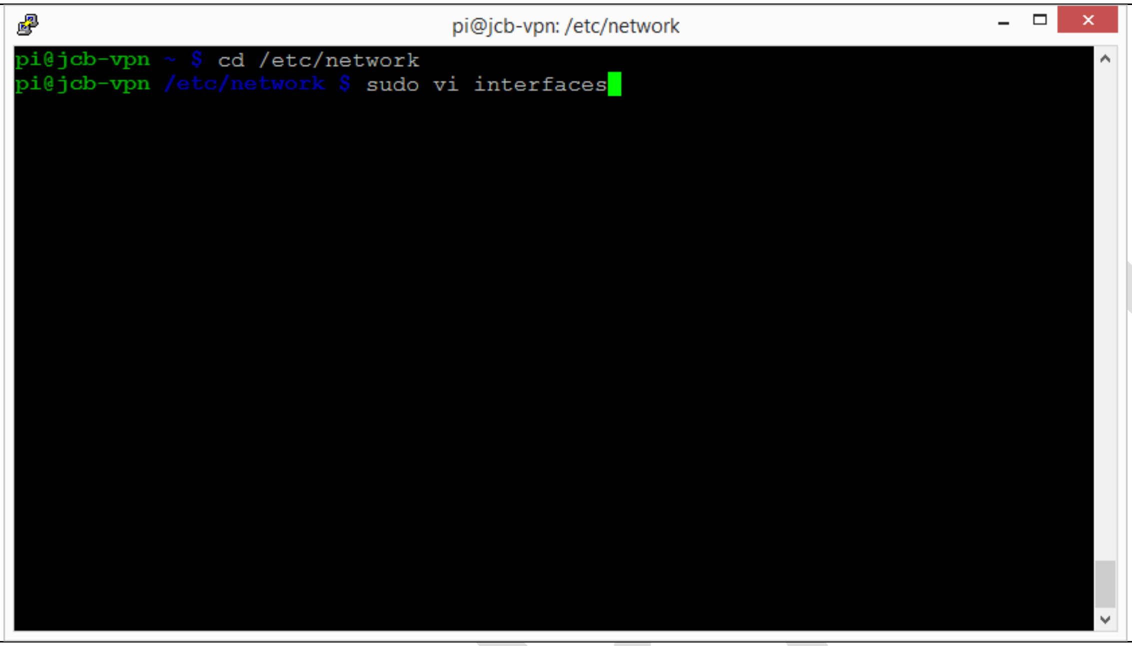
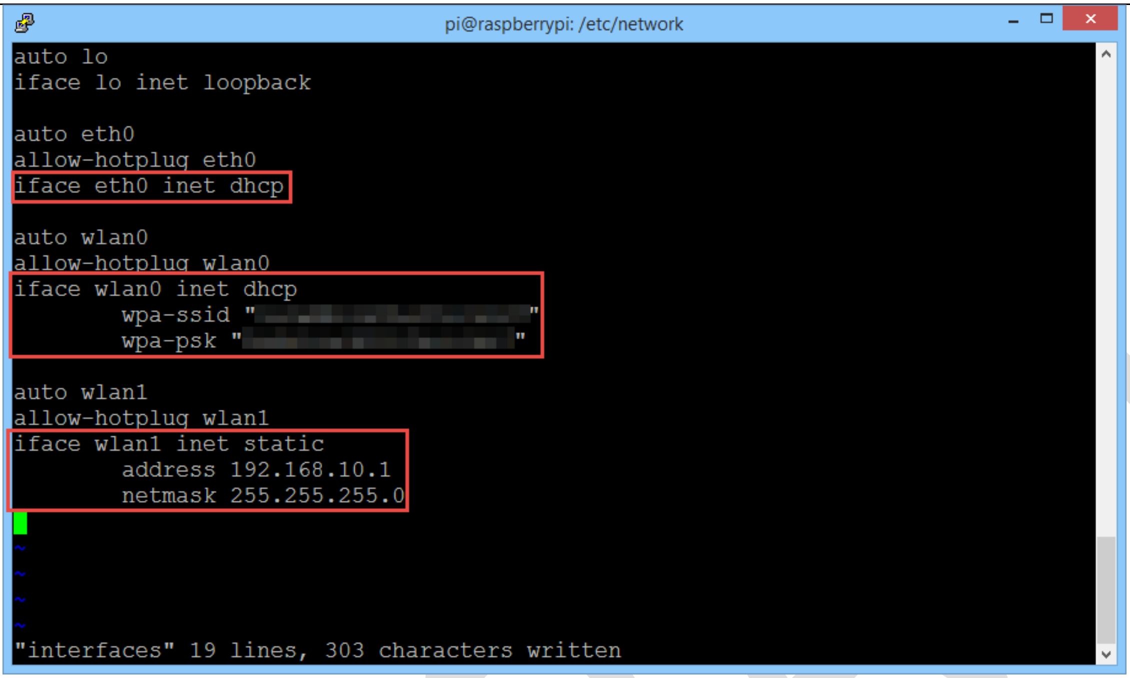
Configure Raspberry Pi as a Hotspot

1	
	pi@jcb-vpn: /etc/network
	`pi@jcb-vpn ~ $ cd /etc/network`
	`pi@jcb-vpn /etc/network $ sudo vi interfaces`
		Execute “cd /etc/network”.
		Execute “sudo vi interfaces”.

2



```
auto lo
iface lo inet loopback

auto eth0
allow-hotplug eth0
iface eth0 inet dhcp

auto wlan0
allow-hotplug wlan0
iface wlan0 inet dhcp
        wpa-ssid "[redacted]"
        wpa-psk "[redacted]"

auto wlan1
allow-hotplug wlan1
iface wlan1 inet static
        address 192.168.10.1
        netmask 255.255.255.0

~
~
~
~
"interfaces" 19 lines, 303 characters written
```

Set the network options for interface wlan0.

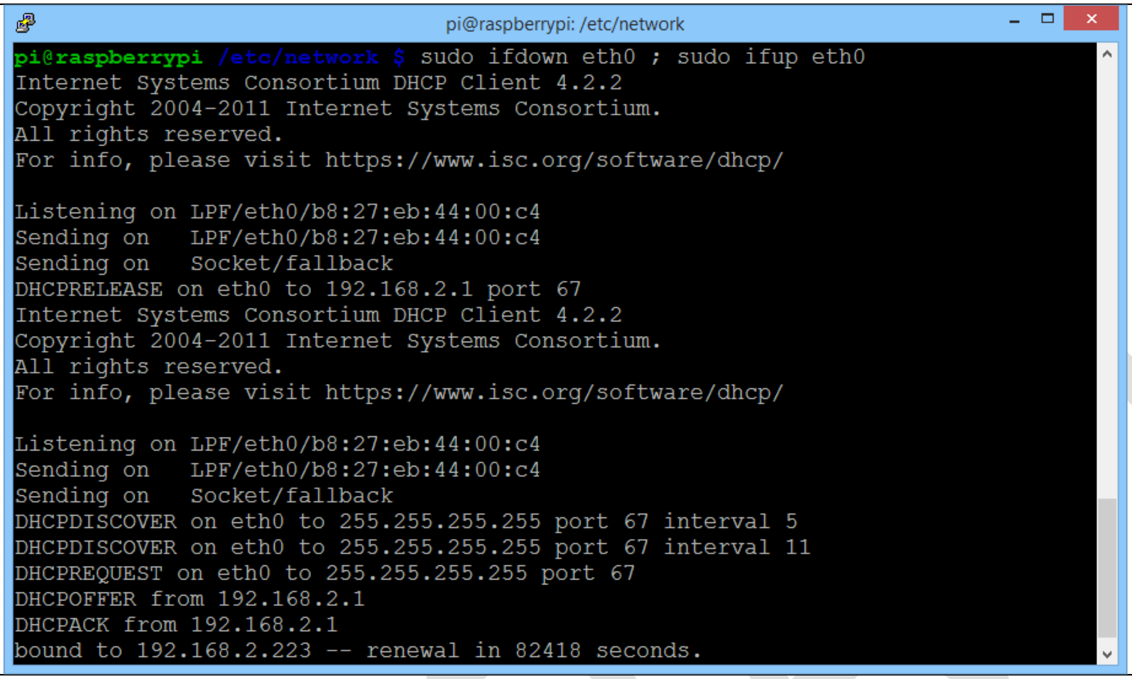
Save the file and exit the editor.

This step configures the various network interfaces:
eth0 – configured for DHCP to allow the Raspberry Pi to become a node on most networks. This helps with management of the Raspberry Pi while the remaining configuration is in progress.

wlan0: allows the Raspberry Pi to connect to the public wireless network or the wireless network at your home. “wlan0” will be the network interface to the “untrusted” wireless network.

wlan1: configures the second wireless as a private hotspot that will provide anonymity services.

3



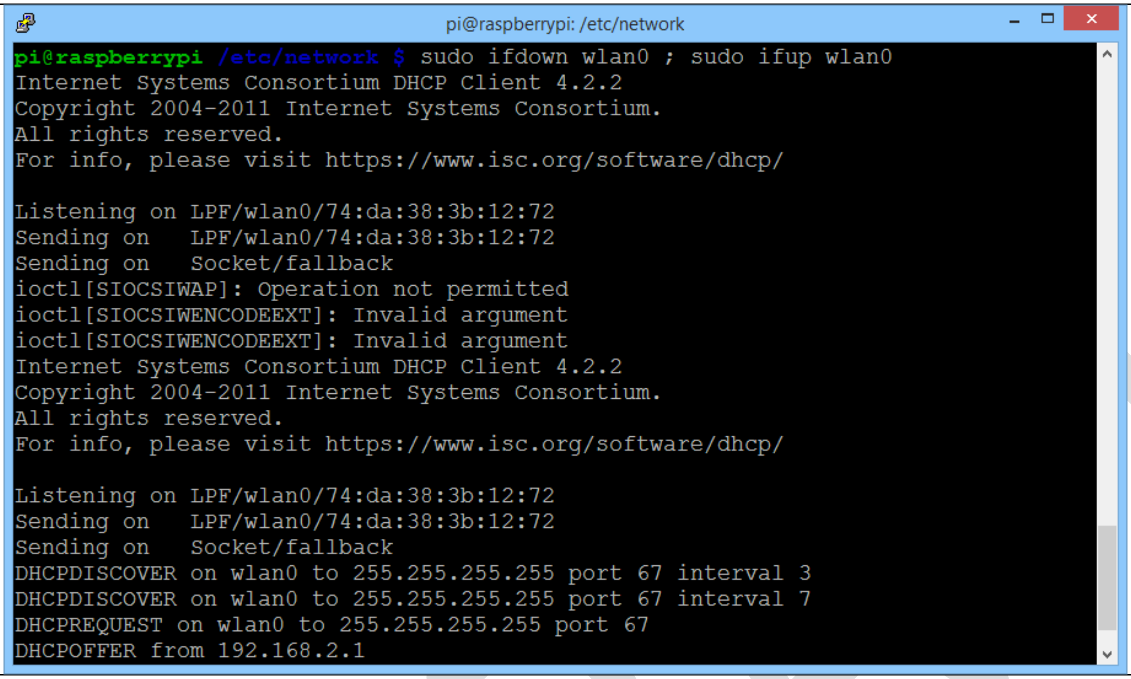
```
pi@raspberrypi /etc/network $ sudo ifdown eth0 ; sudo ifup eth0
Internet Systems Consortium DHCP Client 4.2.2
Copyright 2004-2011 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/

Listening on LPF/eth0/b8:27:eb:44:00:c4
Sending on   LPF/eth0/b8:27:eb:44:00:c4
Sending on   Socket/fallback
DHCPRELEASE on eth0 to 192.168.2.1 port 67
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/eth0/b8:27:eb:44:00:c4
Sending on   LPF/eth0/b8:27:eb:44:00:c4
Sending on   Socket/fallback
DHCPDISCOVER on eth0 to 255.255.255.255 port 67 interval 5
DHCPDISCOVER on eth0 to 255.255.255.255 port 67 interval 11
DHCPREQUEST on eth0 to 255.255.255.255 port 67
DHCPOFFER from 192.168.2.1
DHCPACK from 192.168.2.1
bound to 192.168.2.223 -- renewal in 82418 seconds.
```

Execute “sudo ifdown eth0 ; sudo ifup eth0”.

Execute “sudo ifdown wlan0 ; sudo ifup wlan0”.

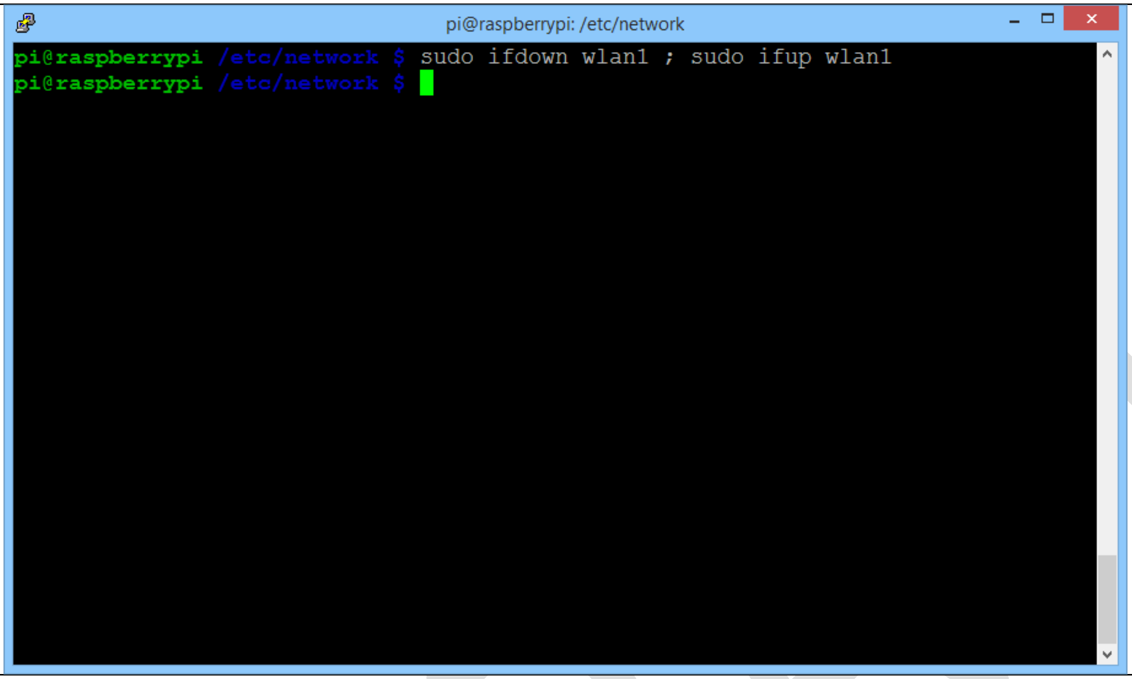
Execute “sudo ifdown wlan1 ; sudo ifup wlan1”.

This step recycles all the network interfaces to reflect the updated configuration.



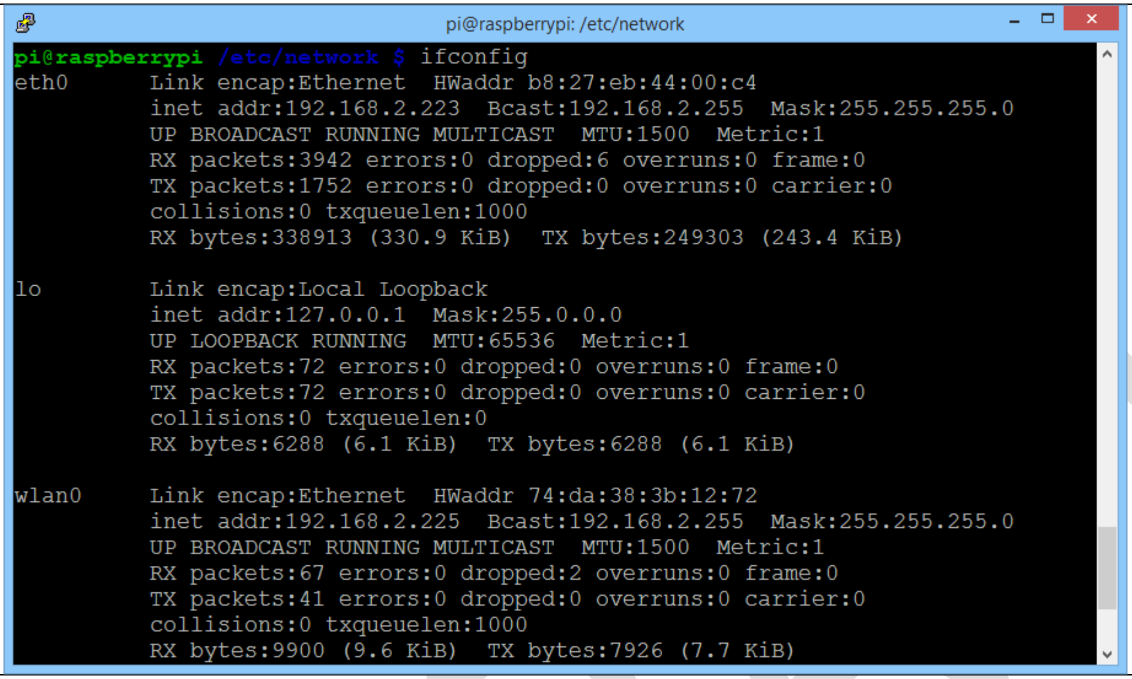
pi@raspberrypi: /etc/network

```
pi@raspberrypi /etc/network $ sudo ifdown wlan0 ; sudo ifup 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/wlan0/74:da:38:3b:12:72
Sending on   LPF/wlan0/74:da:38:3b:12:72
Sending on   Socket/fallback
ioctl[SIOCSIWAP]: Operation not permitted
ioctl[SIOCSIWENCODEEXT]: Invalid argument
ioctl[SIOCSIWENCODEEXT]: Invalid argument
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/wlan0/74:da:38:3b:12:72
Sending on   LPF/wlan0/74:da:38:3b:12:72
Sending on   Socket/fallback
DHCPDISCOVER on wlan0 to 255.255.255.255 port 67 interval 3
DHCPDISCOVER on wlan0 to 255.255.255.255 port 67 interval 7
DHCPREQUEST on wlan0 to 255.255.255.255 port 67
DHCPOFFER from 192.168.2.1
```



```
pi@raspberrypi /etc/network $ sudo ifdown wlan1 ; sudo ifup wlan1
pi@raspberrypi /etc/network $
```

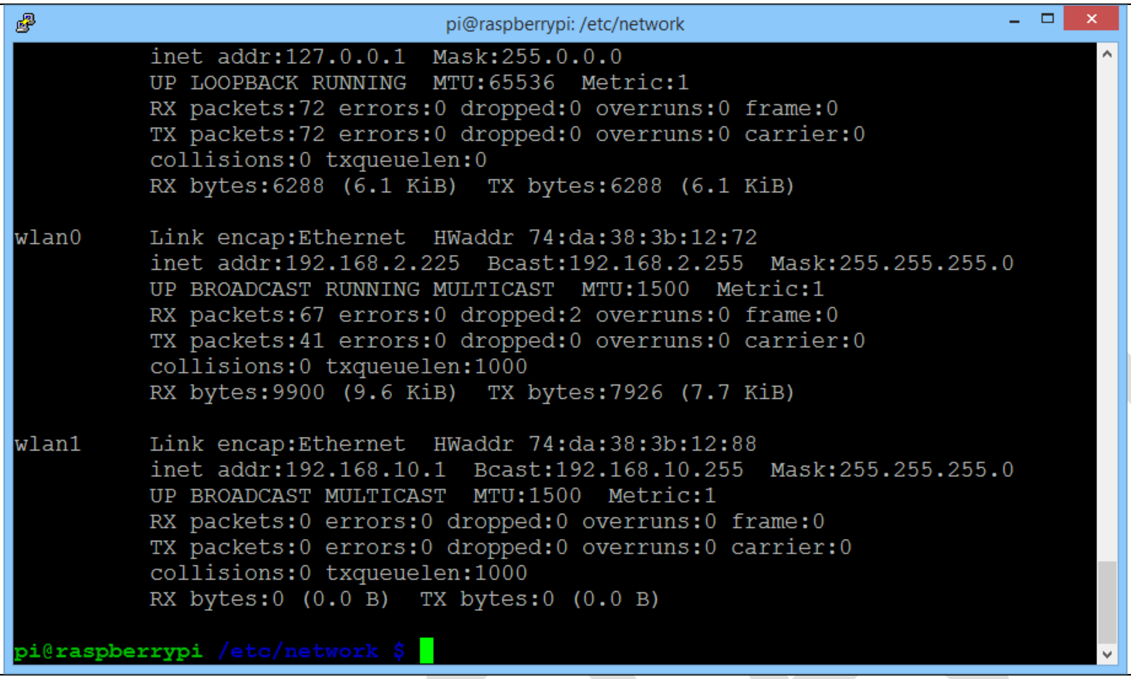
4



```
pi@raspberrypi /etc/network $ ifconfig
eth0      Link encap:Ethernet  HWaddr b8:27:eb:44:00:c4
          inet addr:192.168.2.223  Bcast:192.168.2.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3942 errors:0 dropped:6 overruns:0 frame:0
          TX packets:1752 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:338913 (330.9 KiB)  TX bytes:249303 (243.4 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:72 errors:0 dropped:0 overruns:0 frame:0
          TX packets:72 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:6288 (6.1 KiB)  TX bytes:6288 (6.1 KiB)

wlan0     Link encap:Ethernet  HWaddr 74:da:38:3b:12:72
          inet addr:192.168.2.225  Bcast:192.168.2.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:67 errors:0 dropped:2 overruns:0 frame:0
          TX packets:41 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:9900 (9.6 KiB)  TX bytes:7926 (7.7 KiB)
```

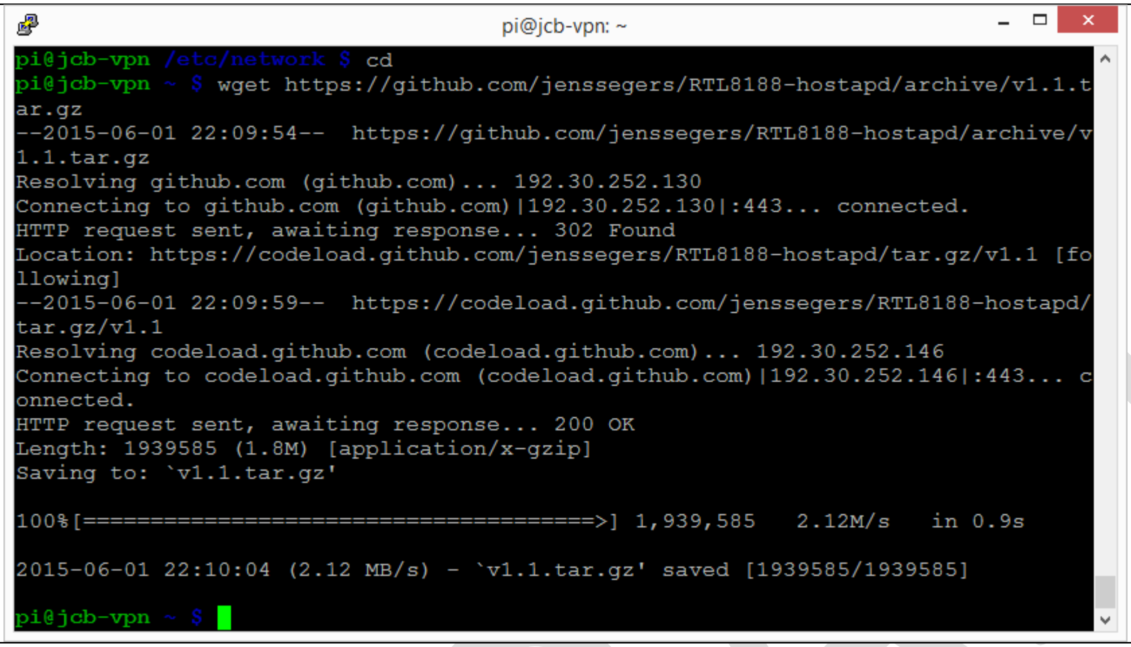
Execute “ifconfig”.

Confirm the network interfaces have the assigned IP address. In particular, eth0 should have an IP address on the wired network, wlan0 should have an IP address on the wireless network, and wlan1 should be assigned 192.168.10.1.



```
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:72 errors:0 dropped:0 overruns:0 frame:0
          TX packets:72 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:6288 (6.1 KiB)  TX bytes:6288 (6.1 KiB)

wlan0     Link encap:Ethernet  HWaddr 74:da:38:3b:12:72
          inet addr:192.168.2.225  Bcast:192.168.2.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:67 errors:0 dropped:2 overruns:0 frame:0
          TX packets:41 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:9900 (9.6 KiB)  TX bytes:7926 (7.7 KiB)

wlan1     Link encap:Ethernet  HWaddr 74:da:38:3b:12:88
          inet addr:192.168.10.1  Bcast:192.168.10.255  Mask:255.255.255.0
          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 /etc/network $
```

<table>
<tr>
<td>5</td>
<td>



```
pi@jcb-vpn /etc/network $ cd
pi@jcb-vpn ~ $ wget https://github.com/jenssegers/RTL8188-hostapd/archive/v1.1.t
ar.gz
--2015-06-01 22:09:54--  https://github.com/jenssegers/RTL8188-hostapd/archive/v
1.1.tar.gz
Resolving github.com (github.com)... 192.30.252.130
Connecting to github.com (github.com)|192.30.252.130|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://codeload.github.com/jenssegers/RTL8188-hostapd/tar.gz/v1.1 [fo
llowing]
--2015-06-01 22:09:59--  https://codeload.github.com/jenssegers/RTL8188-hostapd/
tar.gz/v1.1
Resolving codeload.github.com (codeload.github.com)... 192.30.252.146
Connecting to codeload.github.com (codeload.github.com)|192.30.252.146|:443... c
onnected.
HTTP request sent, awaiting response... 200 OK
Length: 1939585 (1.8M) [application/x-gzip]
Saving to: `v1.1.tar.gz'

100%[======================================>] 1,939,585   2.12M/s   in 0.9s

2015-06-01 22:10:04 (2.12 MB/s) - `v1.1.tar.gz' saved [1939585/1939585]

pi@jcb-vpn ~ $ 
```

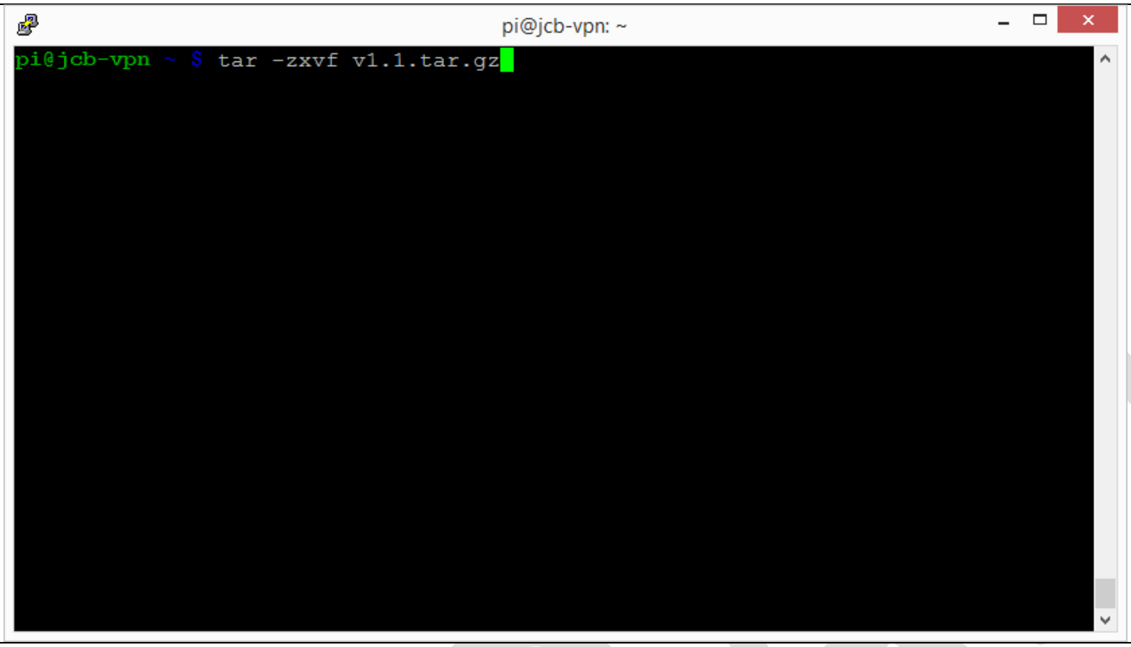
</td>
<td>

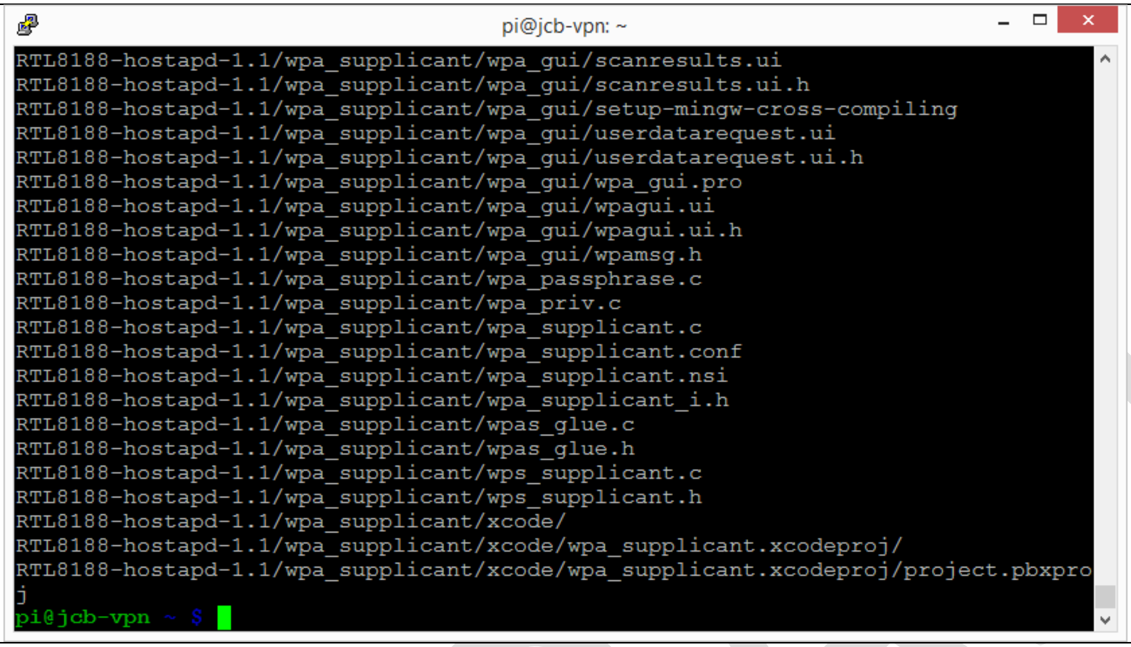
Execute “cd”.

Execute “wget https://github.com/jenssegers/RTL8188-hostapd/archive/v1.1.tar.gz”.

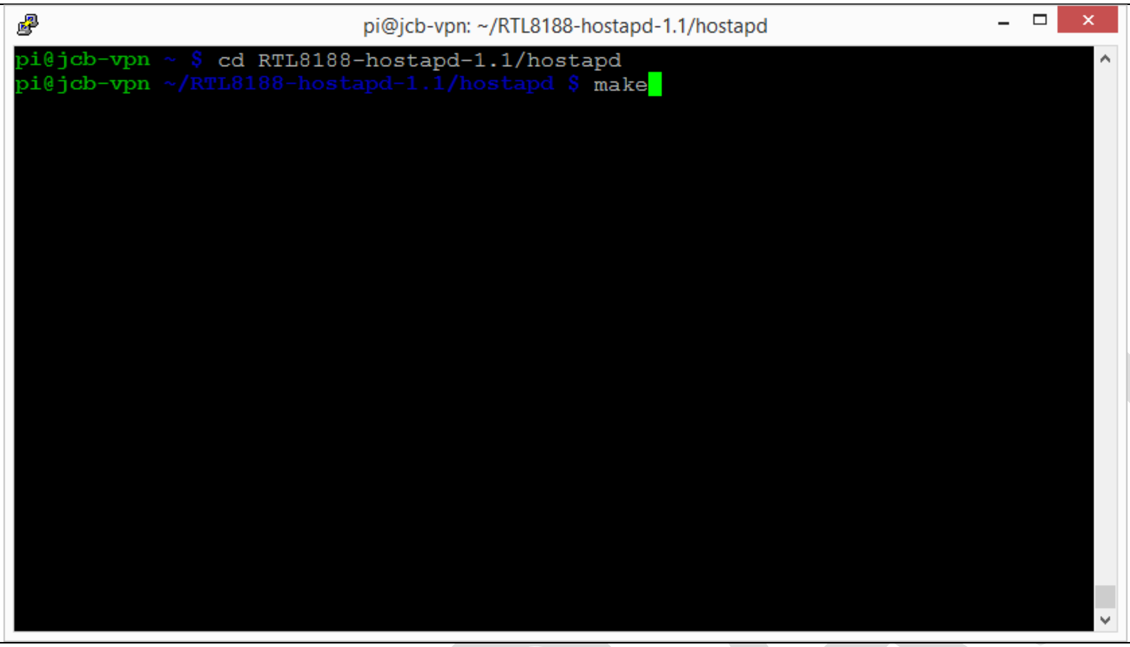
This step builds a version of hostapd that works with the Edimax cads. Hostapd allows the Raspberry Pi to be configured as a hotspot.

</td>
</tr>
</table>

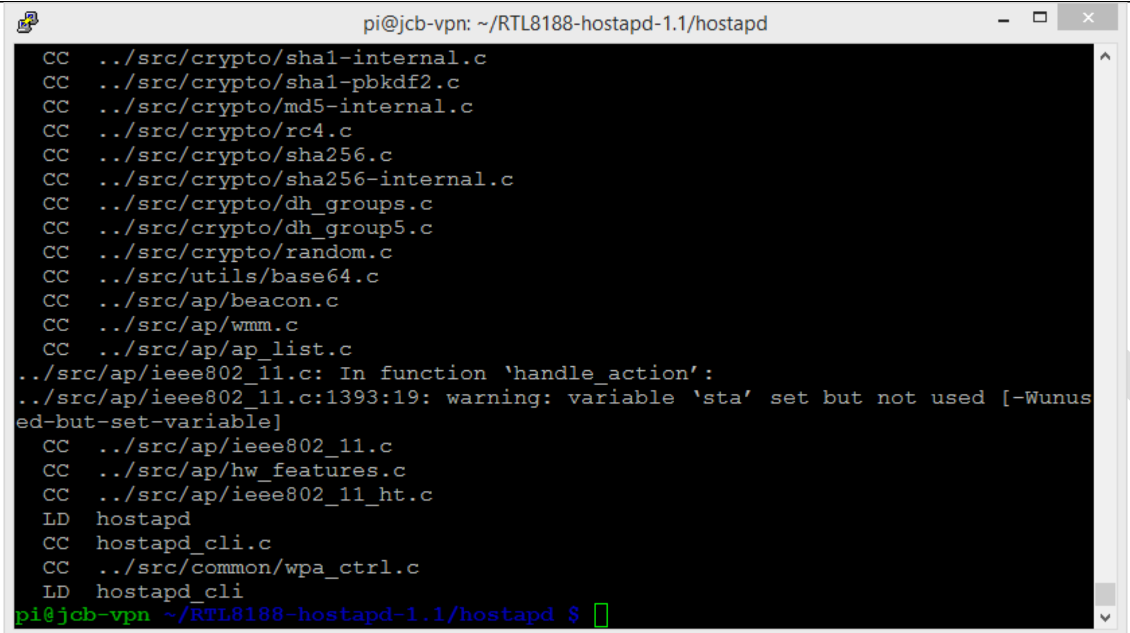
6	 `pi@jcb-vpn ~ $ tar -zxvf v1.1.tar.gz`	Execute “tar –zxvf v1.1.tar.gz”.

7		Confirm the file properly unpacks.

```
RTL8188-hostapd-1.1/wpa_supplicant/wpa_gui/scanresults.ui
RTL8188-hostapd-1.1/wpa_supplicant/wpa_gui/scanresults.ui.h
RTL8188-hostapd-1.1/wpa_supplicant/wpa_gui/setup-mingw-cross-compiling
RTL8188-hostapd-1.1/wpa_supplicant/wpa_gui/userdatarequest.ui
RTL8188-hostapd-1.1/wpa_supplicant/wpa_gui/userdatarequest.ui.h
RTL8188-hostapd-1.1/wpa_supplicant/wpa_gui/wpa_gui.pro
RTL8188-hostapd-1.1/wpa_supplicant/wpa_gui/wpagui.ui
RTL8188-hostapd-1.1/wpa_supplicant/wpa_gui/wpagui.ui.h
RTL8188-hostapd-1.1/wpa_supplicant/wpa_gui/wpamsg.h
RTL8188-hostapd-1.1/wpa_supplicant/wpa_passphrase.c
RTL8188-hostapd-1.1/wpa_supplicant/wpa_priv.c
RTL8188-hostapd-1.1/wpa_supplicant/wpa_supplicant.c
RTL8188-hostapd-1.1/wpa_supplicant/wpa_supplicant.conf
RTL8188-hostapd-1.1/wpa_supplicant/wpa_supplicant.nsi
RTL8188-hostapd-1.1/wpa_supplicant/wpa_supplicant_i.h
RTL8188-hostapd-1.1/wpa_supplicant/wpas_glue.c
RTL8188-hostapd-1.1/wpa_supplicant/wpas_glue.h
RTL8188-hostapd-1.1/wpa_supplicant/wps_supplicant.c
RTL8188-hostapd-1.1/wpa_supplicant/wps_supplicant.h
RTL8188-hostapd-1.1/wpa_supplicant/xcode/
RTL8188-hostapd-1.1/wpa_supplicant/xcode/wpa_supplicant.xcodeproj/
RTL8188-hostapd-1.1/wpa_supplicant/xcode/wpa_supplicant.xcodeproj/project.pbxpro
j
pi@jcb-vpn ~ $
```

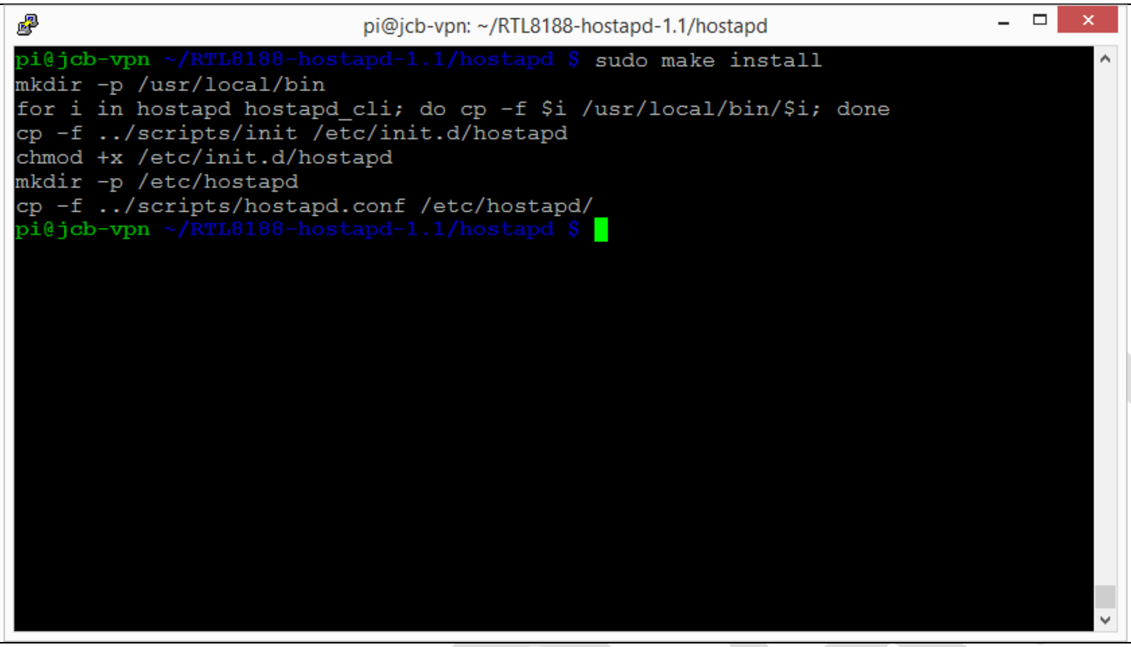
8	
	pi@jcb-vpn: ~/RTL8188-hostapd-1.1/hostapd
	`pi@jcb-vpn ~ $ cd RTL8188-hostapd-1.1/hostapd`
	`pi@jcb-vpn ~/RTL8188-hostapd-1.1/hostapd $ make`
		Execute “cd RTL8188-hostapd-1.1/hostapd”.
		Execute “make”.

9



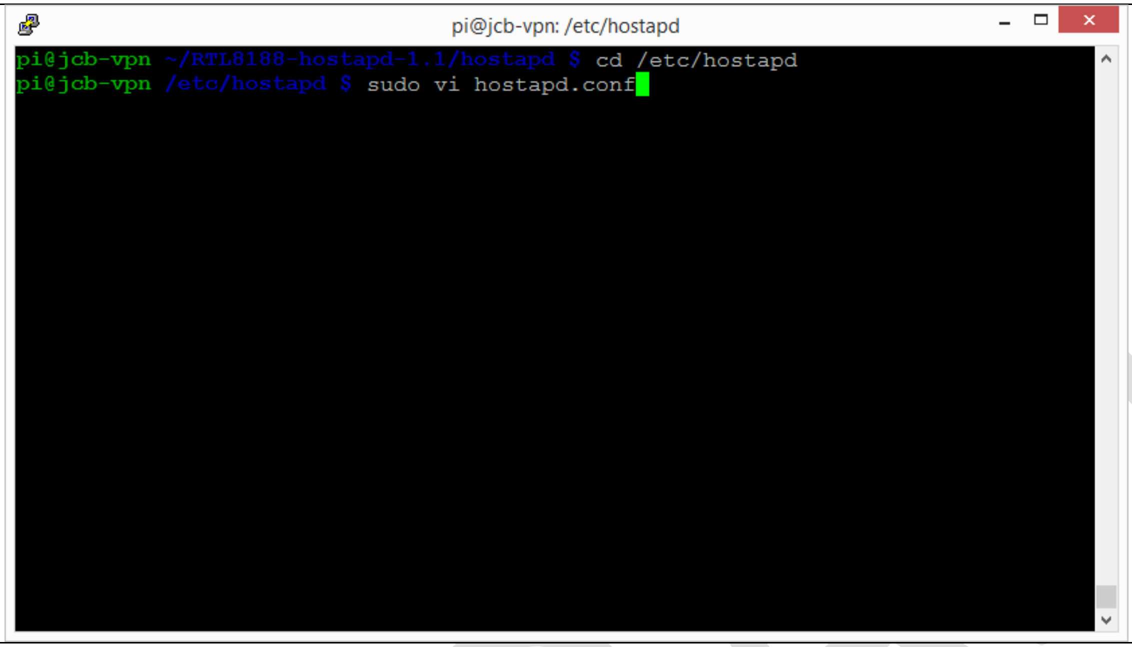
```
  CC  ../src/crypto/sha1-internal.c
  CC  ../src/crypto/sha1-pbkdf2.c
  CC  ../src/crypto/md5-internal.c
  CC  ../src/crypto/rc4.c
  CC  ../src/crypto/sha256.c
  CC  ../src/crypto/sha256-internal.c
  CC  ../src/crypto/dh_groups.c
  CC  ../src/crypto/dh_group5.c
  CC  ../src/crypto/random.c
  CC  ../src/utils/base64.c
  CC  ../src/ap/beacon.c
  CC  ../src/ap/wmm.c
  CC  ../src/ap/ap_list.c
../src/ap/ieee802_11.c: In function ‘handle_action’:
../src/ap/ieee802_11.c:1393:19: warning: variable ‘sta’ set but not used [-Wunus
ed-but-set-variable]
  CC  ../src/ap/ieee802_11.c
  CC  ../src/ap/hw_features.c
  CC  ../src/ap/ieee802_11_ht.c
  LD  hostapd
  CC  hostapd_cli.c
  CC  ../src/common/wpa_ctrl.c
  LD  hostapd_cli
pi@jcb-vpn ~/RTL8188-hostapd-1.1/hostapd $
```

Confirm the package built without error. Some warnings are expected.

10		Execute “sudo make install”.

```
pi@jcb-vpn ~/RTL8188-hostapd-1.1/hostapd $ sudo make install
mkdir -p /usr/local/bin
for i in hostapd hostapd_cli; do cp -f $i /usr/local/bin/$i; done
cp -f ../scripts/init /etc/init.d/hostapd
chmod +x /etc/init.d/hostapd
mkdir -p /etc/hostapd
cp -f ../scripts/hostapd.conf /etc/hostapd/
pi@jcb-vpn ~/RTL8188-hostapd-1.1/hostapd $ 
```

11

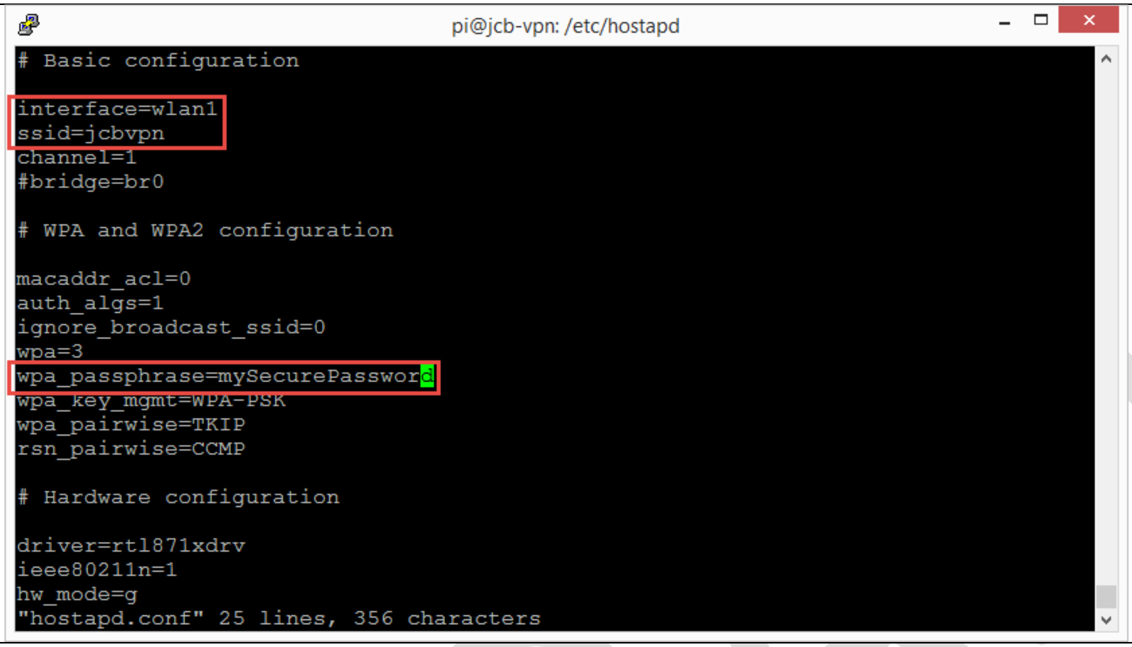


```
pi@jcb-vpn ~/RTL8188-hostapd-1.1/hostapd $ cd /etc/hostapd
pi@jcb-vpn /etc/hostapd $ sudo vi hostapd.conf
```

Execute “cd /etc/hostapd”.

Execute “sudo vi hostapd.conf”.

12



```
# Basic configuration

interface=wlan1
ssid=jcbvpn
channel=1
#bridge=br0

# WPA and WPA2 configuration

macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=3
wpa_passphrase=mySecurePassword
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP

# Hardware configuration

driver=rtl871xdrv
ieee80211n=1
hw_mode=g
"hostapd.conf" 25 lines, 356 characters
```

Set the interface to “wlan1”.

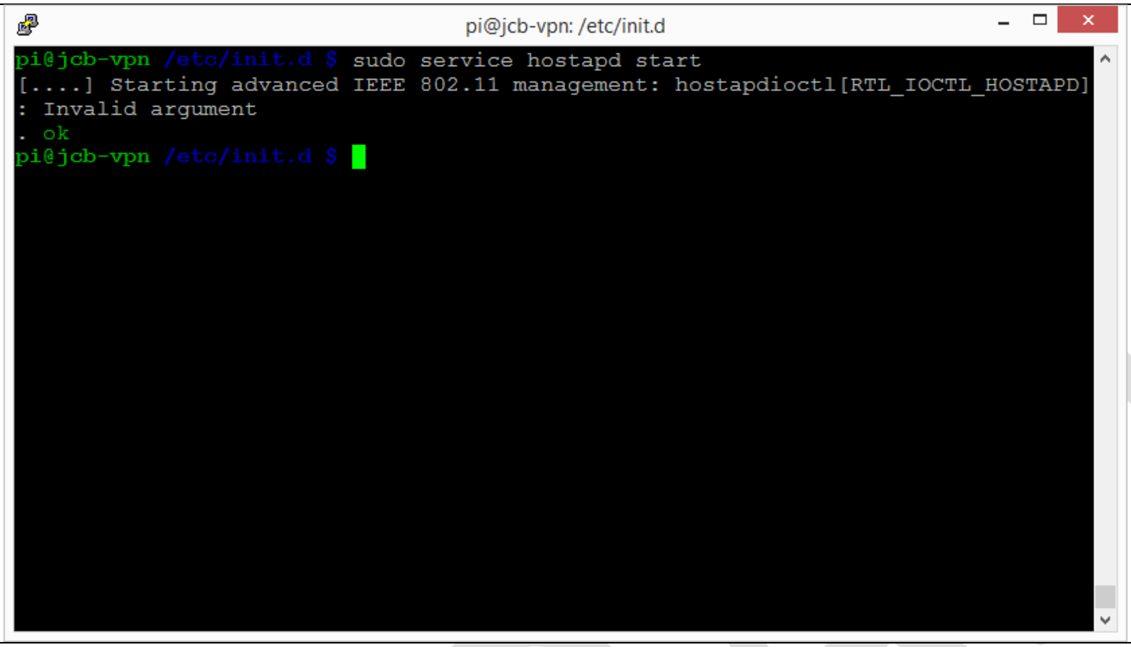
Set the ssid to a suitable value.

Set the wpa_passphrase to a suitable value.

Save the file and exit the editor.

This step configures the Raspberry Pi as a hotspot on wlan1.

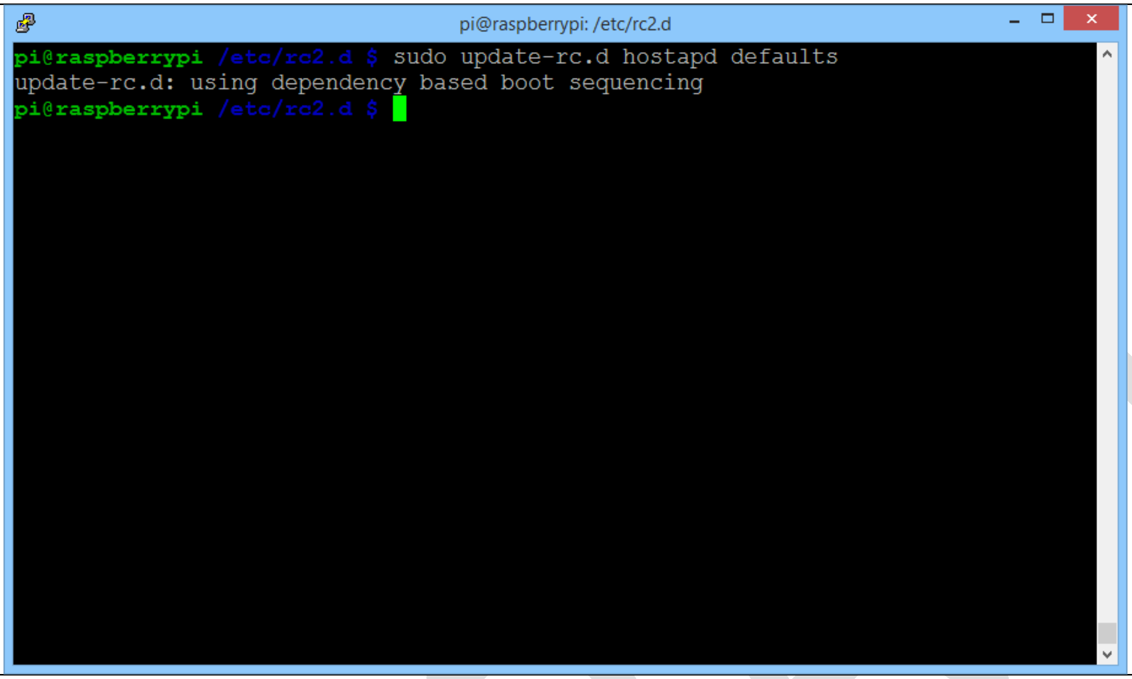
13

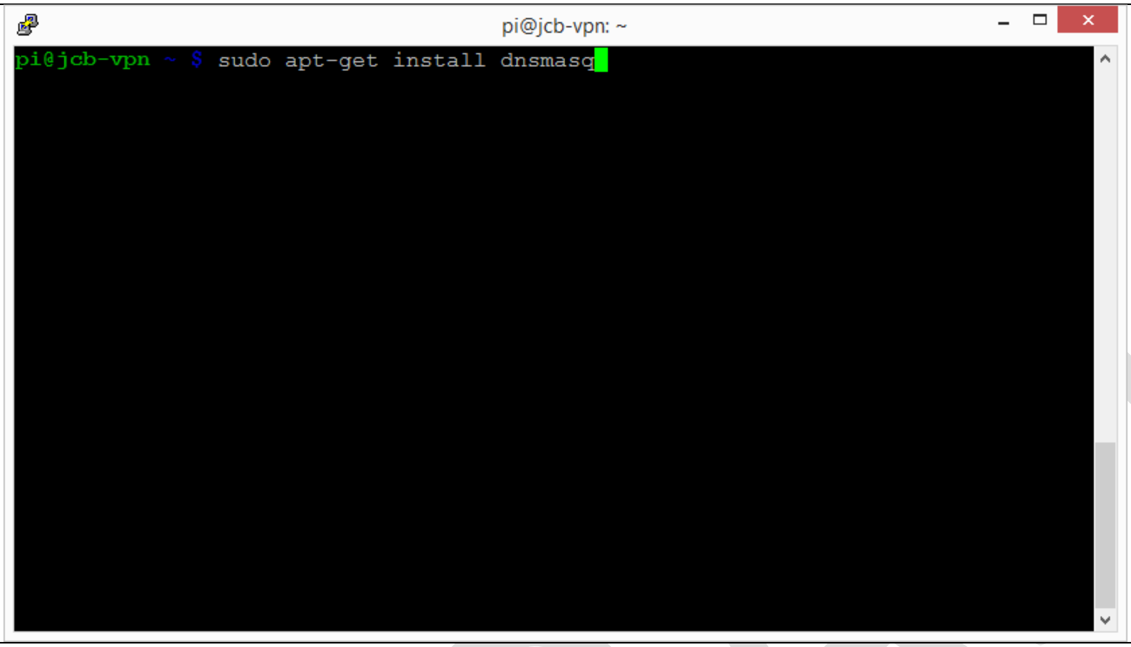


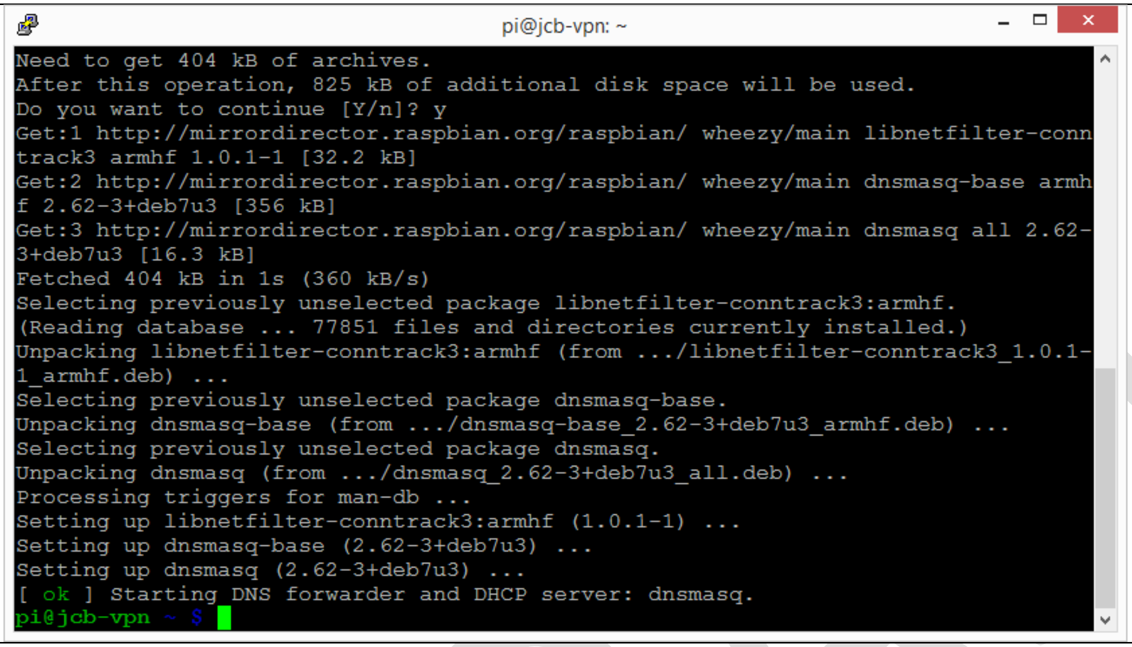
```
pi@jcb-vpn /etc/init.d $ sudo service hostapd start
[....] Starting advanced IEEE 802.11 management: hostapdioctl[RTL_IOCTL_HOSTAPD]
: Invalid argument
. ok
pi@jcb-vpn /etc/init.d $
```

Execute “sudo service hostapd start”.

This steps starts up the hotspot service.

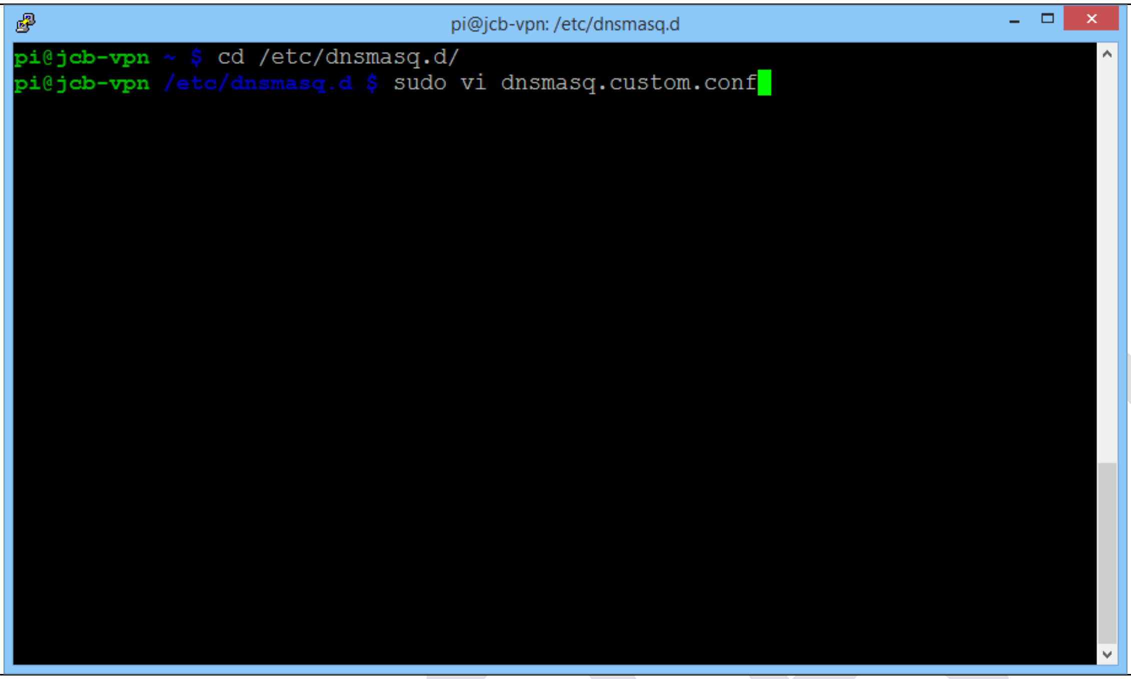
14	
	pi@raspberrypi: /etc/rc2.d
	`pi@raspberrypi /etc/rc2.d $ sudo update-rc.d hostapd defaults`
	`update-rc.d: using dependency based boot sequencing`
	`pi@raspberrypi /etc/rc2.d $`
		Execute “sudo update-rc.d hostapd defaults”.
		This step configures the hotspot service to start when the Raspberry Pi boots.

15	 `pi@jcb-vpn ~ $ sudo apt-get install dnsmasq`	Execute “sudo apt-get install dnsmasq”. Enter “Y” when prompted to continue. This allows the Raspberry Pi to function as a DHCP server and forward DNS requests.

16		Confirm dnsmasq installs and starts.

```
Need to get 404 kB of archives.
After this operation, 825 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 libnetfilter-conn
track3 armhf 1.0.1-1 [32.2 kB]
Get:2 http://mirrordirector.raspbian.org/raspbian/ wheezy/main dnsmasq-base armh
f 2.62-3+deb7u3 [356 kB]
Get:3 http://mirrordirector.raspbian.org/raspbian/ wheezy/main dnsmasq all 2.62-
3+deb7u3 [16.3 kB]
Fetched 404 kB in 1s (360 kB/s)
Selecting previously unselected package libnetfilter-conntrack3:armhf.
(Reading database ... 77851 files and directories currently installed.)
Unpacking libnetfilter-conntrack3:armhf (from .../libnetfilter-conntrack3_1.0.1-
1_armhf.deb) ...
Selecting previously unselected package dnsmasq-base.
Unpacking dnsmasq-base (from .../dnsmasq-base_2.62-3+deb7u3_armhf.deb) ...
Selecting previously unselected package dnsmasq.
Unpacking dnsmasq (from .../dnsmasq_2.62-3+deb7u3_all.deb) ...
Processing triggers for man-db ...
Setting up libnetfilter-conntrack3:armhf (1.0.1-1) ...
Setting up dnsmasq-base (2.62-3+deb7u3) ...
Setting up dnsmasq (2.62-3+deb7u3) ...
[ ok ] Starting DNS forwarder and DHCP server: dnsmasq.
pi@jcb-vpn ~ $
```

17

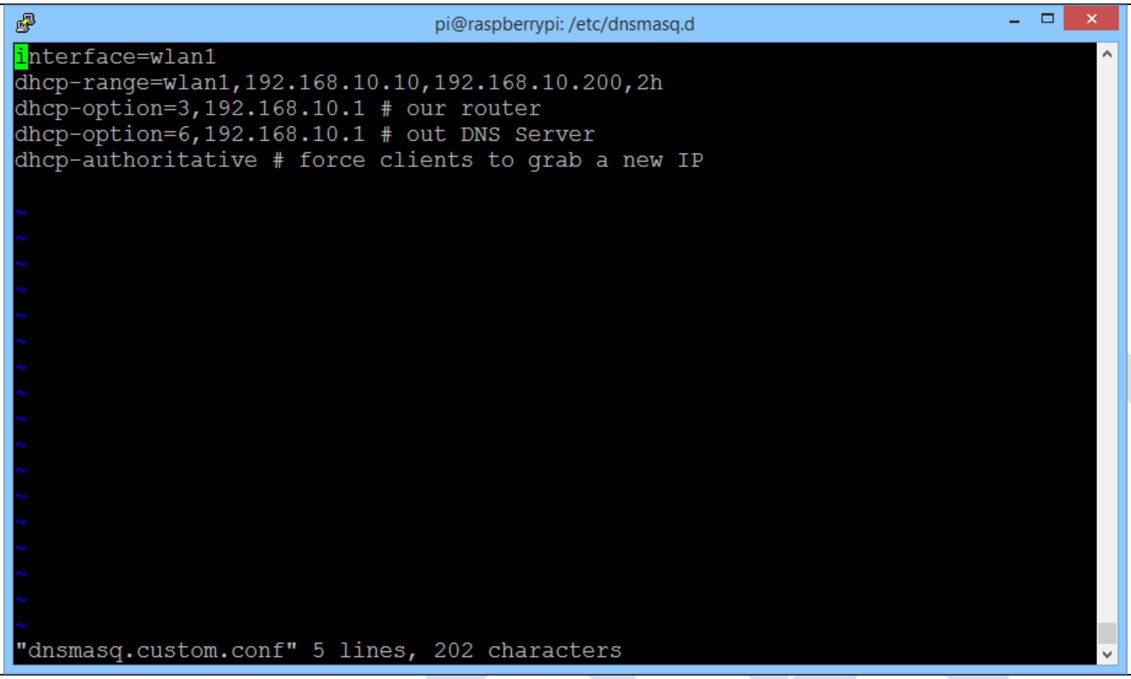


```
pi@jcb-vpn ~ $ cd /etc/dnsmasq.d/
pi@jcb-vpn /etc/dnsmasq.d $ sudo vi dnsmasq.custom.conf
```

Execute “cd /etc/dnsmasq.d”

Execute “sudo vi dnsmasq.custom.conf”

18



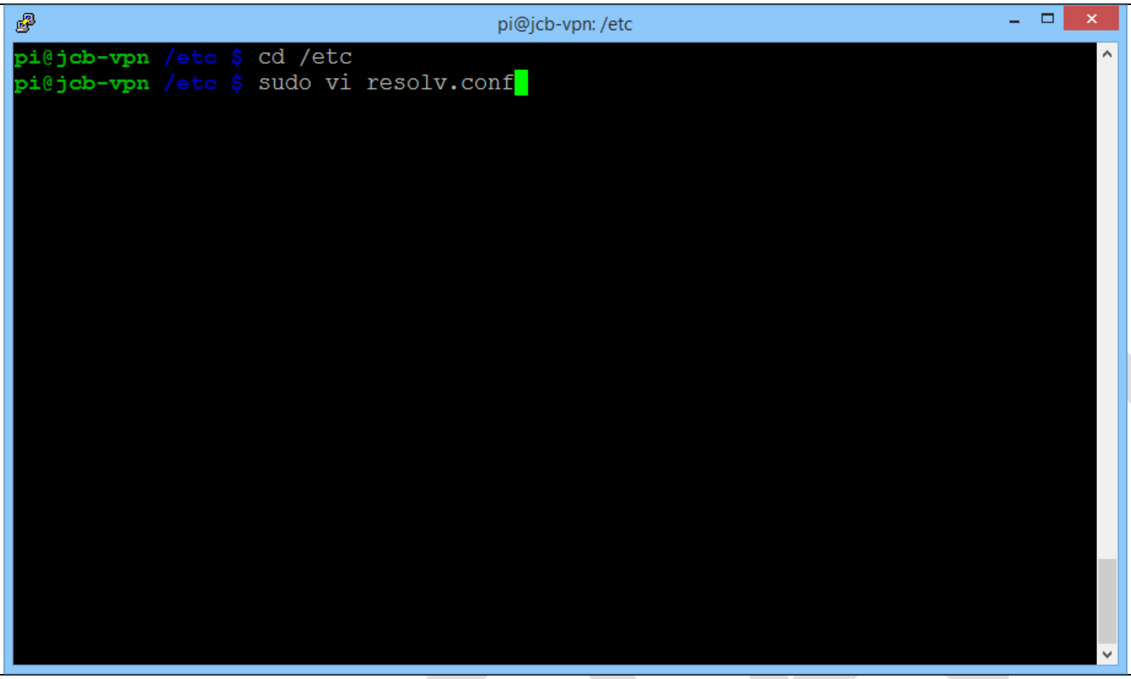
```
interface=wlan1
dhcp-range=wlan1,192.168.10.10,192.168.10.200,2h
dhcp-option=3,192.168.10.1 # our router
dhcp-option=6,192.168.10.1 # out DNS Server
dhcp-authoritative # force clients to grab a new IP

~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"dnsmasq.custom.conf" 5 lines, 202 characters
```

Enter the contents to the left and save the file.

This step sets the DHCP server configuration for the Raspberry Pi hotspot.

```
======================
interface=wlan1
dhcp-range=wlan1,192.168.10.10,192.168.10.200,2h
dhcp-option=3,192.168.10.1 # our router
dhcp-option=6,192.168.10.1 # our DNS Server
dhcp-authoritative # force clients to grab a new IP
```

19

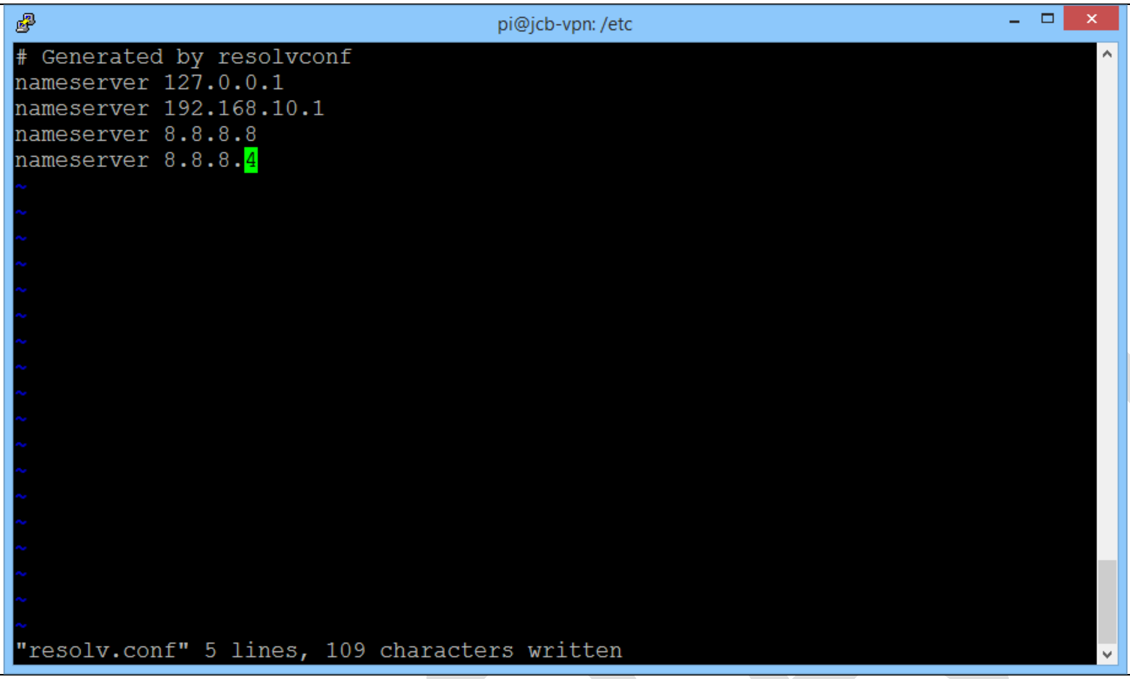


```
pi@jcb-vpn /etc $ cd /etc
pi@jcb-vpn /etc $ sudo vi resolv.conf
```

Execute “cd /etc”.

Execute “sudo vi resolv.conf”.

20



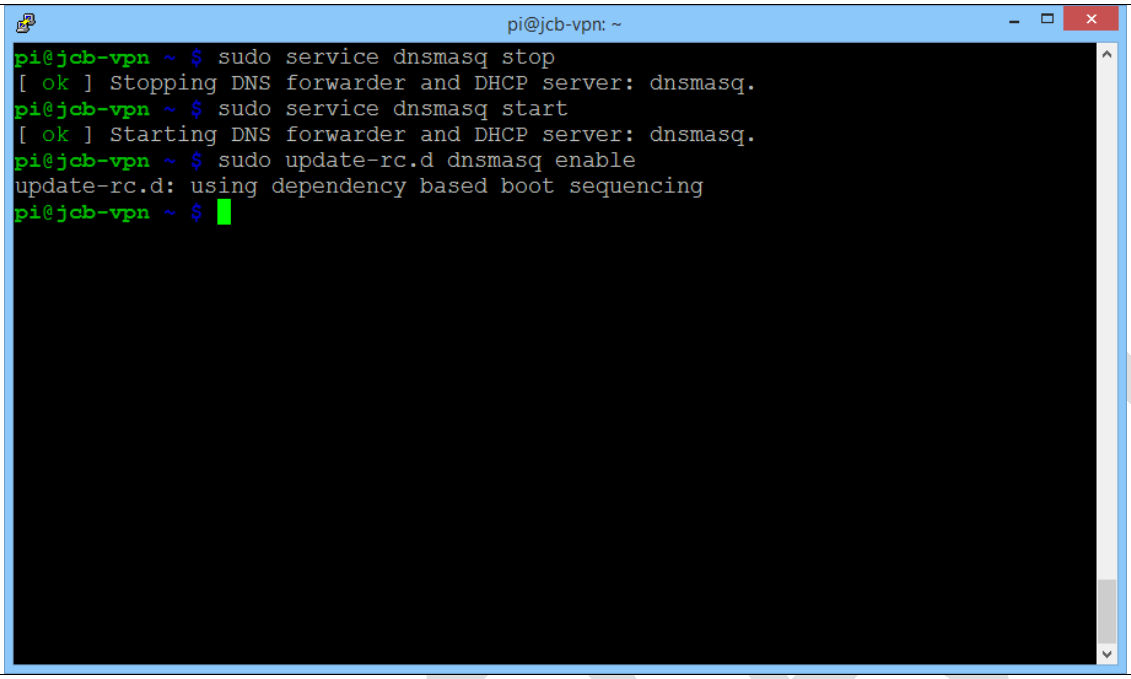
```
# Generated by resolvconf
nameserver 127.0.0.1
nameserver 192.168.10.1
nameserver 8.8.8.8
nameserver 8.8.8.4
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"resolv.conf" 5 lines, 109 characters written
```

Add the following lines to the file:

nameserver 127.0.0.1
nameserver 192.168.10.1
nameserver 8.8.8.8
nameserver 8.8.8.4

This step defines the DNS servers that will be used by this Raspberry Pi to resolve server names.

21



```
pi@jcb-vpn ~ $ sudo service dnsmasq stop
[ ok ] Stopping DNS forwarder and DHCP server: dnsmasq.
pi@jcb-vpn ~ $ sudo service dnsmasq start
[ ok ] Starting DNS forwarder and DHCP server: dnsmasq.
pi@jcb-vpn ~ $ sudo update-rc.d dnsmasq enable
update-rc.d: using dependency based boot sequencing
pi@jcb-vpn ~ $ 
```

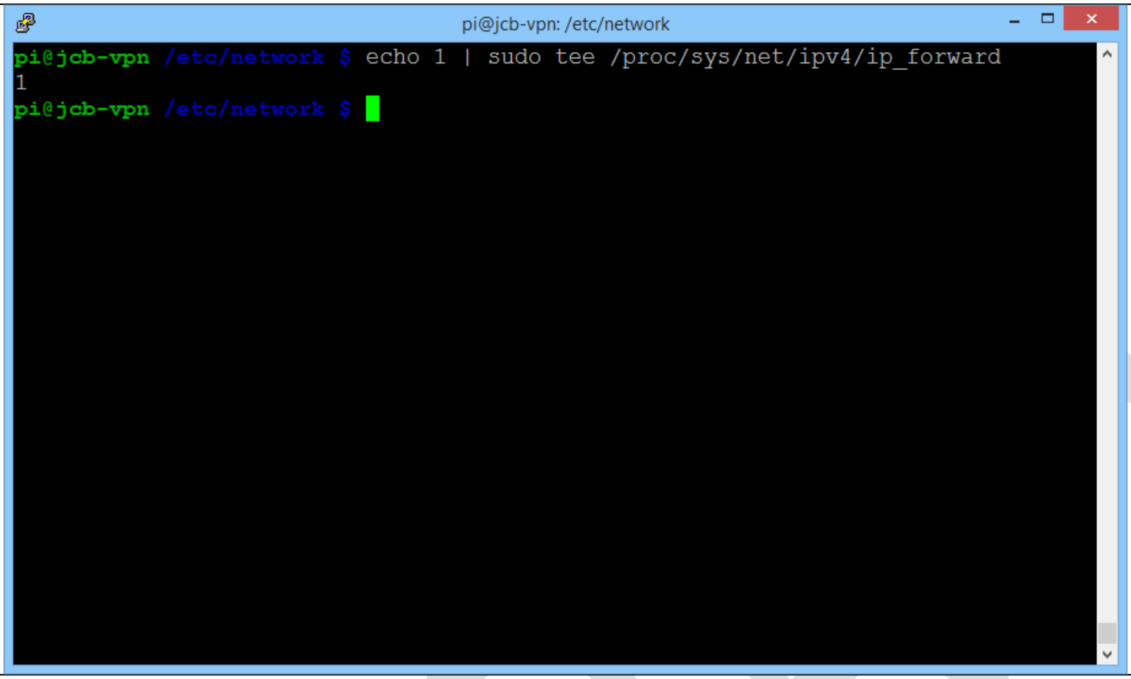
Execute “sudo service dnsmasq stop”.

Execute “sudo service dnsmasq start”.

Execute “sudo update-rc.d dnsmasq enable”.

This step restarts dnsmasq to use the new configuration and sets it to automatically start on boot.

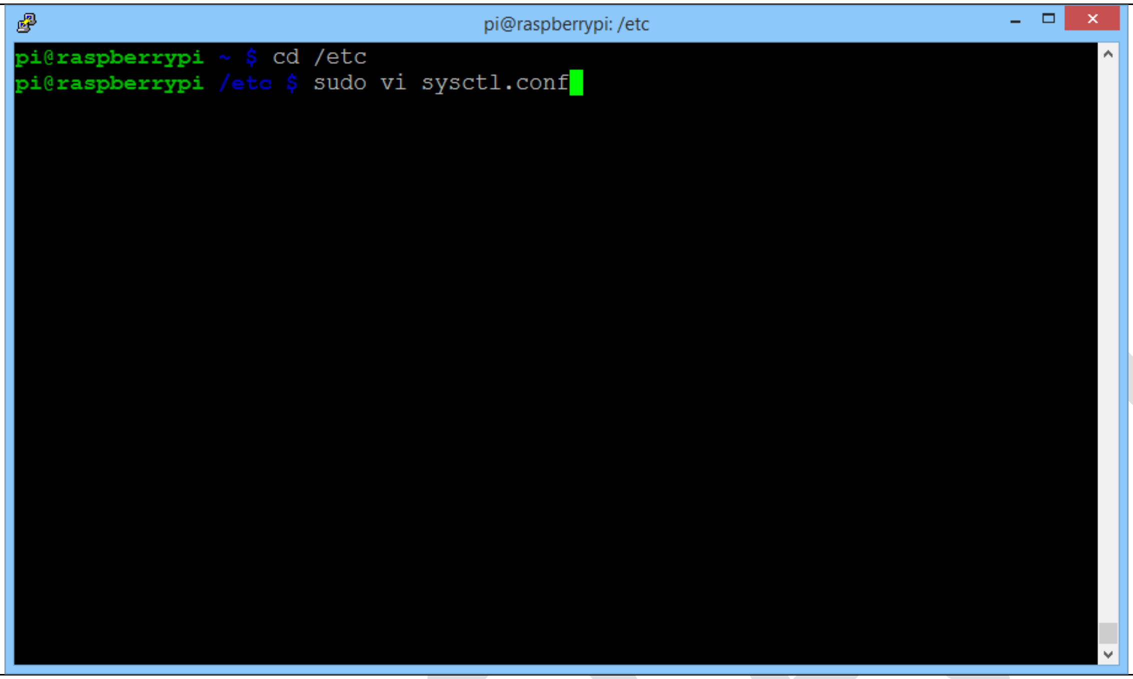
22



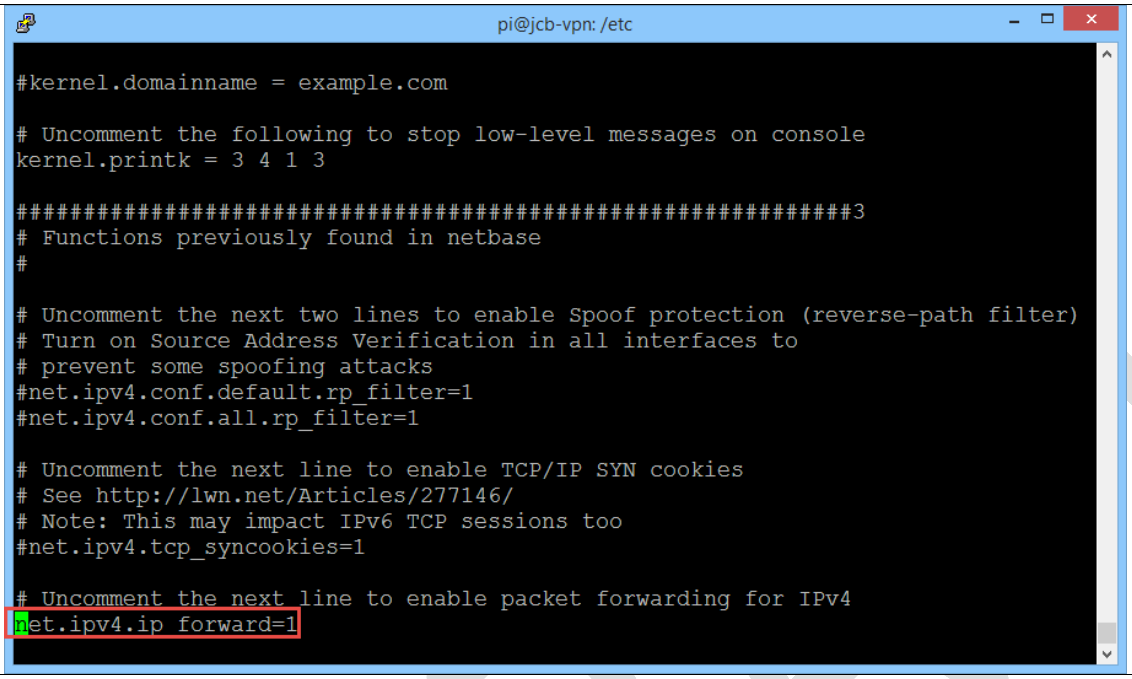
```
pi@jcb-vpn /etc/network $ echo 1 | sudo tee /proc/sys/net/ipv4/ip_forward
1
pi@jcb-vpn /etc/network $ 
```

Execute "`echo 1 | sudo tee /proc/sys/net/ipv4/ip_forward`".

This step enables routing network packets through the Raspberry Pi.

23	
	`pi@raspberrypi ~ $ cd /etc`
	`pi@raspberrypi /etc $ sudo vi sysctl.conf`
		Execute “cd /etc”.
		Execute “sudo vi sysctl.conf”.

24

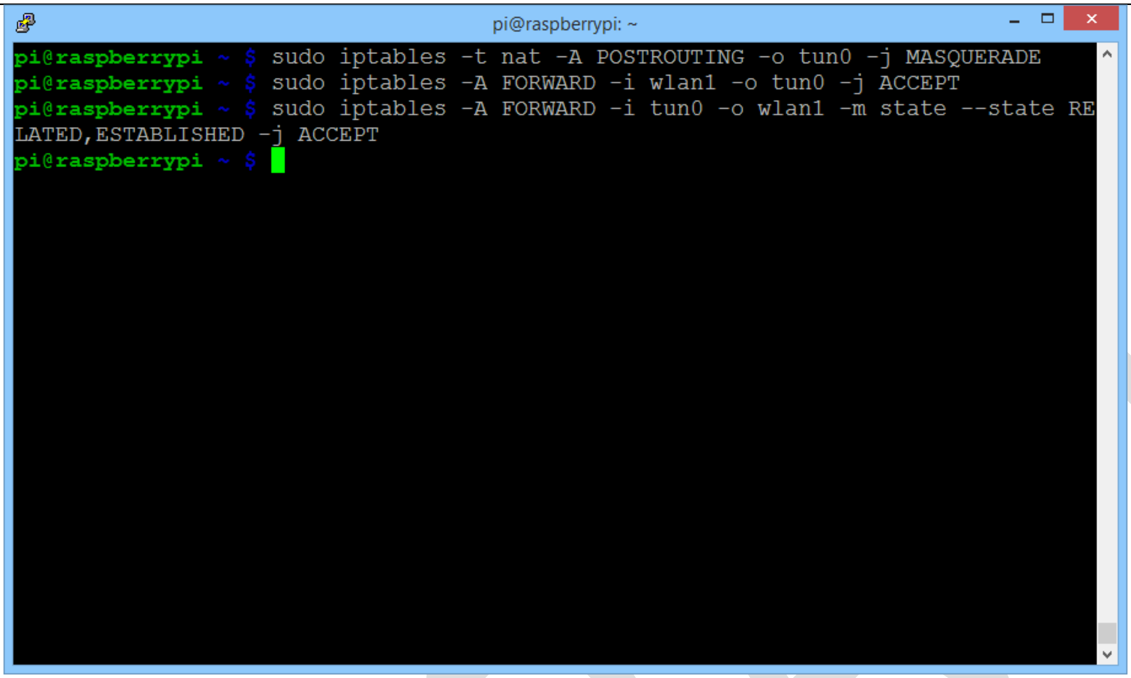


```
#kernel.domainname = example.com

# Uncomment the following to stop low-level messages on console
kernel.printk = 3 4 1 3

##############################################################3
# Functions previously found in netbase
#

# Uncomment the next two lines to enable Spoof protection (reverse-path filter)
# Turn on Source Address Verification in all interfaces to
# prevent some spoofing attacks
#net.ipv4.conf.default.rp_filter=1
#net.ipv4.conf.all.rp_filter=1

# Uncomment the next line to enable TCP/IP SYN cookies
# See http://lwn.net/Articles/277146/
# Note: This may impact IPv6 TCP sessions too
#net.ipv4.tcp_syncookies=1

# Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1
```

In the sysctl.conf file, uncomment the line that contains “net.ipv4.ip_forward”.

This step causes the Raspberry Pi to route network packets on boot.

25



```
pi@raspberrypi ~ $ sudo iptables -t nat -A POSTROUTING -o tun0 -j MASQUERADE
pi@raspberrypi ~ $ sudo iptables -A FORWARD -i wlan1 -o tun0 -j ACCEPT
pi@raspberrypi ~ $ sudo iptables -A FORWARD -i tun0 -o wlan1 -m state --state RE
LATED,ESTABLISHED -j ACCEPT
pi@raspberrypi ~ $ 
```

WARNING: May need to change the “tun0” to “wlan0” in all cases

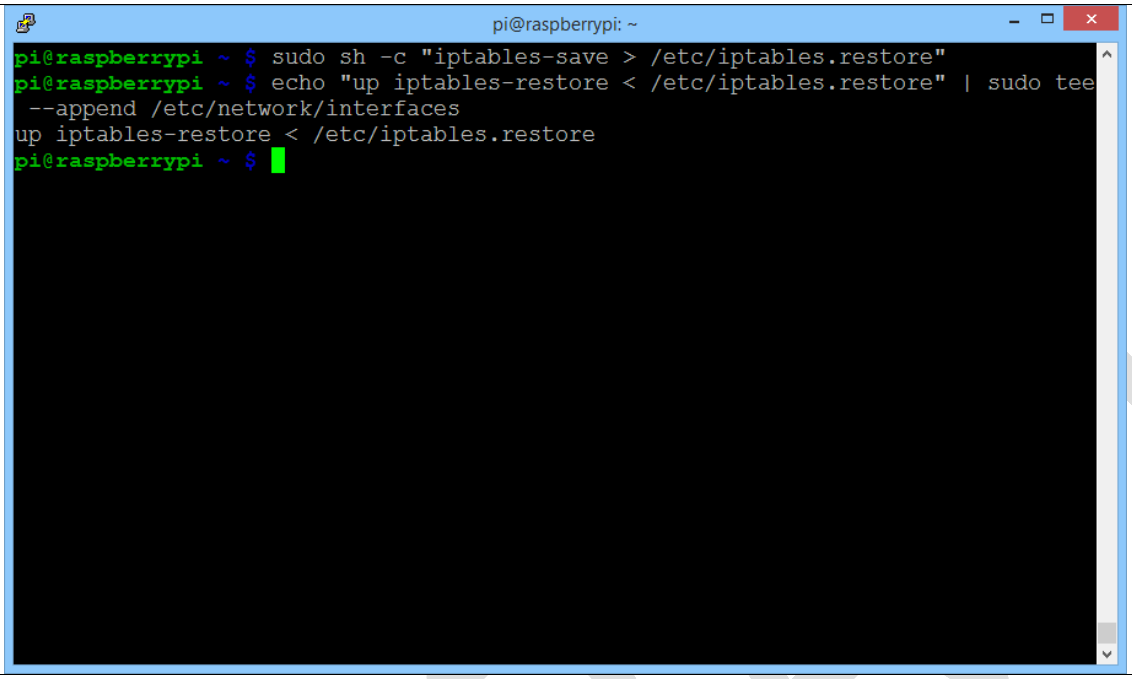
Execute “sudo iptables -t nat -A POSTROUTING -o tun0 -j MASQUERADE”

Execute “sudo iptables -A FORWARD -i wlan1 -o tun0 -j ACCEPT”

Execute “sudo iptables -A FORWARD -i tun0 -o wlan1 -m state --state RELATED,ESTABLISHED -j ACCEPT”

These steps configure the Raspberry Pi for NAT, configures it to forward all traffic coming in on wlan1, and returns data to the appropriate client.

26



```
pi@raspberrypi ~ $ sudo sh -c "iptables-save > /etc/iptables.restore"
pi@raspberrypi ~ $ echo "up iptables-restore < /etc/iptables.restore" | sudo tee
 --append /etc/network/interfaces
up iptables-restore < /etc/iptables.restore
pi@raspberrypi ~ $ 
```

Execute “sudo sh -c "iptables-save > /etc/iptables.restore"”.

Execute “echo "up iptables-restore < /etc/iptables.restore" | sudo tee --append /etc/network/interfaces”.

This steps configures the routing when the Raspberry Pi boots.