

# Setting up a Raspberry Pi as an access point with a captive Portal and optional Internet access

## 1) Guide to set up Raspbree pi SDCard with a Wheezy Image

Material required:

- Raspberry Pi Model B
- Power adapter or USB cable and power source
- SD storage card, preferably >8GB
- USB WiFi Adapter (ASUS USB-N13)
- Ethernet cable
- USB storage (USB key or thumbdrive)
- Raspbian “wheezy” Linux distribution (2013-02-09-wheezy-raspbian)

### a) Flash an SD Card Using Linux

<http://www.raspberrypi.org/documentation/installation/installing-images/linux.md>

The Linux distributions provided, for Raspberry Pi, are done so as a compressed image file. You will need to download, uncompress and then install onto your SD card. For the purposes of this tutorial I am using **Ubuntu**, one of the most popular Linux distributions.

### *Format the SD Card*

<http://qdosmsg.dunbar-it.co.uk/blog/2013/06/noobs-for-raspberry-pi/>

It may be necessary to format the SD card so that it can be read by your Raspberry Pi. If this is the case, you can use the **Disk Utility** application in Ubuntu.

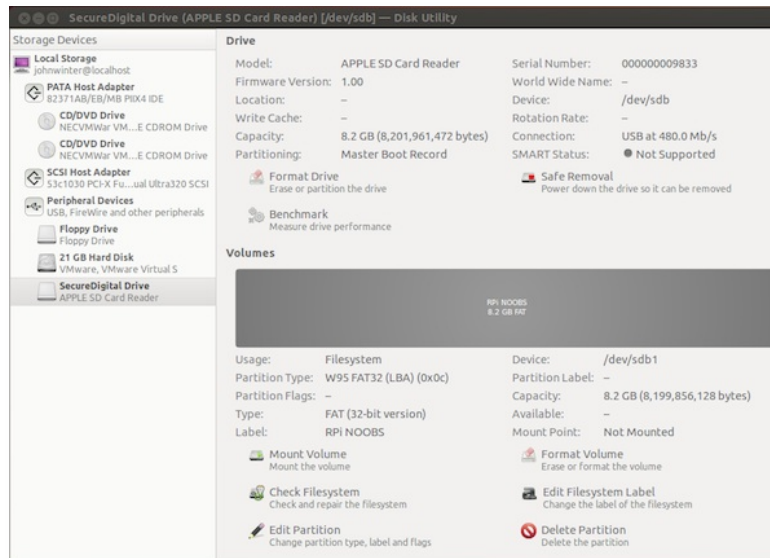


Figure 1: The Ubuntu Disk Utility to format the SD card

Select your SD card and format as **FAT**.



Figure 2: Formatting the SD card using Ubuntu Linux

## Identify the SD Card

Plug the SD card into the SD card drive on your Linux machine.

Open a terminal window and, to list the attached discs, type:

1 `sudo fdisk -l`

```
johnwinter@ubuntu: ~  
johnwinter@ubuntu:~$ sudo fdisk -l  
[sudo] password for johnwinter:  
  
Disk /dev/sda: 21.5 GB, 21474836480 bytes  
255 heads, 63 sectors/track, 2610 cylinders, total 41943040 sectors  
Units = sectors of 1 * 512 = 512 bytes  
Sector size (logical/physical): 512 bytes / 512 bytes  
I/O size (minimum/optimal): 512 bytes / 512 bytes  
Disk identifier: 0x000a7683  
  
   Device Boot      Start         End      Blocks   Id  System  
/dev/sda1  *          2048       39845887   19921920   83   Linux  
/dev/sda2                39847934   41940991    1046529    5   Extended  
/dev/sda5                39847936   41940991    1046528   82   Linux swap / Solaris  
  
Disk /dev/sdb: 8201 MB, 8201961472 bytes  
255 heads, 63 sectors/track, 997 cylinders, total 16019456 sectors  
Units = sectors of 1 * 512 = 512 bytes  
Sector size (logical/physical): 512 bytes / 512 bytes  
I/O size (minimum/optimal): 512 bytes / 512 bytes  
Disk identifier: 0x00000000  
  
   Device Boot      Start         End      Blocks   Id  System  
/dev/sdb1                8192       16019455    8005632    b   W95 FAT32  
johnwinter@ubuntu:~$
```

Listing the attached volumes in Ubuntu Linux

Identify the SD card and make a note of the device address, such as `/dev/sdX` where X denotes the storage device. Note, you can easily identify your SD card by looking at the capacities of the discs listed.

## Download Raspian *Wheezy* Image and Flash the SD Card

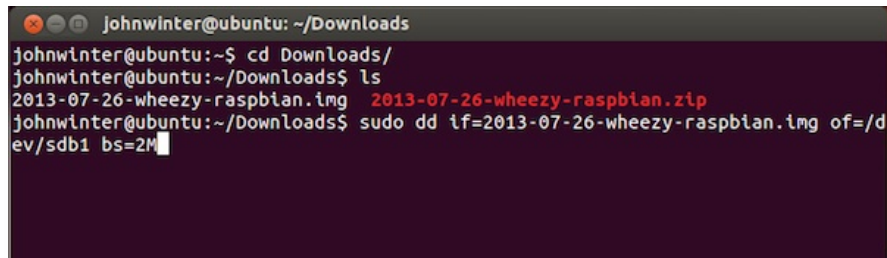
Once you have identified and formatted your SD card, you need to flash the SD card. This is done with the unix command `dd`.

Before proceeding, navigate to the directory that contains the downloaded (<http://www.raspberrypi.org/downloads/>) and extracted `.img` file.

cd Downloads

In a terminal window, enter the following, ensuring that the distribution name (in this example: if=2013-07-26-wheezy-raspbian.img) is correct and that the destination (in this example: /dev/sdb1) as they may be different to the ones used in this example:

**sudo dd if=2013-07-26-wheezy-raspbian.img of=/dev/sdb1 bs=2M**

A terminal window titled 'Johnwinter@ubuntu: ~/Downloads' showing the following commands and output:

```
Johnwinter@ubuntu:~/Downloads$ cd Downloads/
Johnwinter@ubuntu:~/Downloads$ ls
2013-07-26-wheezy-raspbian.img  2013-07-26-wheezy-raspbian.zip
Johnwinter@ubuntu:~/Downloads$ sudo dd if=2013-07-26-wheezy-raspbian.img of=/dev/sdb1 bs=2M
```

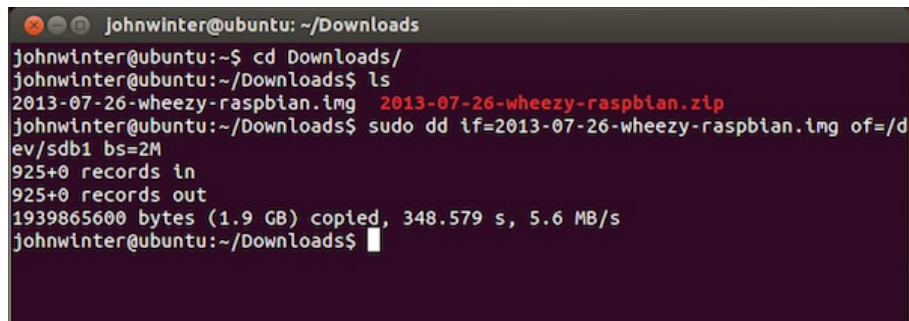
Locating the downloaded image file

**TIP:** You will need to exercise care using the dd command. Careless use may result in the corruption of your computer's main hard drive if you incorrectly identify the drive required.

The process of flashing the SD card will take around two or three minutes during which time you are given no indication that anything is happening, unless your SD drive has a flashing LED.

Once the SD card has been flashed, you will see a further message in the terminal window.

Unmount the SD card and it is now ready for use in your Raspberry Pi.

A terminal window titled 'Johnwinter@ubuntu: ~/Downloads' showing the completion of the dd command:

```
Johnwinter@ubuntu:~/Downloads$ sudo dd if=2013-07-26-wheezy-raspbian.img of=/dev/sdb1 bs=2M
925+0 records in
925+0 records out
1939865600 bytes (1.9 GB) copied, 348.579 s, 5.6 MB/s
Johnwinter@ubuntu:~/Downloads$
```

Terminal confirmation of completion

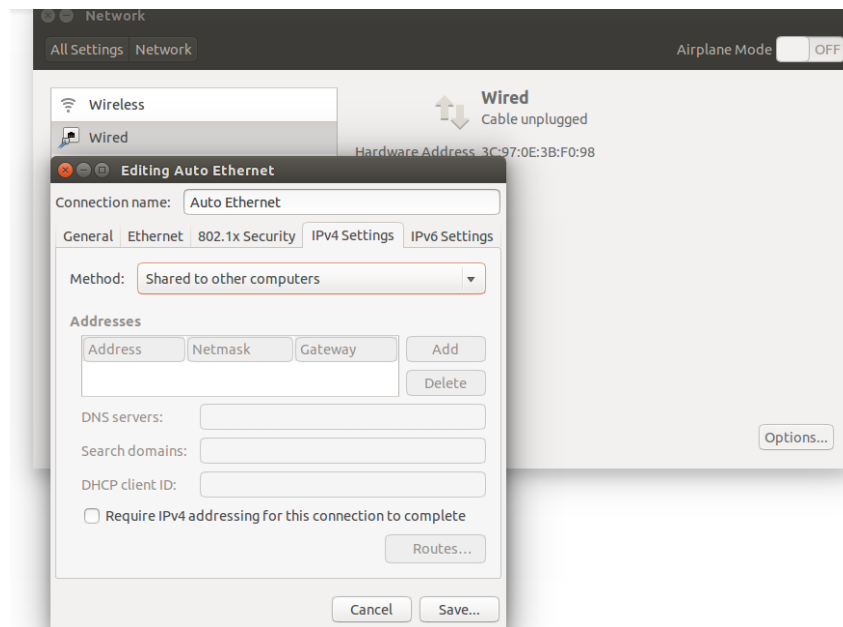
# Boot your Raspberry Pi:

1- Add the SD card in the right slot in the RP

## Connect via SSH to Raspberry Pi (No screen nor keyboard required):

1- Connect your laptop directly to the the Raspberry PI using a LAN/Ethernet cable

2- In ubuntu, go to Settings->Network=>Wired->Options->IPv4Settings and choose “Shared to other computers”



In windows, enable “Internet Sharing” through the laptop’s wireless connection. In the network and sharing center, right click on your enabled/connected wireless connection, select “properties”, then select the tap “sharing”. Choose to enable the sharing, and make sure it’s through the Laptop’s LAN.

3- Connect your laptop to a working wireless connection which will be the RP’s access to the internet (to install libraries and packages required in the next steps). Your ethernet connection will now be used for the communication with the Raspberry Pi.

4- Plug the Raspberry to an energy source (if you have a USB cable you can plug it on your laptop)

It should be assigned an IP address by the Laptop's DHCP. At this point, both the laptop and the RP share the LAN connection. This makes it possible to SSH the RP from your laptop, and even provides internet access to the RP.

5- To know which IP address is assigned to the RP (required for SSH) : use the command "arp -a" on your laptop

(perhaps it would be useful to run this command before connecting the RPi to see the output before and thus recognize more easily the RPi address)

6- In a command prompt: "ssh pi@ip\_address\_of\_the\_PI"

Are you sure you want to continue connecting (yes/no)? yes  
password: raspberry

7 - In a Terminal type 'sudo apt-get update' and press Enter to update 'apt-get' catalog

(This may take several minutes and will ensure that all online software packages and references are up to date.)

## **Install Apache Webserver and PHP**

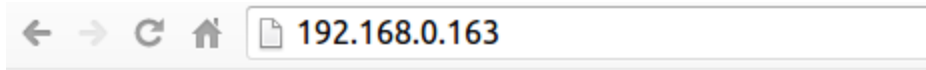
Apache is a popular web server that you can install on the Raspberry Pi to allow it to host web pages. On its own, Apache can serve HTML files over HTTP, and with additional modules can serve dynamic web pages using scripting languages such as PHP.

First install the `apache2` package by typing the following command in the Terminal

```
sudo apt-get install apache2 -y
```

By default, Apache puts a test HTML file in the web folder. This default web page is served when you browse to `http://localhost/` on the Pi itself, or `http://192.168.1.10` (or whatever the Pi's IP address is) from another computer on the network. To find the Pi's IP address, type `hostname -I` at the command line (or read more about finding your [IP address](#)). It should be the same you found using "arp -a" at step 6 above ...

Browse to the default web page either on the Pi or from another computer on the network and you should see the following:



# It works!

This is the default web page for this server.

The web server software is running but no content has been added, yet.

This default web page is just a HTML file on the filesystem. It is located at `/var/www/index.html`. Navigate to this directory in the Terminal and have a look at what's inside:

```
cd /var/www
ls -al
```

This will show you:

```
total 12
drwxr-xr-x  2 root root 4096 Jan  8 01:29 .
drwxr-xr-x 12 root root 4096 Jan  8 01:28 ..
-rw-r--r--  1 root root  177 Jan  8 01:29 index.html
```

This shows that there is one file in `/var/www/` called `index.html`. The `.` refers to the directory itself `/var/www/` and the `..` refers to the parent directory `/var/`.

## WHAT THE COLUMNS MEAN

1. The permissions of the file or directory
2. The number of files in the directory (or 1 if it's a file).
3. The user which owns the file or directory
4. The group which owns the file or directory
5. The file size
6. The last modification date & time

As you can see, by default the `www` directory and `index.html` file are both owned by the `root` user. In order to edit the file, you must gain root permissions. Either change the owner to your own user (`sudo chown pi: index.html`) before editing, or edit with `sudo`, e.g. `sudo nano index.html`.

Try editing this file and refreshing the browser to see the web page change.

**Notes:**

- In my case, I had to add "ServerName localhost" to /etc/apache2/apache2.conf to get the apache server running correctly. (better add it at the "#Global configuration" section)

- After any modification in the apache2.conf file, you can restart the server by running:

```
"sudo service apache2 restart"
```

**ADDITIONAL - INSTALL PHP**

To allow your Apache server to process PHP files, you'll need to install PHP5 and the PHP5 module for Apache. Type the following command to install these:

```
sudo apt-get install php5 libapache2-mod-php5 php5-common php5-cgi -y
```

Now move the index.html file to index.php:

```
sudo mv index.html index.php
```

and edit the file:

```
sudo nano index.php
```

to put some PHP content in it:

```
<?php echo "hello world"; ?>
```

Now save and refresh your browser. You should see "hello world". This is not dynamic but still served by PHP. Try something dynamic:

```
<?php echo date('Y-m-d H:i:s'); ?>
```

or show your PHP info:

```
<?php phpinfo(); ?>
```

**ADDITIONAL TIPS**

- 1) Set permissions on the web directory /var/www/ (as seen here:  
<http://www.penguintutor.com/linux/light-webserver>)

Change the directory owner and group



**sudo chown www-data:www-data /var/www**

allow the group to write to the directory

**sudo chmod 775 /var/www**

Add the pi user to the www-data group

**sudo usermod -a -G www-data p**

2) Type 'ifconfig' for interface/IP information

3) To shutdown, sudo shutdown -h now

4) To change the pi user password type passwd (you have to do this if you don't want anyone to have access to your Raspberry Pi!)

## **Wireless Networking (Access point configurations and DHCP):**

Get a USB wireless adapter for the Raspberry pi. The following is how to configure the wireless interface as a step to create a wireless networking zone for nearby users.

1. Open a Terminal and type 'sudo apt-get install hostapd'. Enter
2. Plug in the wireless adapter to an available USB port on the Pi. The terminal might freeze, and takes sometime to recognize the adapter. You might need to reboot by typing:  
sudo reboot

Or in the worst case, unplug and replug the power cable of the RPi. And then access again via SSH as described previously.

- a. We are using an ASUS USB-N13
3. In a terminal type 'lsusb', Enter to get a list of recognized devices. The USB-N13 should appear on the list

Edit the “interfaces” file

- a. Type ‘cd /etc/network’ Enter.
- b. Type ‘sudo nano interfaces’ to open file in text editor
- c. Comment (prepend with “#”) the following lines, which should look like this:

```
#auto lo
#iface wlan0 init manual
#wpa-roam /etc/wpa_supplicant/wpa_supplicant/conf
```

- d. Add the following lines to the bottom of the file

```
auto wlan0
iface wlan0 inet static
address 10.0.0.1
netmask 255.255.255.0
```

- e. Close and save “interfaces”
  - i. Control+X, Y, Enter
4. Type ‘cd /etc/default’ Enter
5. Edit the “hostapd” file:
  - a. Type ‘sudo nano hostapd’ Enter
  - b. Remove the pound sign (#) that appears before the line ‘DAEMON\_CONF=’ and inside the quotes type DAEMON\_CONF=“/etc/hostapd/hostapd.conf”.
  - c. Close and save
    - i. CTRL+X, Y, Enter.
6. Type ‘cd /etc/hostapd’ Enter.
7. Create the config file specified
  - a. Type ‘sudo nano hostapd.conf’, Enter
  - b. add the following lines

```
interface=wlan0
ssid= JukeBox_”your_name”
hw_mode=g
channel=6

beacon_int=100
auth_algs=3
wmm_enabled=1
```

If you would like to add a password put the following lines in hostapd.conf

```
interface=wlan0
driver=nl80211
ssid=Jukebox_Abdullah
hw_mode=g
channel=6
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=3
wpa_passphrase=my_password
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
```

- c. Close and save
  - i. CTRL+X, Y and Enter
- 8. Test SSID broadcast
  - a. Directory doesn't matter ... (Type 'cd /etc' Enter)
  - b. sudo service hostapd start
  - c. sudo service hostapd stop
- 9. Install and configure dnsmasq (application for managing DHCP)
  - a. Type 'sudo apt-get install dnsmasq', Enter
  - b. Hit Y when prompted.
  - c. Type 'sudo nano /etc/dnsmasq.conf' to edit dnsmasq configuration and add the following lines to the top of the file:

```
# This sets the subnet mask to 255.255.255.0 and specifies a lease
time of 12 hours.
```

```
interface=wlan0
dhcp-range=10.0.0.2,10.0.0.10,255.255.255.0,12h
```

```
#required for traffic forwarding
dhcp-option=3,10.0.0.1
```

**USE THE FOLLOWING IF IN OFF-LINE MODE**

```
address=:/10.0.0.1 #redirect all DNS requests to 10.0.0.1 (but I comment to do
forwarding later using iptables)
```

d. CTRL+X, Y, Enter

## 10. Reboot

a. Services should start automatically if all required devices (WiFi interface) are connected.

b. From now on, when you connect to the Raspberry through the WiFi interface you can connect to it always through the IP address **10.0.0.1**

c. If you wish, you can now get rid of the ethernet cable

### **TIP:**

If you want to manually Start (or Restart) services, use the following

a. `sudo service hostapd start` (restart for hostapd does not work, need to stop/start)

b. `sudo service dnsmasq restart`

Within range of the Pi, using a laptop, smart phone or other mobile device, connect to the Wireless Access Point labeled 'JukeBox\_yourname'..

For more advanced web development, bootstrap could be installed to your Raspbree PI from: <http://twitter.github.io/bootstrap/index.html>

## **Forwarding Wireless traffic to LAN (Internet access) using Iptables:**

We will turn the Raspberry Pi into a router. The Pi will use the ethernet port as the "WAN" port, and the wireless adapter as the "LAN" side of things.

First thing we need to do is to enable packet forwarding.

In the file `/etc/sysctl.conf`, we need to uncomment the following line (should be **line 28**).

before:

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

after:

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

After changing that, run this command to re-read the sysctl.conf file

```
sudo sysctl -p
```

We will also need to install the **iptables** utilities if they are not already installed, and alter the dnsmasq.conf file so that dnsmasq will assign a gateway to the computers.

Firstly we install iptables.

```
sudo apt-get install iptables
```

Check that there is this line at your dnsmasq.conf (see above).

The IP address is the same one we gave to the WiFi adapter before

```
dhcp-option=3,10.0.0.1
```

We will need to create an iptables script to tell the Raspberry Pi to forward packets from the WiFi interface to the LAN interface. This script will need to run at startup in order for it to be a router.

You need to create a file called “router” in /etc/network/if-up.d/. By placing the script in that directory, it will be run every time the Raspberry Pi comes online (if you need to put additional scripts that you wish to run when the network interface goes up put it there).

This file contains the following lines

```
#!/bin/sh
```

```
iptables -F
```

```
iptables -X
```

```
iptables -A INPUT -i lo -j ACCEPT
```

```
iptables -A OUTPUT -o lo -j ACCEPT
```

```
iptables -A INPUT -i wlan0 -j ACCEPT
```

```
iptables -A OUTPUT -o wlan0 -j ACCEPT
```

```
iptables -A POSTROUTING -t nat -o eth0 -j MASQUERADE
```

```
iptables -A FORWARD -i wlan0 -j ACCEPT
```

After creating this file, make it executable by using this command

```
sudo chmod +x /etc/network/if-up.d/router
```

Once that router.sh file has been run, it should start forwarding traffic from the WiFi Interface to the LAN interface!

### **sudo reboot**

If everything worked well, after reboot typing the command **ifconfig** you should see the interface wlan0 being assigned the IP address 10.0.0.1. If you don't see the IP Address try `sudo /etc/init.d/networking restart` and `/etc/init.d/hostapd stop/start`

Now, if you connect to the selected SSID on the Raspberry Pi you will be directly forwarded to the Internet connection (of your laptop).

Check if it works with iptables router file active

If you want to disconnect your Raspberry Pi from your laptop (unplug the ethernet cable) and connect through the WiFi interface you will need to remove the router file from `/etc/network/if-up.d/router`

## **Forwarding wireless traffic to a captive portal (before providing access to the Internet)**

In a previous section we generally enabled forwarding automatically when the RPi starts. Here we will keep only the following command in the `/etc/network/if-up.d/router` script::

```
iptables -A POSTROUTING -t nat -o eth0 -j MASQUERADE
```

or if you use the WiFi interface for the Internet

If you operate on offline mode ...

Comment previous commands in the router script that we added in the previous sections. Later in the guide, we will add the required commands to forward the traffic to the Captive portal where the web php files will run advanced and custom iptable rules.

a "splash" page is used for capturing the user IP and MAC address and display the page to user when they were redirected to this page by the portal system. Here's an example on how to capture user's IP and MAC address using PHP. You should add this code to your /var/www/index.php

```
<?php
// capture their IP address
$ip = $_SERVER['REMOTE_ADDR'];

// this is the path to the arp command used to get user MAC address
// from it's IP address in linux environment.
$arp = "/usr/sbin/arp";

// execute the arp command to get their mac address
$mac = shell_exec("sudo $arp -an " . $ip);
preg_match('/...../', $mac, $matches);
$mac = @$matches[0];

// if MAC Address couldn't be identified.
if( $mac === NULL) {
    echo "Access Denied.";
    exit;
}
?>
```

From the example above, user's IP and MAC address are stored in variables \$ip and \$mac respectively. For getting their MAC address, we uses 'sudo /usr/sbin/arp -an' command. Please note that for this command to work, the linux user 'www-data' (default user for Apache webserver daemon) must be in the sudoers file.

Add www-data as sudoers in linux box

First, you have to be root to add user as sudoers. Execute this command:

```
$ sudo visudo
```

Then add the following line at the bottom of the file, so that www-data user can execute arp command without entering a password

```
www-data ALL=NOPASSWD: /usr/sbin/arp
```

Save the file.

Create a simple HTML form (in index.php) for user to submit (and agree the portal Terms of Service if any).

Here's an example of the form in PHP:

```
<form method="post" action="process.php">
  <input type="hidden" name="mac" value="<?php echo $mac; ?>" />
  <input type="hidden" name="ip" value="<?php echo $ip; ?>" />
  <input type="submit" value="OK" style="padding:10px 20px;" />
</form>
```

User's IP and MAC address are embedded in the form as hidden field for further processing in the firewall. Save the file with name index.php in the document root directory of your web portal. So now it's accessible from a web browser.

To test if redirecting works, connect by your phone/computer to the “JukeBox” wireless network of the RPi, then type any website in the browser. The new splash page should appear. You can also test by typing the eth0 ip address of the RPi from a computer in the same Network.

### Redirecting every “unknown user” HTTP traffic to the “splash” portal

Here's the command on how to redirect every HTTP traffic to portal using iptables. Please note that you have to be root in order to be able to modify the iptables table. Note also that those rules should be added to the router script file in `/etc/network/if-up.d/router` to run all commands automatically when booting the

1. Create a new chain named 'internet' in mangle table with this command

**sudo iptables -t mangle -N internet**

2. Send all HTTP traffic from eth1 to the newly created chain for further processing

**sudo iptables -t mangle -A PREROUTING -i eth0 -p tcp -m tcp --dport 80 -j internet**

3. Mark all traffic from internet chain with 99

**sudo iptables -t mangle -A internet -j MARK --set-mark 99**

4. Redirect all marked traffic to the portal (10.0.0.1) which is the local ip address of the Apache web server (WiFi ip address specified before in the networking configurations)

**sudo iptables -t nat -A PREROUTING -i wlan0 -p tcp -m mark --mark 99 -m tcp --dport 80 -j DNAT --to-destination 10.0.0.1**

OK from now on, every HTTP request from wlan0 will be redirected to the portal page.

**Bypass firewall redirection rules when users submit themselves from the splash page**



When user is forwarded to the portal splash page, their IP and MAC address will be submitted to stored. Their MAC address will be “entered” into iptables firewall so that they won’t be redirected to the portal splash page anymore. The logic is, executing the following command from PHP so that the redirection logic in the firewall will be bypassed:

```
sudo iptables -t mangle -I internet 1 -m mac --mac-source USER-MAC-ADDRESS-HERE  
-j RETURN
```

And remove their “redirection” connection track. How? We can do that with a simple script. Create an executable file /usr/bin/rmtrack and put the following code inside

```
/usr/sbin/conntrack -L |grep $1 |grep ESTAB |grep 'dport=80' |awk '{ system("conntrack -D  
--orig-src $1 --orig-dst " substr($6,5) " -p tcp --orig-port-src " substr($7,7) " --orig-port-dst  
80"); }'
```

Change the file permission to 755 or 700 for better safety with the following command:

```
$ chmod 755 /usr/bin/rmtrack
```

With those rmtrack, we can remove user’s connection track with the following command

```
$ sudo rmtrack USER_IP_ADDRESS
```

Here’s an example of the php code:

```
<?php  
if( isset( $_POST['ip'] ) && isset ( $_POST['mac'] ) )  
{ $ip = $_POST['ip']; $mac = $_POST['mac'];  
exec("sudo iptables -I internet 1 -t mangle -m mac --mac-source $mac -j RETURN");  
exec("sudo rmtrack " . $ip);  
sleep(1); // allowing rmtrack to be executed // OK, redirection bypassed.  
// Show the logged in message or directly redirect to other website  
echo "User logged in.";  
exit; }  
else { echo "Access Denied"; exit;  
} ?>
```

Don’t forget to add the following line to the sudoers file so that the iptables and rmtrack command can be executed by the web server:

```
www-data ALL=NOPASSWD: /sbin/iptables www-data ALL=NOPASSWD: /usr/bin/rmtrack  
[0-9]*.[0-9]*.[0-9]*.[0-9]*
```

After the user had been logged in, they won’t be redirected to the portal again because of the iptables bypassing rules command. That’s it! Congratulations, A (very simple) captive portal using PHP and iptables has been successfully created.

## How to remove the user from the iptables bypassing rules so that they have to be authenticated again?

Here's on how to do it. Delete user's bypassing iptables rules with the following command (run it from the Raspberry Pi terminal):

```
sudo iptables -D internet -t mangle -m mac --mac-source USER_MAC_ADDRESS -j RETURN
```

and remove their connection track (again):

```
sudo rmtrack USER_IP_ADDRESS
```

if you know user's IP address, both command can be easily done by a simple PHP script below:

```
<?php
// get the user IP address from the query string
$ip = $_GET['ip'];
// this is the path to the arp command used to get user MAC address
// from it's IP address in linux environment.
$arp = "/usr/sbin/arp";
// execute the arp command to get their mac address
$mac = shell_exec("sudo $arp -an " . $ip);
preg_match('/...../', $mac, $matches);
$mac = @$matches[0];
// if MAC Address couldn't be identified.
if( $mac === NULL ) {
    echo "Error: Can't retrieve user's MAC address.";
    exit;
}
// Delete it from iptables bypassing rules entry.
while( $chain = shell_exec("sudo iptables -t mangle -L | grep ".strtoupper($mac) ) !== NULL )
{
    exec("sudo iptables -D internet -t mangle -m mac --mac-source ".strtoupper($mac)." -j RETURN");
}
// Why in this while loop?
// Users may have been logged through the portal several times.
// So they may have chances to have multiple bypassing rules entry in iptables firewall.
// remove their connection track.
exec("sudo rmtrack " . $ip);
// remove their connection track if any echo "Kickin' successful."; ?>
```

Save it to a file named kick.php in your web server document root. Get into your web browser and put the following URL to the address bar to kick someone from the iptables bypassing rules:

[http://192.168.3.1/kick.php?ip=USER\\_IP\\_ADDRESS](http://192.168.3.1/kick.php?ip=USER_IP_ADDRESS)

And you're DONE!

Feel free to modify the code to suit your requirements.

## **How to use two WiFi cards on the RPi to avoid using Ethernet:**

In the previous configurations, we used one wifi interface to broadcast a local network to users, and the Ethernet interface eth0 for internet access. In this part, we will explain how to use another WiFi card wlan1 for internet access.



First you need to define a new network interface in /etc/network/interfaces:

Add the following lines:

### **Case 1 (Open wireless network):**

```
auto wlan1
allow-hotplug wlan1
iface wlan1 inet dhcp
wpa-ssid "SSID_NAME"
wpa-key-mgmt NONE
wpa-scan-ssid 1
```

## Case 2 (Wireless network with a pre-shared key PSK)

If the Wlan SSID that you would like to connect through has a different key management and requires a password, Add the following lines:

```
auto wlan1
allow-hotplug wlan1
iface wlan1 inet dhcp
wpa-ssid "SSID_NAME"
wpa-key-mgmt WPA-PSK
wpa-scan-ssid 1
wpa-psk <network_key>
```

hint: you need to type the SSID in quotations as shown

## Case 3 (Enterprise wireless network with username/identity and password)

```
auto wlan1
allow-hotplug wlan1
iface wlan1 inet dhcp
wpa-ssid "SSID_NAME"
wpa-key-mgmt WPA-EAP
wpa-scan-ssid 1
wpa-identity <your_identity>
wpa-password <your_password>
```

**CHECK here for hints:** <http://ubuntuforums.org/showthread.php?t=318539>  
<http://www.raspberrypi.org/forums/viewtopic.php?p=353961#p353961>

Second, you will need to change the directing iptable rules in the /etc/network/if-up.d/router

Change the MASQUERADE rule so that you masquerade the interface wlan1 instead of eth0. This way, traffic will be directed from wlan0 to wlan1 which is connected to a wifi network with internet access.

## Restricting access to the Internet using Iptables and Traffic shaping:

**\*\*Note\*\* All command are run as root. If you are not running as root, prepend sudo to all commands**

## Setting up tc

tc is the program that is in charge of setting up the shaping rules.

### Step 1

Firstly, we will setup the default rule for the interface, which is wlan0 in this instance.

These 2 commands sets the default policy on wlan0 to shape everyone's download speed to 64 kilobytes a second.

```
tc qdisc add dev wlan0 root handle 1:0 htb default 10
```

```
tc class add dev wlan0 parent 1:0 classid 1:10 htb rate 64kbps  
ceil 64kbps prio 0
```

### Step 2

Next, we'll setup another class to shape certain addresses to a higher speed.

We also need to setup a filter so that any packets marked as such go through this rule

```
tc class add dev wlan0 parent 1:1 classid 1:5 htb rate 256kbps  
ceil 256kbps prio 1
```

```
tc filter add dev wlan0 parent 1:0 prio 1 handle 5 fw flowid 1:5
```

Once that class is setup, we'll need to setup iptables to mark the specific packets we want to shape as such.

## Setting up iptables

### Step 1

Firstly, we'll create the mangle table that we need. I've used a custom chain in the mangle table in this snippet

The below code creates the new chains of shaper-in and shaper-out, and then sets up some rules for any packets coming in and out of wlan0 and eth0 to go through the new chains.

```
iptables -t mangle -N shaper-out
```

```
iptables -t mangle -N shaper-in
```

```
iptables -t mangle -I POSTROUTING -o wlan0 -j shaper-in
```

```
iptables -t mangle -I PREROUTING -i wlan0 -j shaper-out
```

```
iptables -t mangle -I PREROUTING -i eth0 -j shaper-in
```

```
iptables -t mangle -I POSTROUTING -o eth0 -j shaper-out
```

## Step 2

Once that is done, we can then setup the packet marking so that any packets from the 10.0.0.0/24 subnet gets marked with a 1, otherwise if the IP address is 10.0.0.5, they will get marked with a 5

```
iptables -t mangle -A shaper-out -s 10.0.0.0/24 -j MARK  
--set-mark 1
```

```
iptables -t mangle -A shaper-in -d 10.0.0.0/24 -j MARK  
--set-mark 1
```

```
iptables -t mangle -A shaper-out -s 10.0.0.5 -j MARK --set-mark  
5
```

```
iptables -t mangle -A shaper-in -d 10.0.0.5 -j MARK --set-mark 5
```

With that done, any connections going through wlan0 should now be shaped to 64kbps unless you have the IP address of 10.0.0.5 in which case you will be shaped to 256kbps

If more shaping speeds or IP addresses are required, then step 2 in the respective sections will need to be modified / added onto to give you the extra options.

I used Speedtest Mini hosted on my own webserver to test the speeds while I was shaping the connections as it provided an easy to use interface, but wget on a big file can also be used to test the speeds.

## Setting up router to auto-start

If you want the shaping to start automatically, put all the commands into 1 or 2 shell script files in /etc/network/if-up.d and they will be run automatically.

e.g. /etc/network/if-up.d/router and /etc/network/if-up.d/shaper

make sure they have both been chmodded to be executable by running this on both files

```
chmod +x /etc/network/if-up.d/router
```

## QUOTA PER MAC ADDRESS - iptables

You could that with iptables quota

<http://wiki.openwrt.org/doc/howto/netfilter>

some more info how to craft your custom firewall rules:

<http://varinderjhand.wordpress.com/2012/05/21/iptables-rules-to-limit-time-quota-based-access/>

the custom firewall rules:

```
iptables -N quota
```

```
iptables -A quota -m quota --quota 5368709120 -j RETURN
```

```
iptables -A quota -j reject
```

```
iptables -A zone_lan_forward -j quota -m mac --mac-source AA:D2:BD:1C:19:35 #  
example 1
```

```
iptables -A zone_lan_forward -j quota -s 192.168.0.4 #example 2
```

You can copy the line with the mac address or ip as many times if you like because it points to the newly created chain quota that does the actual quota checking. (also you don't need the ip and mac both)  
also rebooting the router resets the quota again as well, with traffic shaping and limiting the bandwidth available is a nicer solution in my opinion.  
Also if the quota is reached you could use iptables REDIRECT to redirect traffic on port 80 to a webserver (that doesn't have to run on port 80). to inform the quota has been reached.

For resetting the quota you could opt for restarting the firewall via cron (Scheduled Tasks)

```
0 0 1 * * /etc/init.d/firewall restart
```

this will restart the firewall at midnight first day of each month

<http://www.thegeekstuff.com/2009/06/15-practical-crontab-examples/>

some more info on traffic shaping:

<http://wiki.openwrt.org/doc/howto/packet.scheduler/packet.scheduler>

And a final note, since it is rather easy to change a mac or ip, you can also make a separate zone for them and assign it to a specific port on the router or a separate ssid on the wifi (or both). and monitor that instead.

Also 5GBytes translates to more or less 15 Kbps continues per month, damn with my dialup years ago i could download more.

opkg update

opkg install iptables-mod-extra

### **More Hints (to be considered while building the Jukebox system):**

- In case you don't have a screen connected through HDMI to the RPi, initially you could follow the steps mentioned above to SSH the Rpi via the ethernet interface. But once the local wireless network works, you could then SSH the raspberry pi through that wifi network, and perform further required modifications. In our case, you can SSH the raspberry pi by writing the following command from any computer connected to the wlan0:

```
$ssh pi@10.0.0.1
```



- After modifying your network interfaces in `/etc/network/`, run the following command to apply changes:

**\$ sudo /etc/init.d/networking restart**

- From the hosts file on the RPI, you could change the hostname of the system by adding this line: **10.0.0.1 *jukebox***

10.0.0.1 is the ip address assigned before to the wlan0. and *jukebox/* written in the browser will direct the user to the home captive portal.

- Grant read and write permissions to all php files in the `/var/www` by running:  
**\$sudo chmod 777 var/www**

- To run a video on the RPi terminal, you can use this command:

**omxplayer -o hdmi videoname.type**

The video will played through hdmi, so make sure to direct the video output and the audio output to the hdmi by the following command:

**amixer cset numid=3 2**

--If you want to make advanced processing of image, you have to install PHP GD image processing library. For example, like the photo gallery applications.

Here is the command:`sudo apt-get update` `sudo apt-get -y install php5-gd`

## OFFLINE JUKEBOX:

In case there is no internet access through any interface, and you want to just operate the jukebox offline. Users will be just directed to the web page and the local services of the jukebox offline. Then you will need to remove the MASQUERADE iptable rule from `/etc/network/if-up.d/router`

