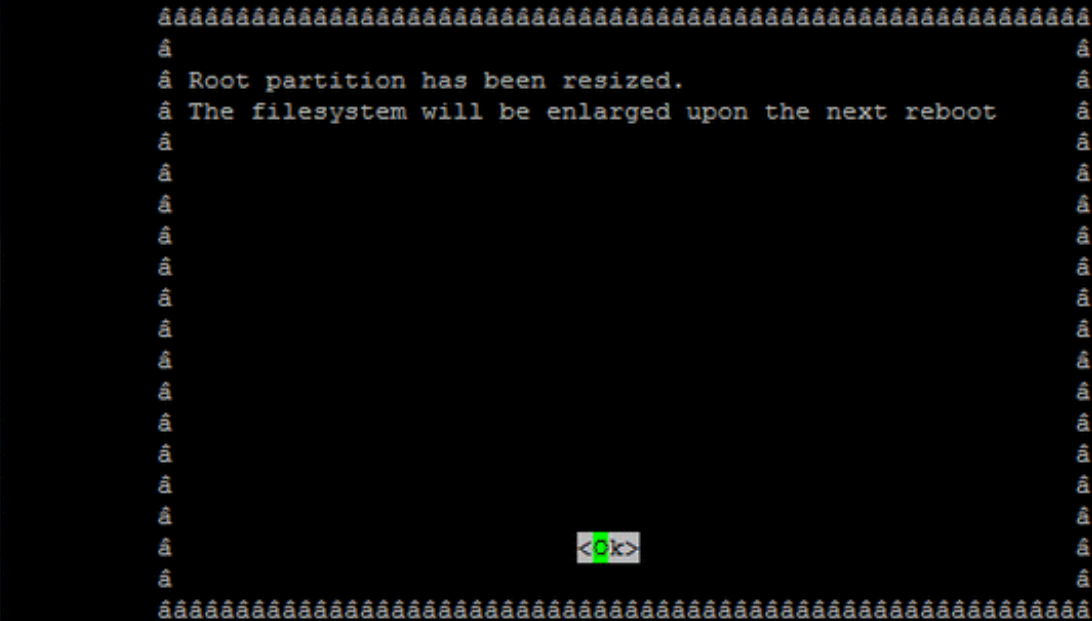
1. [Install the OS onto your SD card \(http://adafru.it/aWq\)](http://adafru.it/aWq)
2. [Boot the Pi and configure \(http://adafru.it/aUa\)](http://adafru.it/aUa)  
**Don't forget to change the default password for the 'pi' account!**
3. [Set up and test the Ethernet and Wifi connection \(http://adafru.it/aUB\)](http://adafru.it/aUB)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

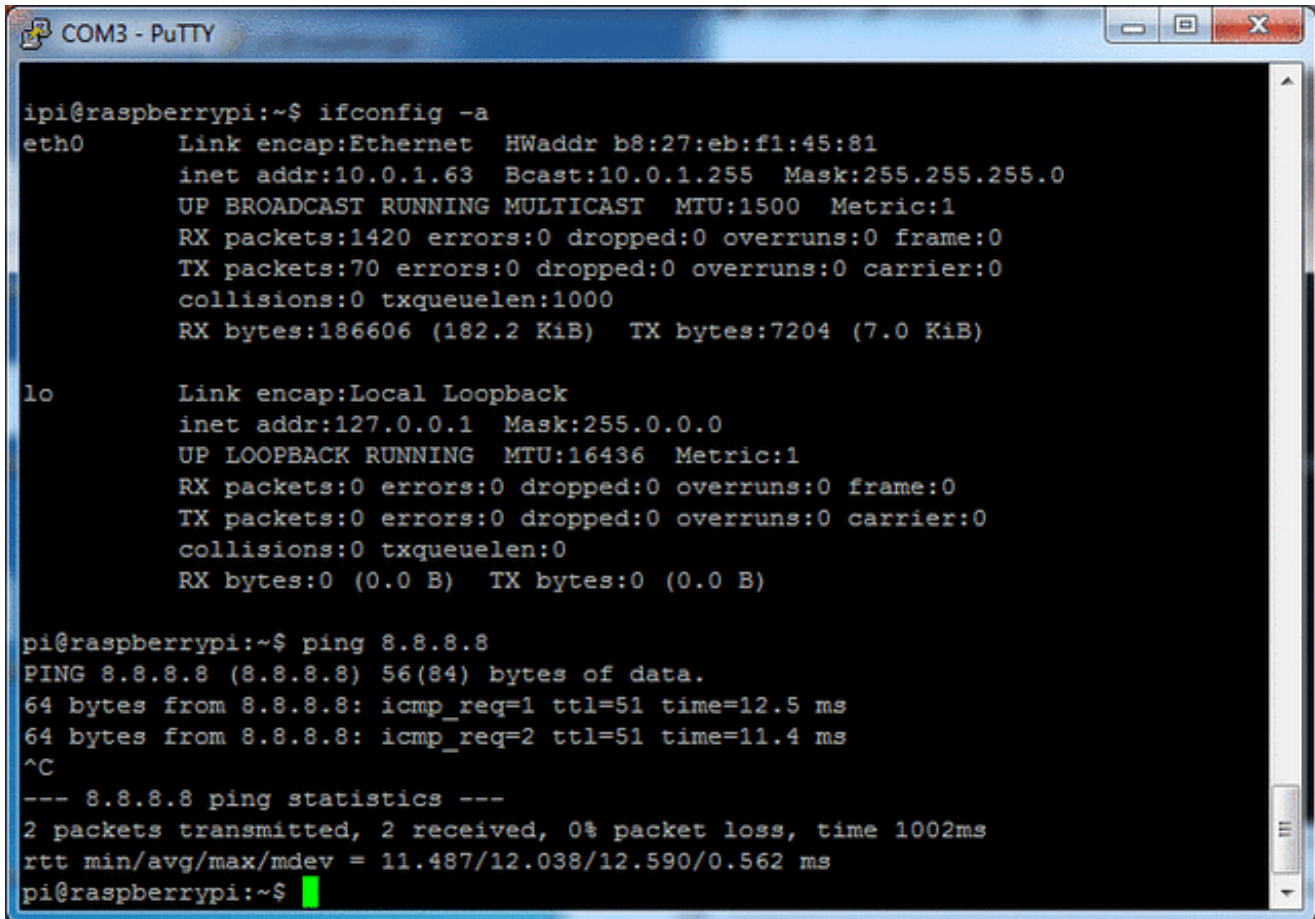




Don't forget to expand the SD card, or you may run out of space!

## Check Ethernet & Wifi

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



```
COM3 - PuTTY

ipi@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:1420 errors:0 dropped:0 overruns:0 frame:0
          TX packets:70 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:186606 (182.2 KiB)  TX bytes:7204 (7.0 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)

pi@raspberrypi:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_req=1 ttl=51 time=12.5 ms
64 bytes from 8.8.8.8: icmp_req=2 ttl=51 time=11.4 ms
^C
--- 8.8.8.8 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 11.487/12.038/12.590/0.562 ms
pi@raspberrypi:~$
```

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.



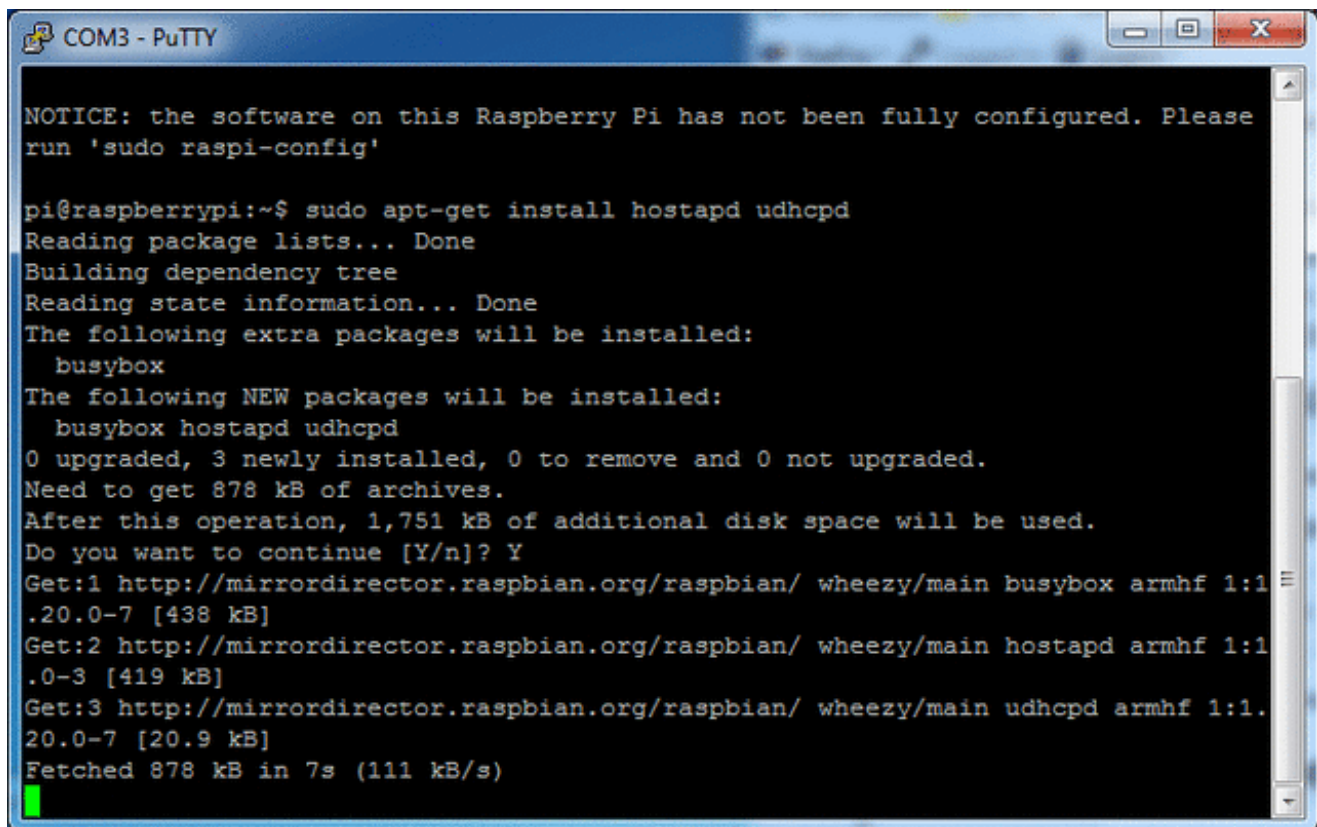
```
pi@raspberrypi: ~  
config pi@raspberrypi ~ $ ifconfig -a  
eth0      Link encap:Ethernet  HWaddr b8:27:eb:f1:45:81  
          inet addr:10.0.1.63  Bcast:10.0.1.255  Mask:255.255.255.0  
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1  
          RX packets:112 errors:0 dropped:0 overruns:0 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)



```
COM3 - PuTTY

NOTICE: the software on this Raspberry Pi has not been fully configured. Please
run 'sudo raspi-config'

pi@raspberrypi:~$ sudo apt-get install hostapd udhcpd
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  busybox
The following NEW packages will be installed:
  busybox hostapd udhcpd
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 878 kB of archives.
After this operation, 1,751 kB of additional disk space will be used.
Do you want to continue [Y/n]? Y
Get:1 http://mirrordirector.raspbian.org/raspbian/ wheezy/main busybox armhf 1:1
.20.0-7 [438 kB]
Get:2 http://mirrordirector.raspbian.org/raspbian/ wheezy/main hostapd armhf 1:1
.0-3 [419 kB]
Get:3 http://mirrordirector.raspbian.org/raspbian/ wheezy/main udhcpd armhf 1:1
.20.0-7 [20.9 kB]
Fetched 878 kB in 7s (111 kB/s)
```

*(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

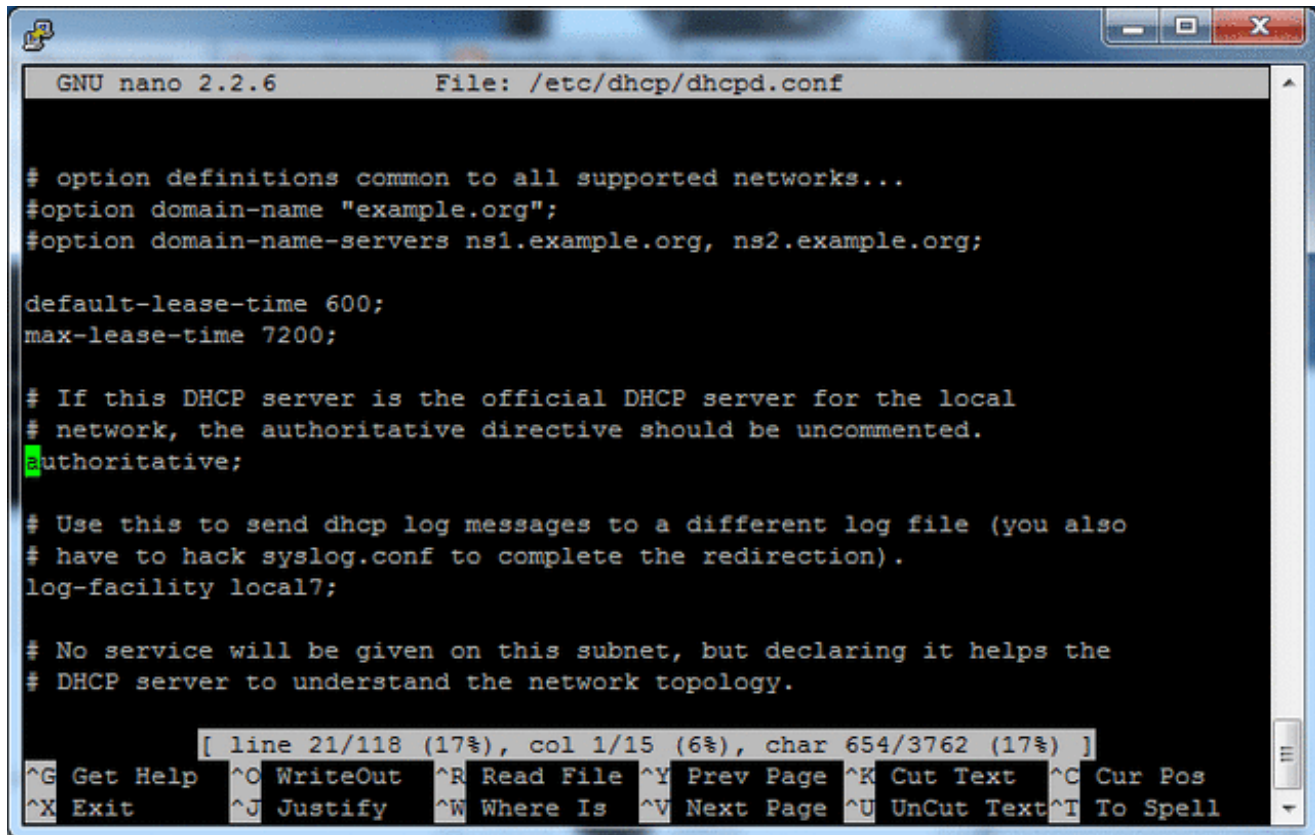
```
#option domain-name "example.org";  
#option domain-name-servers ns1.example.org, ns2.example.org;
```

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;
```



```
GNU nano 2.2.6 File: /etc/dhcp/dhcpd.conf

# option definitions common to all supported networks...
#option domain-name "example.org";
#option domain-name-servers ns1.example.org, ns2.example.org;

default-lease-time 600;
max-lease-time 7200;

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

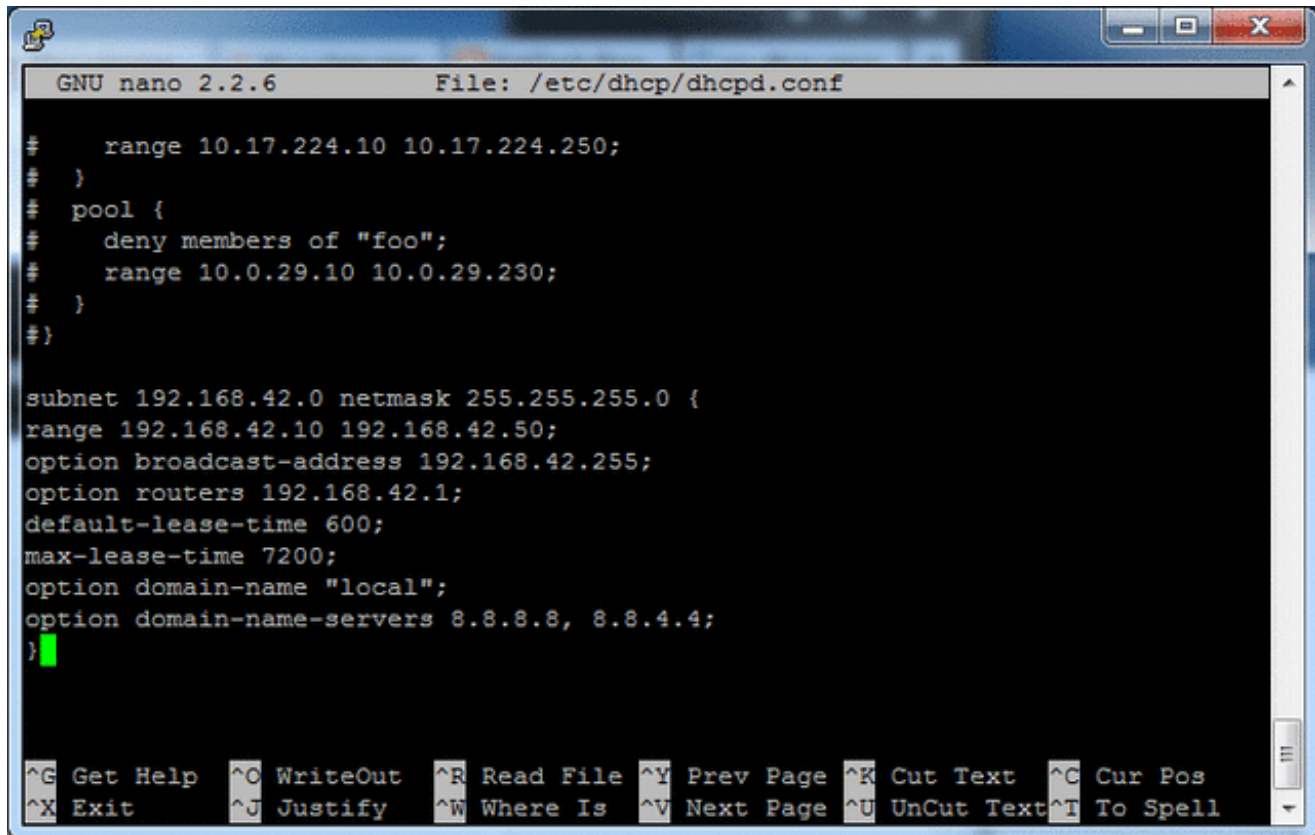
# Use this to send dhcp log messages to a different log file (you also
# have to hack syslog.conf to complete the redirection).
log-facility local7;

# No service will be given on this subnet, but declaring it helps the
# DHCP server to understand the network topology.

[ line 21/118 (17%), col 1/15 (6%), char 654/3762 (17%) ]
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
```

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;
}
```



```
GNU nano 2.2.6      File: /etc/dhcp/dhcpd.conf

#       range 10.17.224.10 10.17.224.250;
#     }
#   pool {
#     deny members of "foo";
#     range 10.0.29.10 10.0.29.230;
#   }
# }

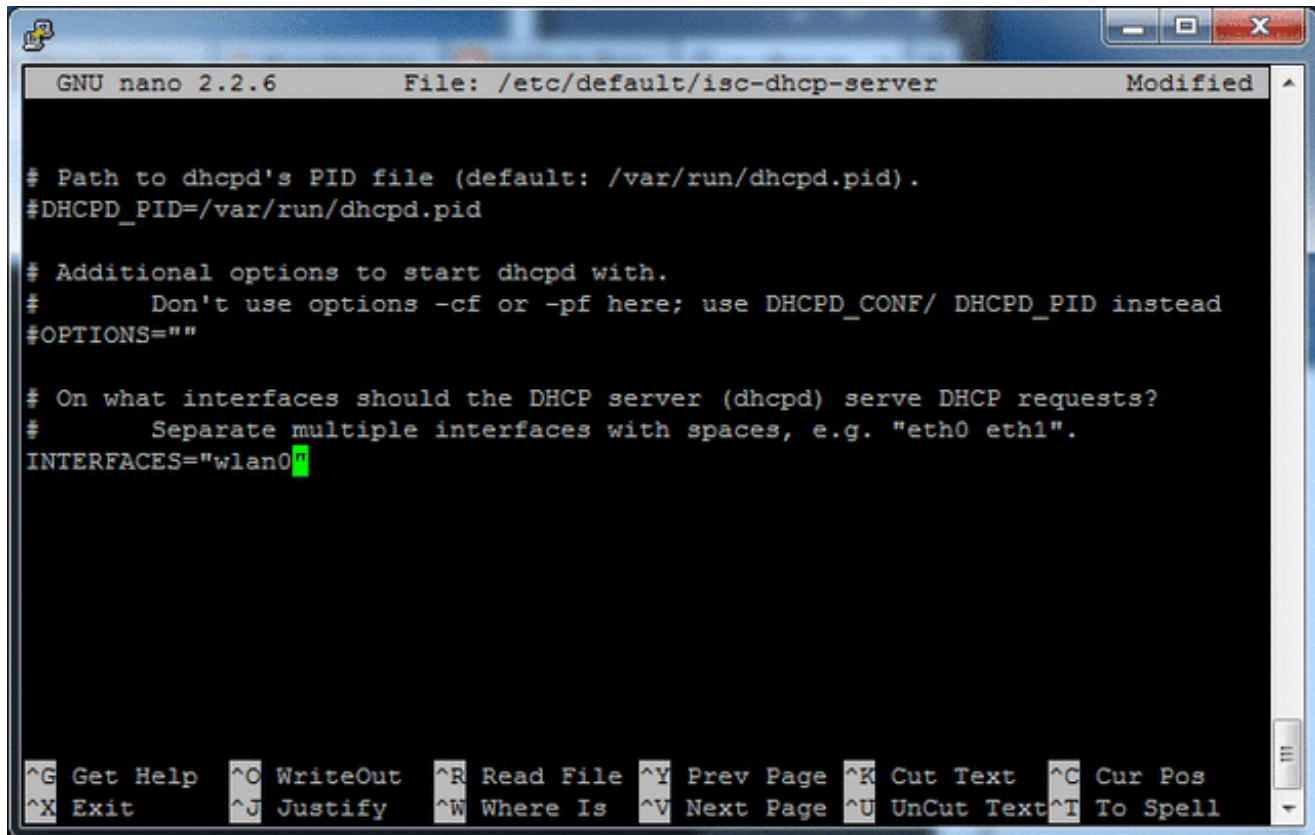
subnet 192.168.42.0 netmask 255.255.255.0 {
range 192.168.42.10 192.168.42.50;
option broadcast-address 192.168.42.255;
option routers 192.168.42.1;
default-lease-time 600;
max-lease-time 7200;
option domain-name "local";
option domain-name-servers 8.8.8.8, 8.8.4.4;
}
```

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"**



```
GNU nano 2.2.6      File: /etc/default/isc-dhcp-server      Modified

# Path to dhcpd's PID file (default: /var/run/dhcpd.pid) .
#DHCPD_PID=/var/run/dhcpd.pid

# Additional options to start dhcpd with.
#       Don't use options -cf or -pf here; use DHCPD_CONF/ DHCPD_PID instead
#OPTIONS=""

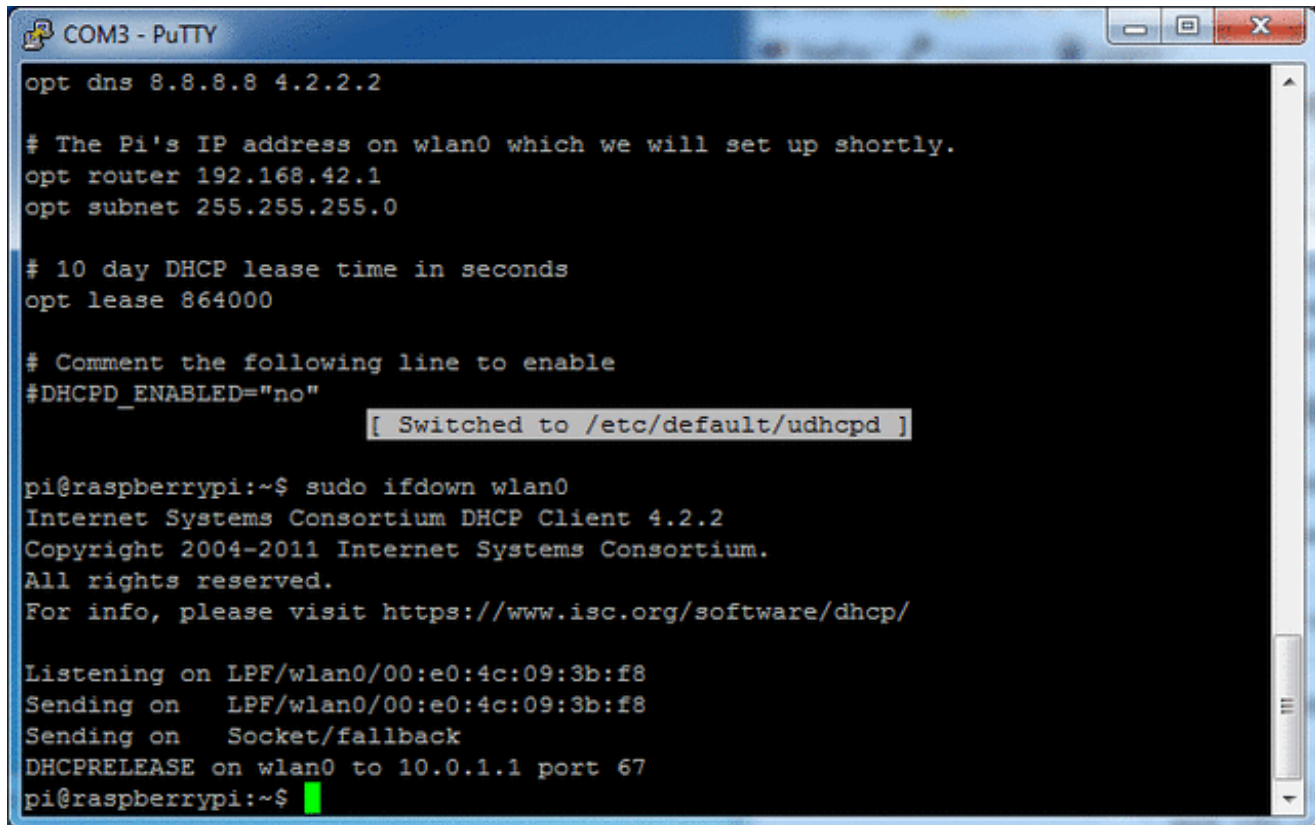
# On what interfaces should the DHCP server (dhcpd) serve DHCP requests?
#       Separate multiple interfaces with spaces, e.g. "eth0 eth1".
INTERFACES="wlan0"
```

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





```
COM3 - PuTTY
opt dns 8.8.8.8 4.2.2.2

# The Pi's IP address on wlan0 which we will set up shortly.
opt router 192.168.42.1
opt subnet 255.255.255.0

# 10 day DHCP lease time in seconds
opt lease 864000

# Comment the following line to enable
#DHCPD_ENABLED="no"
[ Switched to /etc/default/udhcpd ]

pi@raspberrypi:~$ sudo ifdown wlan0
Internet Systems Consortium DHCP Client 4.2.2
Copyright 2004-2011 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/

Listening on LPF/wlan0/00:e0:4c:09:3b:f8
Sending on   LPF/wlan0/00:e0:4c:09:3b:f8
Sending on   Socket/fallback
DHCPRELEASE on wlan0 to 10.0.1.1 port 67
pi@raspberrypi:~$
```

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

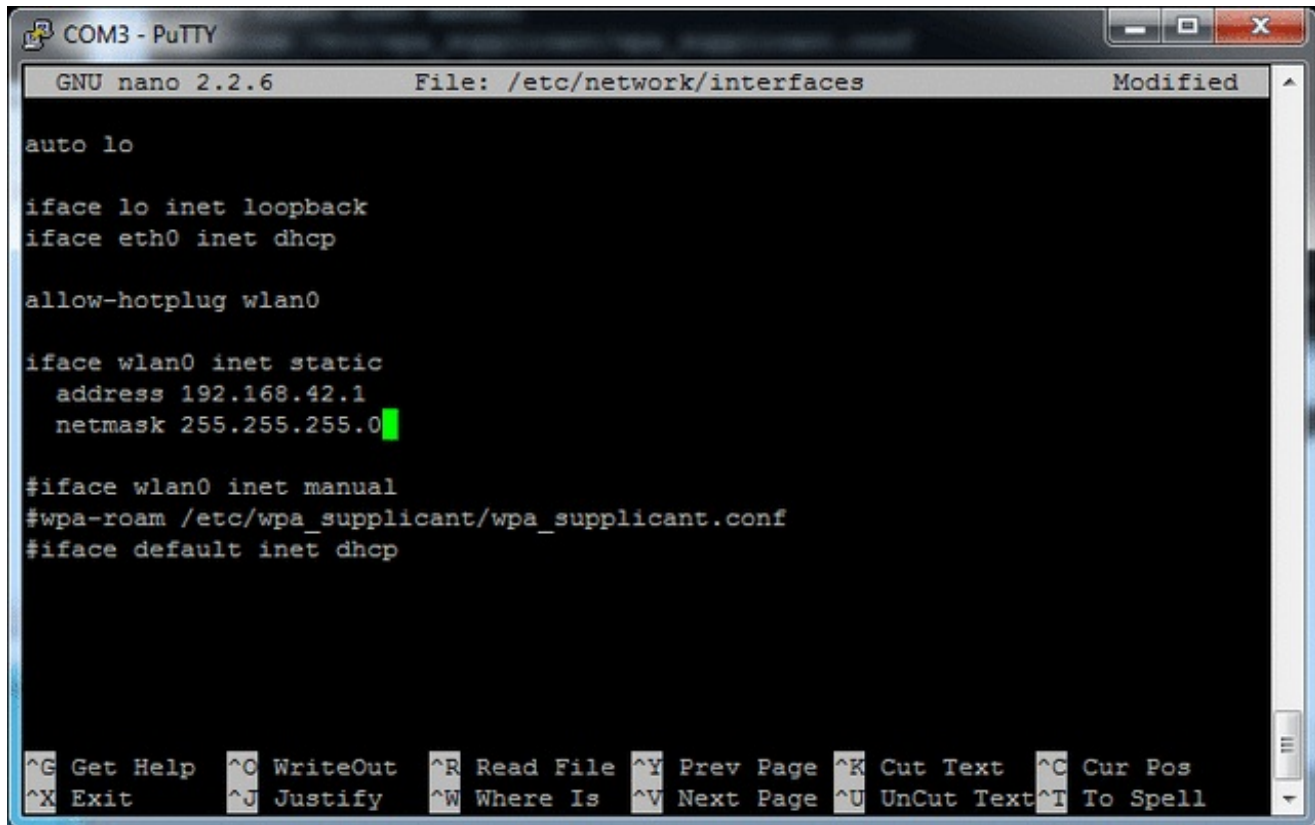
Find the line **auto wlan0** and add a **#** in front of the line, and in front of every line afterwards. If you don't have that line, just make sure it looks like the screenshot below in the end! Basically just remove any old **wlan0** configuration settings, we'll be changing them up

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



```
COM3 - PuTTY
GNU nano 2.2.6      File: /etc/network/interfaces      Modified
auto lo

iface lo inet loopback
iface eth0 inet dhcp

allow-hotplug wlan0

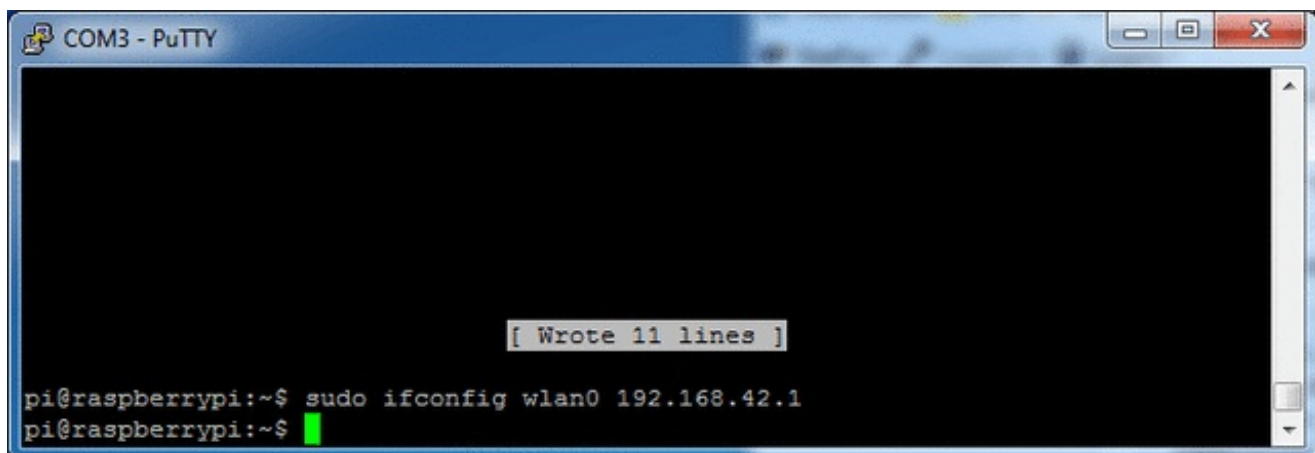
iface wlan0 inet static
    address 192.168.42.1
    netmask 255.255.255.0

#iface wlan0 inet manual
#wpa-roam /etc/wpa_supplicant/wpa_supplicant.conf
#iface default inet dhcp

^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
```

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**



```
COM3 - PuTTY

[ Wrote 11 lines ]

pi@raspberrypi:~$ sudo ifconfig wlan0 192.168.42.1
pi@raspberrypi:~$
```

## 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.

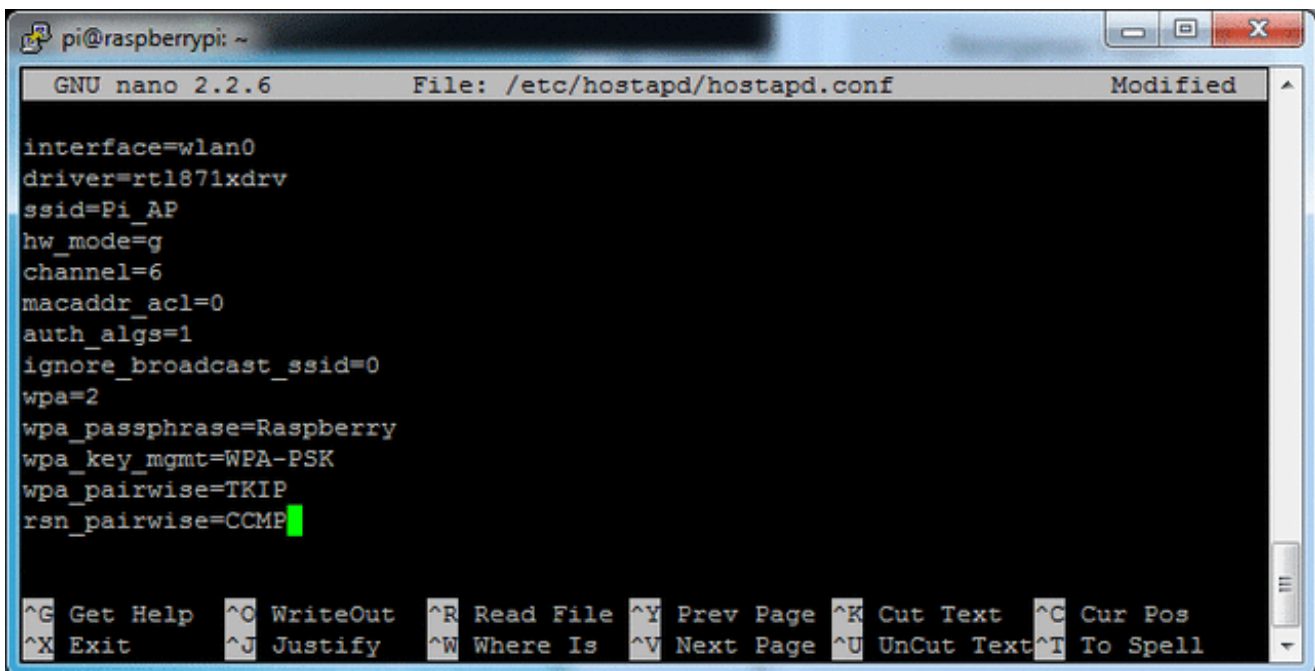


Create a new file by running **sudo nano /etc/hostapd/hostapd.conf**

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!

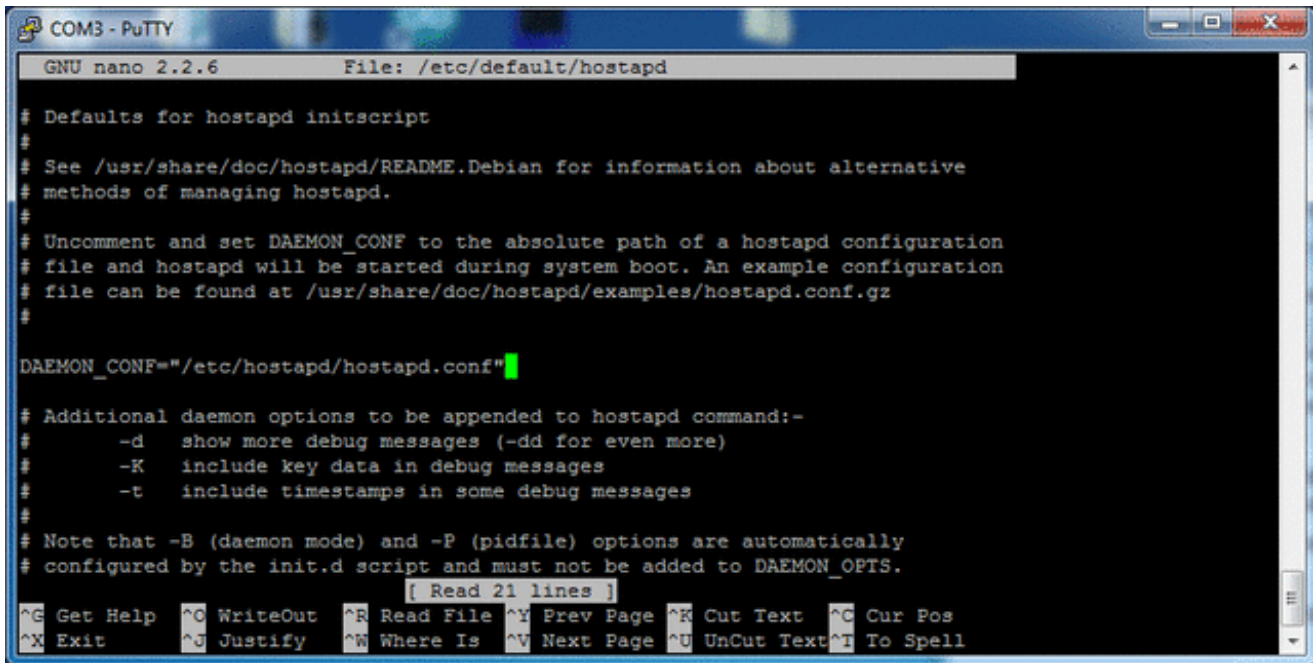
A screenshot of a terminal window on a Raspberry Pi. The terminal shows the nano text editor editing the file /etc/hostapd/hostapd.conf. The configuration text is as follows: interface=wlan0, driver=rtl871xdrv, ssid=Pi\_AP, hw\_mode=g, channel=6, macaddr\_acl=0, auth\_algs=1, ignore\_broadcast\_ssid=0, wpa=2, wpa\_passphrase=Raspberry, wpa\_key\_mgmt=WPA-PSK, wpa\_pairwise=TKIP, and rsn\_pairwise=CCMP. The cursor is at the end of the last line. The nano editor's status bar at the bottom shows various keyboard shortcuts like ^G Get Help, ^O WriteOut, ^R Read File, etc. The window title bar indicates the user is pi@raspberrypi and the file is /etc/hostapd/hostapd.conf.

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



```
COM3 - PuTTY
GNU nano 2.2.6      File: /etc/default/hostapd

# Defaults for hostapd initscript
#
# See /usr/share/doc/hostapd/README.Debian for information about alternative
# methods of managing hostapd.
#
# Uncomment and set DAEMON_CONF to the absolute path of a hostapd configuration
# file and hostapd will be started during system boot. An example configuration
# file can be found at /usr/share/doc/hostapd/examples/hostapd.conf.gz
#
DAEMON_CONF="/etc/hostapd/hostapd.conf"

# Additional daemon options to be appended to hostapd command:-
#
#   -d   show more debug messages (-dd for even more)
#   -K   include key data in debug messages
#   -t   include timestamps in some debug messages
#
# Note that -B (daemon mode) and -P (pidfile) options are automatically
# configured by the init.d script and must not be added to DAEMON_OPTS.
[ Read 21 lines ]
^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page ^U UnCut Text ^T To Spell
```

## 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

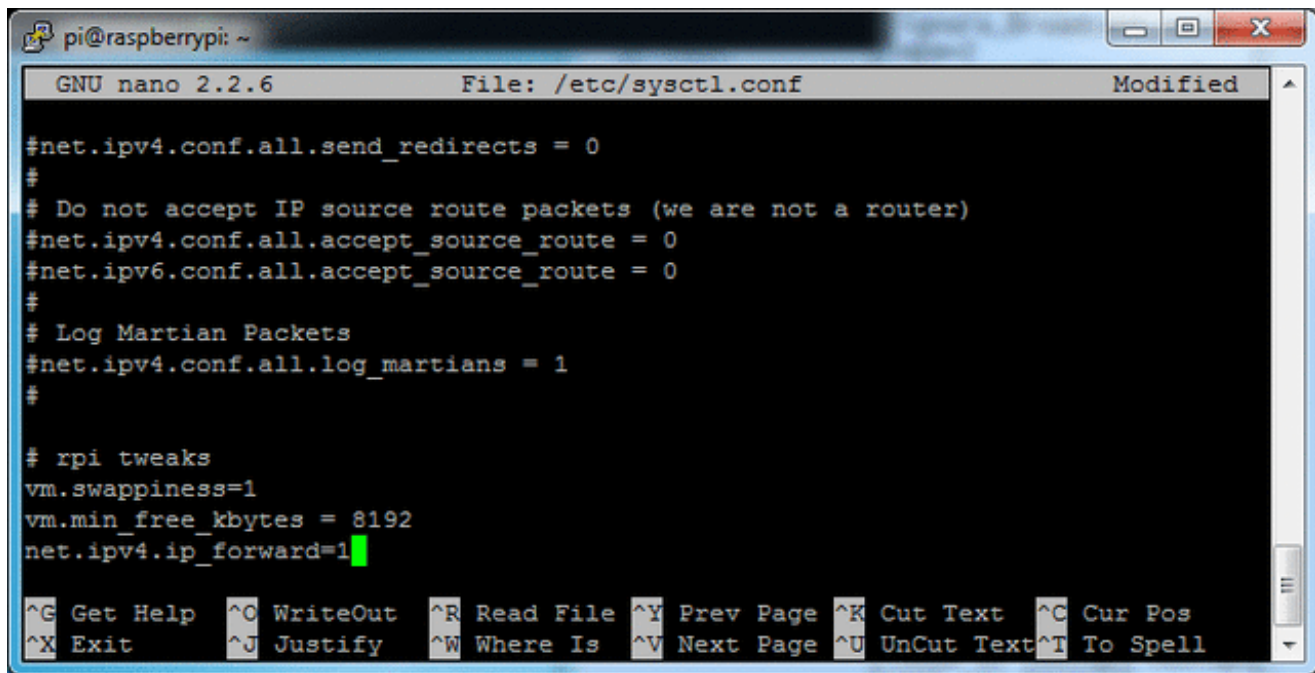
```
net.ipv4.ip_forward=1
```

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



```
pi@raspberrypi: ~
GNU nano 2.2.6      File: /etc/sysctl.conf      Modified

#net.ipv4.conf.all.send_redirects = 0
#
# Do not accept IP source route packets (we are not a router)
#net.ipv4.conf.all.accept_source_route = 0
#net.ipv6.conf.all.accept_source_route = 0
#
# Log Martian Packets
#net.ipv4.conf.all.log_martians = 1
#
# rpi tweaks
vm.swappiness=1
vm.min_free_kbytes = 8192
net.ipv4.ip_forward=1

^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
```

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

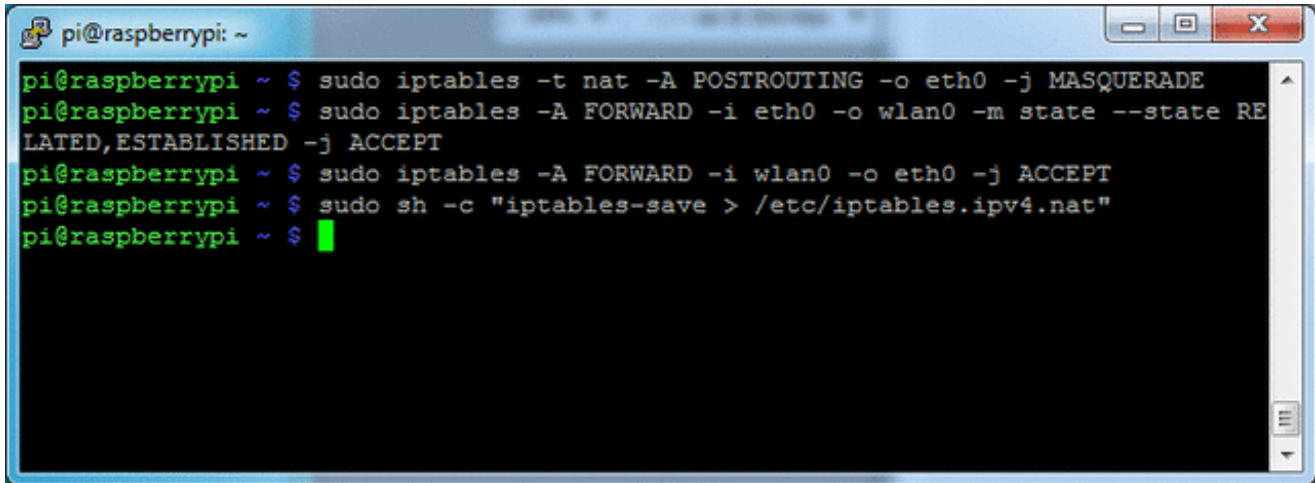
```
sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
sudo iptables -A FORWARD -i eth0 -o wlan0 -m state --state RELATED,ESTABLISHED -j ACCEPT
sudo iptables -A FORWARD -i wlan0 -o eth0 -j ACCEPT
```

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"
```

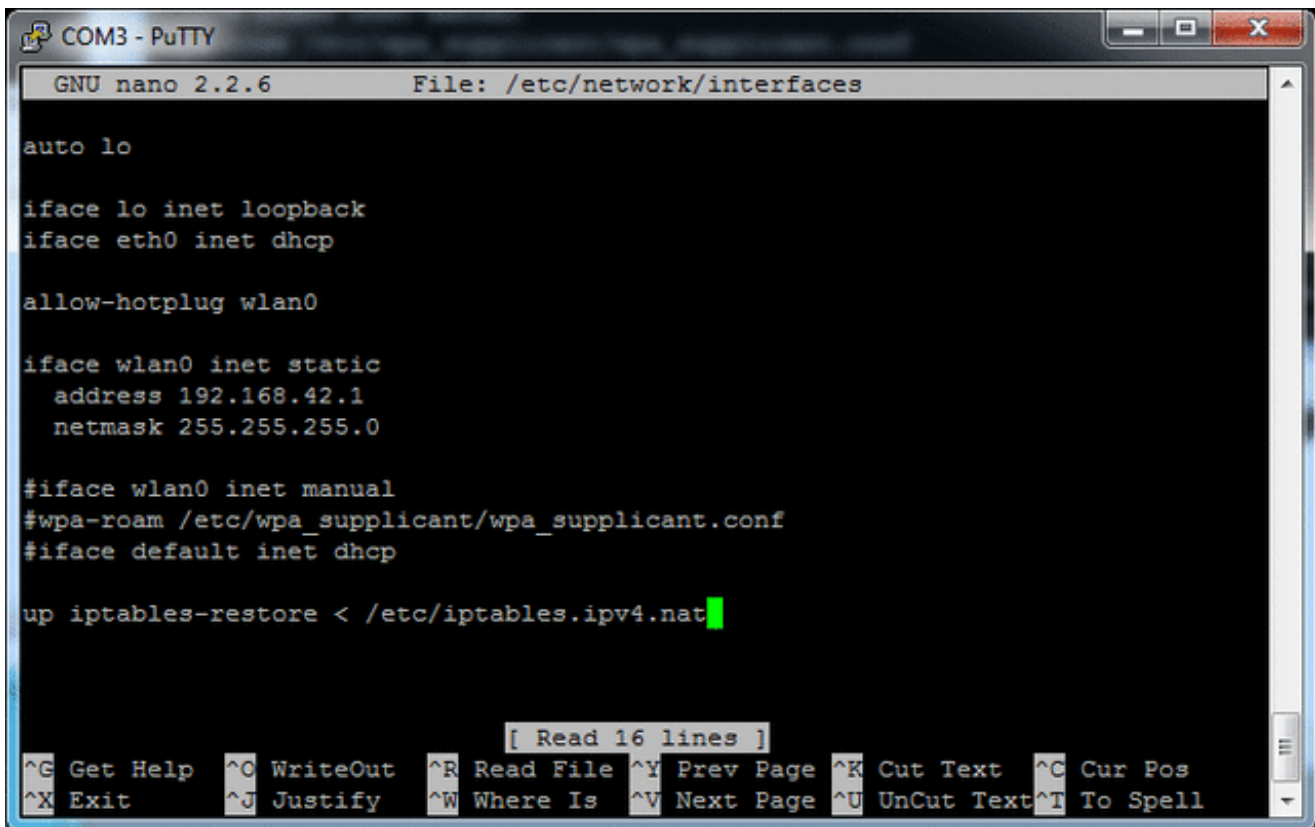
A terminal window titled 'pi@raspberrypi: ~' showing a series of iptables commands being executed. The commands are: 'sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE', 'sudo iptables -A FORWARD -i eth0 -o wlan0 -m state --state RELATED,ESTABLISHED -j ACCEPT', 'sudo iptables -A FORWARD -i wlan0 -o eth0 -j ACCEPT', and 'sudo sh -c "iptables-save > /etc/iptables.ipv4.nat"'. The prompt 'pi@raspberrypi ~ \$' is visible at the end of the last command.

```
pi@raspberrypi ~ $ sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
pi@raspberrypi ~ $ sudo iptables -A FORWARD -i eth0 -o wlan0 -m state --state RE
LATED,ESTABLISHED -j ACCEPT
pi@raspberrypi ~ $ sudo iptables -A FORWARD -i wlan0 -o eth0 -j ACCEPT
pi@raspberrypi ~ $ sudo sh -c "iptables-save > /etc/iptables.ipv4.nat"
pi@raspberrypi ~ $
```

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

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

to the very end

A nano editor window titled 'COM3 - PuTTY' showing the file '/etc/network/interfaces'. The file contains network configuration for 'lo', 'eth0', and 'wlan0'. At the bottom, the line 'up iptables-restore < /etc/iptables.ipv4.nat' has been added. The nano status bar at the bottom shows '[ Read 16 lines ]' and various keyboard shortcuts.

```
GNU nano 2.2.6      File: /etc/network/interfaces

auto lo

iface lo inet loopback
iface eth0 inet dhcp

allow-hotplug wlan0

iface wlan0 inet static
    address 192.168.42.1
    netmask 255.255.255.0

#iface wlan0 inet manual
#wpa-roam /etc/wpa_supplicant/wpa_supplicant.conf
#iface default inet dhcp

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

## 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
```

```
pi@raspberrypi: ~  
pi@raspberrypi ~ $ wget http://www.adafruit.com/downloads/adafruit_hostapd.zip  
--2013-06-12 16:06:50-- http://www.adafruit.com/downloads/adafruit_hostapd.zip  
Resolving www.adafruit.com (www.adafruit.com)... 207.58.139.247  
Connecting to www.adafruit.com (www.adafruit.com)|207.58.139.247|:80... connecte  
d.  
HTTP request sent, awaiting response... 200 OK  
Length: 709582 (693K) [application/zip]  
Saving to: `adafruit_hostapd.zip'  
  
100%[=====>] 709,582      3.65M/s   in 0.2s  
2013-06-12 16:06:50 (3.65 MB/s) - `adafruit_hostapd.zip' saved [709582/709582]  
  
pi@raspberrypi ~ $ unzip adafruit_hostapd.zip  
Archive:  adafruit_hostapd.zip  
  inflating: hostapd  
pi@raspberrypi ~ $ sudo mv /usr/sbin/hostapd /usr/sbin/hostapd.ORIG  
pi@raspberrypi ~ $ sudo mv hostapd /usr/sbin  
pi@raspberrypi ~ $ sudo chmod 755 /usr/sbin/hostapd  
pi@raspberrypi ~ $
```

## 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.



```
pi@raspberrypi: ~  
pi@raspberrypi ~ $ sudo mv /usr/sbin/hostapd /usr/sbin/hostapd.ORIG  
pi@raspberrypi ~ $ sudo mv hostapd /usr/sbin  
pi@raspberrypi ~ $ sudo chmod 755 /usr/sbin/hostapd  
pi@raspberrypi ~ $ sudo /usr/sbin/hostapd /etc/hostapd/hostapd.conf  
Configuration file: /etc/hostapd/hostapd.conf  
drv->ifindex=3  
l2_sock_recv==l2_sock_xmit=0x0x1fb638  
+rtl871x_sta_deauth_ops, ff:ff:ff:ff:ff:ff is deauth, reason=2  
rtl871x_set_key_ops  
rtl871x_set_key_ops  
rtl871x_set_key_ops  
rtl871x_set_key_ops  
Using interface wlan0 with hwaddr 00:e0:4c:09:3b:f8 and ssid 'Pi_AP'  
rtl871x_set_wps_assoc_resp_ie  
rtl871x_set_wps_beacon_ie  
rtl871x_set_wps_probe_resp_ie  
rtl871x_set_key_ops  
rtl871x_set_beacon_ops  
rtl871x_set_hidden_ssid_ops
```



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

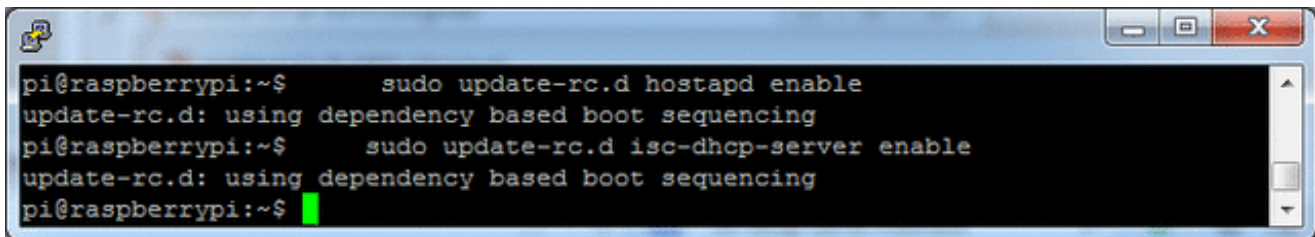
```
sudo service hostapd start  
sudo service isc-dhcp-server start
```

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
```

A terminal window with a blue title bar and standard window controls. The terminal shows the user 'pi' at the 'raspberrypi' prompt. They run 'sudo update-rc.d hostapd enable', which outputs 'update-rc.d: using dependency based boot sequencing'. Then they run 'sudo update-rc.d isc-dhcp-server enable', which also outputs 'update-rc.d: using dependency based boot sequencing'. The prompt returns to 'pi@raspberrypi:~\$' with a green cursor.

```
pi@raspberrypi:~$ sudo update-rc.d hostapd enable  
update-rc.d: using dependency based boot sequencing  
pi@raspberrypi:~$ sudo update-rc.d isc-dhcp-server enable  
update-rc.d: using dependency based boot sequencing  
pi@raspberrypi:~$
```

## 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**)

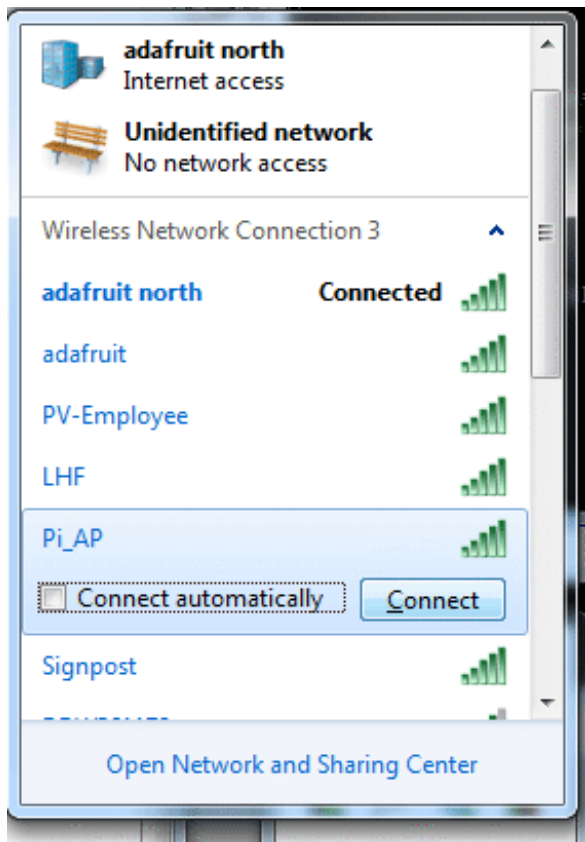


# Connect and Test

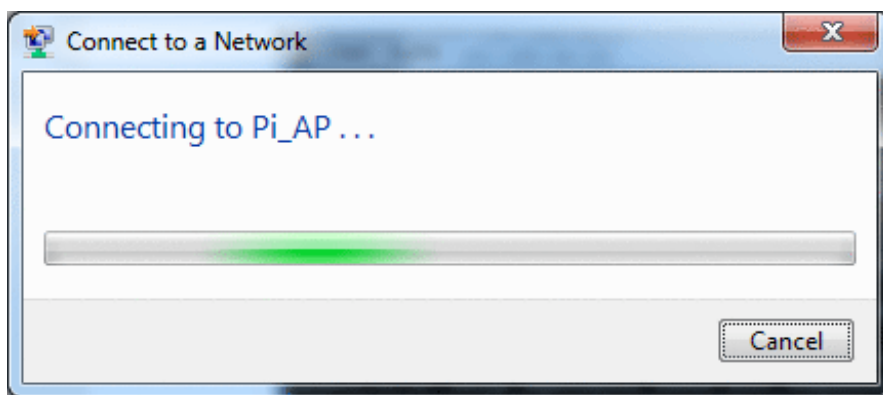
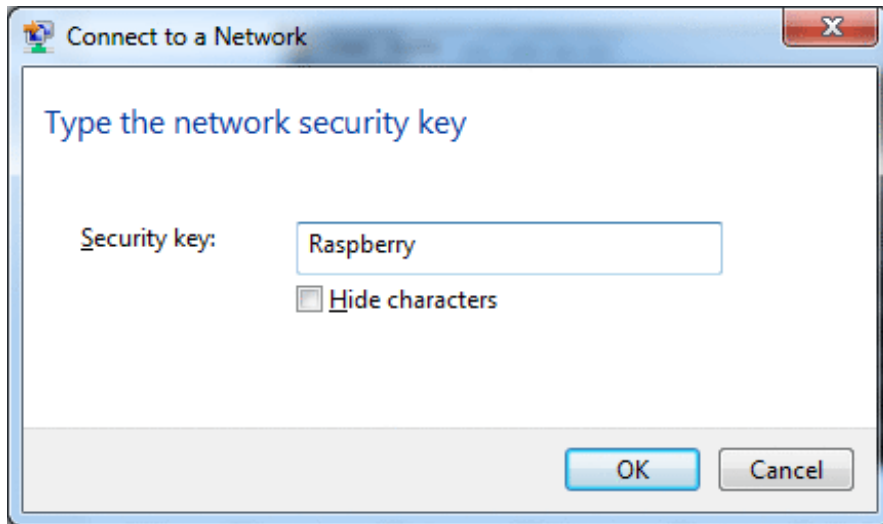
Now that we have the software installed on a Pi, it's time to connect to it and test the connection. I'm using a Windows computer but any kind should work fine

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

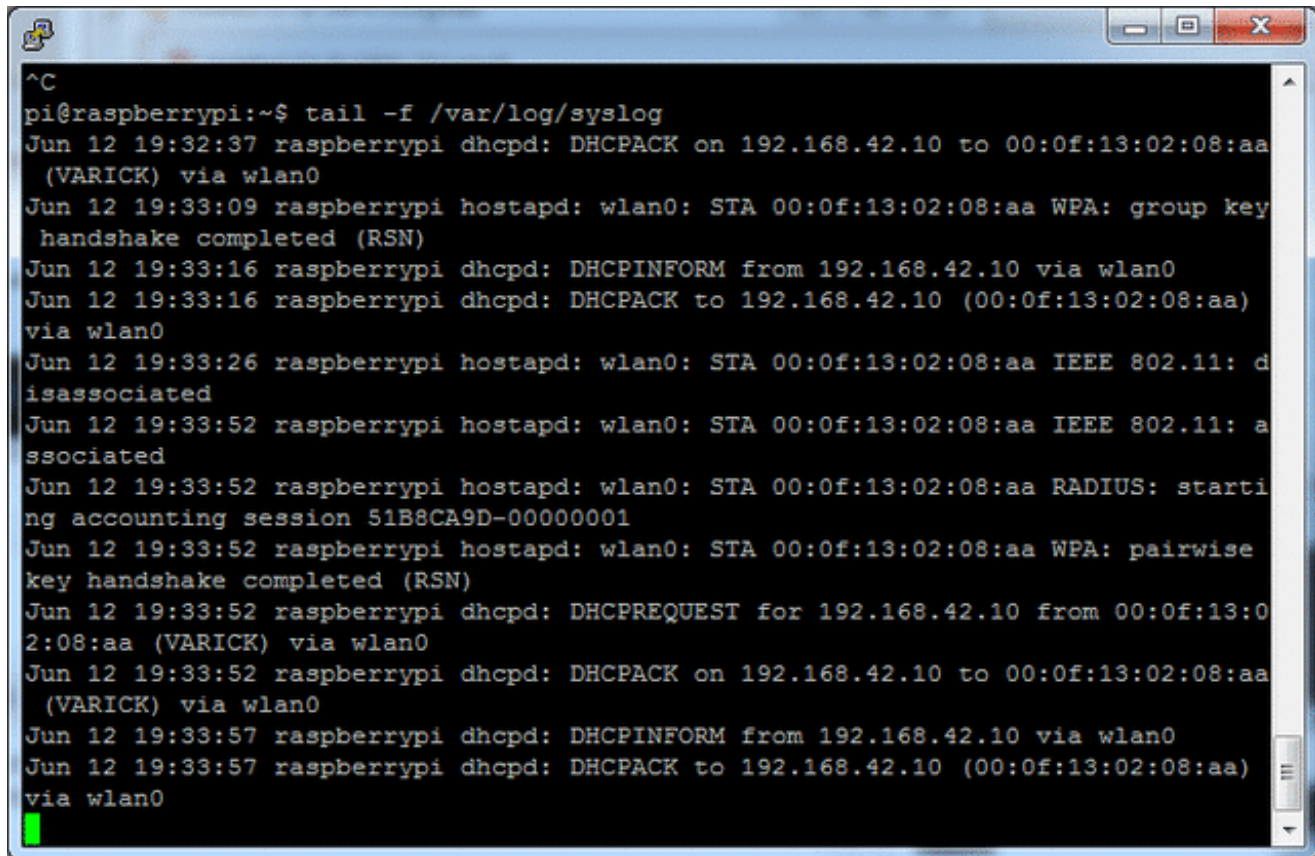


Enter the WPA key you specified in the previous step



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**

A screenshot of a terminal window on a Raspberry Pi. The window has a title bar with standard Linux window controls (minimize, maximize, close). The terminal text shows the user pressing Ctrl+C (^C) and then running the command 'tail -f /var/log/syslog'. The output displays a series of network-related log entries from the system log, including DHCP transactions and WPA handshakes on the wlan0 interface. The logs are timestamped with dates in June 2012. A green cursor is visible at the bottom left of the terminal area.

```
^C
pi@raspberrypi:~$ tail -f /var/log/syslog
Jun 12 19:32:37 raspberrypi dhcpd: DHCPACK on 192.168.42.10 to 00:0f:13:02:08:aa
(VARICK) via wlan0
Jun 12 19:33:09 raspberrypi hostapd: wlan0: STA 00:0f:13:02:08:aa WPA: group key
handshake completed (RSN)
Jun 12 19:33:16 raspberrypi dhcpd: DHCPINFORM from 192.168.42.10 via wlan0
Jun 12 19:33:16 raspberrypi dhcpd: DHCPACK to 192.168.42.10 (00:0f:13:02:08:aa)
via wlan0
Jun 12 19:33:26 raspberrypi hostapd: wlan0: STA 00:0f:13:02:08:aa IEEE 802.11: d
isassociated
Jun 12 19:33:52 raspberrypi hostapd: wlan0: STA 00:0f:13:02:08:aa IEEE 802.11: a
ssociated
Jun 12 19:33:52 raspberrypi hostapd: wlan0: STA 00:0f:13:02:08:aa RADIUS: starti
ng accounting session 51B8CA9D-00000001
Jun 12 19:33:52 raspberrypi hostapd: wlan0: STA 00:0f:13:02:08:aa WPA: pairwise
key handshake completed (RSN)
Jun 12 19:33:52 raspberrypi dhcpd: DHCPREQUEST for 192.168.42.10 from 00:0f:13:0
2:08:aa (VARICK) via wlan0
Jun 12 19:33:52 raspberrypi dhcpd: DHCPACK on 192.168.42.10 to 00:0f:13:02:08:aa
(VARICK) via wlan0
Jun 12 19:33:57 raspberrypi dhcpd: DHCPINFORM from 192.168.42.10 via wlan0
Jun 12 19:33:57 raspberrypi dhcpd: DHCPACK to 192.168.42.10 (00:0f:13:02:08:aa)
via wlan0
█
```

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

```
C:\Windows\system32\cmd.exe
C:\Users\ladyada>ipconfig

Windows IP Configuration

Wireless LAN adapter Wireless Network Connection 3:

    Connection-specific DNS Suffix  . : local
    Link-local IPv6 Address . . . . . : fe80::e9:e1c:1ef9:7a0b%30
    IPv4 Address. . . . . : 192.168.42.10
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 192.168.42.1

Ethernet adapter Local Area Connection:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

Ethernet adapter VirtualBox Host-Only Network:

    Connection-specific DNS Suffix  . :
    Link-local IPv6 Address . . . . . : fe80::e91e:be0d:6eb9:b792%21
    IPv4 Address. . . . . : 192.168.56.1
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . :

Tunnel adapter isatap.{6E34487D-1AB2-46BD-A955-5D6945E39890}:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

Tunnel adapter isatap.{F52288E5-61A3-464B-92B6-20E0FA8E2152}:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

Tunnel adapter Teredo Tunneling Pseudo-Interface:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

Tunnel adapter isatap.{A76BF87D-040E-4B0B-8099-A50EBA854757}:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

Tunnel adapter isatap.local:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . : local

C:\Users\ladyada>
```

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 **dhcpcd** (more likely)

```
C:\Windows\system32\cmd.exe
^C
C:\Users\ladyada>ping 192.168.42.1

Pinging 192.168.42.1 with 32 bytes of data:
Reply from 192.168.42.1: bytes=32 time=29ms TTL=64
Reply from 192.168.42.1: bytes=32 time=49ms TTL=64
Reply from 192.168.42.1: bytes=32 time=11ms TTL=64
Reply from 192.168.42.1: bytes=32 time=317ms TTL=64

Ping statistics for 192.168.42.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 11ms, Maximum = 317ms, Average = 101ms

C:\Users\ladyada>_
```

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

```
C:\Windows\system32\cmd.exe
^C
C:\Users\ladyada>ping 8.8.8.8

Pinging 8.8.8.8 with 32 bytes of data:
Reply from 8.8.8.8: bytes=32 time=167ms TTL=50
Reply from 8.8.8.8: bytes=32 time=142ms TTL=50
Reply from 8.8.8.8: bytes=32 time=327ms TTL=50

Ping statistics for 8.8.8.8:
    Packets: Sent = 3, Received = 3, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 142ms, Maximum = 327ms, Average = 212ms

Control-C
^C
C:\Users\ladyada>_
```

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

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