Building a standalone access point using a Raspberry Pi Zero W

by Barry Robinson

The Raspberry Pi Zero W is a small, single board computer (SBC) that has WiFi and Bluetooth connectivity built-in. It is cheap, at $10 U.S., and runs the Linux open-source operating system (OS). You can program the Pi in a variety of languages including Python and C++. The Pi has a 40-pin general purpose input/output (GPIO) that take boards called “hats.” A Pi hat is installed by pushing it onto the GPIO pins; easy! One example of a Pi hat is a motor controller; these are priced as low as $20 U.S. and can control several motors. In the following I’ll describe how to set up the Pi Zero as a standalone access point.

A standalone access point is a WiFi connection, usually to the Internet. In this case, the access point does not connect to the Internet but does connect to an Apache web server that can serve active (PHP) web pages and run CGI (Computer Gateway Interface) scripts. What good is this? It lets you control devices using any device which has WiFi and a web browser. You don’t need to download any special apps, just use web pages to do what you want. This means most smart phones, tablets, laptops and other devices. What you do, or can do, with this is left up to your imagination.

Note: The command "sudo" gives the user the permissions of the superuser. Since you own the Pi (and Pi doesn't know any better) you CAN logon as the superuser but this is generally not a good idea. The superuser can change anything and it is easy to make a mistake and mess up your operating system when you are logged in as the superuser. It won't physically damage the Pi but it may cost you time and cause you frustration. sudo is a safer way of executing some system commands since it only gives you permission once each time you call it. Think of sudo as (s)uper (u)ser (do). It prefaces the commands that it provides permission to run. For example:

```
sudo chmod 666 anyfile
```

In the above example sudo lets the user change the permissions on a file called “anyfile” using chmod. You’ll also see a program called nano; this is a simple text editor available on the Raspberry Pi operating system.

To build your own stand-alone access point you’ll need to:

- purchase a Raspberry Pi Zero W (built-in WiFi and Bluetooth)
- download and install PuTTY on your computer. This is a free (GPL license) terminal program.

- purchase an 8GB SD micro card
- get Raspian lite image. Raspian is a distribution (“distro”) of Linux built for the Raspberry Pi. The “lite” refers to the small size and simplicity of this particular distro. Format the micro SD card then image the SD card with the Raspian image. I use SDFormatter and Win32DiskImager; both are Windows programs.

Raspian lite is a command-line based system; no fancy-pants graphical user interface (GUI) that will slow down the OS. The Pi is quite impressive for what it is, but, it is not as powerful as a current desktop or laptop computer.

Using your computer, open the image on the SD. If you’ve used your computer to put the image on the micro SD card, you can just leave it in the card reader and open it from there. I'm using Windows; if you're using something else you will have to figure it out for yourself. The image should open in the boot folder.

Right click with mouse and select **`"New" > "Text file"`**

Name the file **`"wpa_supplicant.conf"`**. **<u>DO NOT USE</u> the ".txt" extension**, use ".conf" instead. The program will ask you "If you change a file name extension, the file might become unusable. Are you sure you want to change it?" Click on "Yes" to accept the file name change.

Click on the **`wpa_supplicant.conf`** file you've created; it should open in Notepad. Add the following lines:

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=CA

network={
ssid="SSID"
psk="PASSWORD"
scan_ssid=1
}
```

The SSID and PASSWORD can usually be found on your wireless modem. These two items must be in quotes as shown. Do not add any spaces between the "=". The country code given is for Canada, change for the country where you are. Save the file and close Notepad.

Right click again and select **`"New" > "Text file"`**

Name file **`"ssh"`**. There is no file extension; DO NOT USE the ".txt" extension. The program will prompt you to accept the change. Leave the ssh file empty.

Find the `"config.txt"` file; it already exists. Now click on the `"config.txt"`; it will open in Notepad. You may find it easier to open this file in Wordpad due to the formatting. Either program seems to work. Scroll down to the end of the file and add in the two following lines:

```
# Enable UART
enable_uart=1
```

Now eject the SD card and insert into your Pi Zero W. Power up your Pi Zero W with a recommended 5vdc power supply connecting to the micro USB port near the end opposite of the SD card holder. The green LED should begin blinking. Wait 2 minutes for the OS to load, then connect to the Pi with PuTTY using the following setting:

- Port: 22
- Connectiion type: SSH
- Host name: pi@raspberrypi.local

You may be warned about a POTENTIAL SECURITY BREACH! especially if you've reset the Pi (the server's host key doesn't match the one that your terminal program has in the registry) Don't worry, just proceed. The login will prompt you for a password which is "raspberry" (without quotes). Enter the password. You are now logged into your Raspberry Pi over a WiFi connection and you are connected to the Internet.

Now you can update and upgrade your system.

Note: The system may ask you to approve updates/upgrades that take space on your system. Answer yes. Each command below is run separately. The script raspi-config lets you configure your system.

```
sudo apt-get update
sudo apt-get upgrade
sudo raspi-config
```

Expand filesystem using "Advanced Options" from the menu.

- Check keyboard installation using "Localisation Options" from the menu. Correct as necessary.
- Enable I2C and Serial interfaces using "Interfacing Options".
- Finish raspi-config and reboot the Pi. You may be given the option to reboot by raspi-config or you may use the following command to reboot:

```
sudo shutdown -r now
```

Wait a couple of minutes for everything to reload then reconnect to the Pi using PuTTY (just like you have done above)

Now you're ready to install the Apache web server. Enter the following.

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

Once the Apache web server is installed, find your web address with the following (enter on the command line):

```
hostname -I
```

Enter the web address you get in your desktop browser and you should see the Apache home page. If you do, your web server is working.

Now add MySQL database server and Python support.

```
sudo apt-get install mysql-server python-mysqldb
```

Now you can install php for active web pages:

```
sudo apt install php php-mbstring
```

Now begin the setup as an access point:

```
sudo apt-get install dnsmasq hostapd
```

Then stop the new software from running (at least for now):

```
sudo systemctl stop dnsmasq
sudo systemctl stop hostapd
```

Now configure a static IP address. First, open the configuration file in nano:

```
sudo nano /etc/dhcpcd.conf
```

Add these two lines to the end of the file:

```
interface wlan0
static ip_address=192.168.4.1/24
```

Save the file and restart the dhcpcd service:

```
sudo service dhcpcd restart
```

Get rid of the dnsmasq configuration file and make a new one (use the command below):

```
sudo mv /etc/dnsmasq.conf /etc/dnsmasq.conf.orig
```

Open the new file with nano:

```
sudo nano /etc/dnsmasq.conf
```

Add the following lines to the end of the file and save:

```
interface=wlan0 # use wlan0 wireless interface
dhcp-range=192.168.4.2,192.168.4.20,255.255.255.0,24h
```

Configure hostapd by opening the file with nano:

```
sudo nano /etc/hostapd/hostapd.conf
```

And add this to the end of the file. This configures your access point:

```
interface=wlan0
ssid=nurse123
hw_mode=g
channel=7
wmm_enabled=0
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_passphrase=nurse123
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
```

Note: The SSID will show up as a wireless network on other devices. In the above I have made my SSID nurse123 and my wpa_passphrase nurse123 to be the same. I want to let people log into the access point very, very easily and I'm not concerned with security since I am not connected to a public network and I am usually using this in a restricted area. If you are concerned about security, don't do this!! You can choose a more secure SSID and passphrase if you wish.

Open:

```
sudo nano /etc/default/hostapd
```

Tell the program here to find the configuration file, modify the line `DAEMON_CONF` as follows:

```
DAEMON_CONF="/etc/hostapd/hostapd.conf"
```

Start these services one at a time:

```
sudo systemctl start hostapd
sudo systemctl start dnsmasq
```

Open the following:

```
sudo nano /etc/sysctl.conf
```

Uncomment (that is, remove the `#` from in front of the line) the following line:

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

Add a masquerade for eth0 (your hardwired connection if you are using one):

```
sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
sudo sh -c "iptables-save > /etc/iptables.ipv4.nat"
```

Open:

```
sudo nano /etc/rc.local
```

Add the following just before the `"exit 0"` line:

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

Rename your `wpa_supplicant.conf` to `wpa_supplicant.conf.orig` (the following is all on one line and is one command):

```
sudo mv /etc/wpa_supplicant/wpa_supplicant.conf
/etc/wpa_supplicant/wpa_supplicant.conf.orig
```

If you do not do this step, the Pi will still be connected to your original network! Once you enter the above in do this you will not be able to access the Internet unless you use a wired connection.

You’re almost finished. Reboot the Pi. Use:

```
sudo shutdown -r now
```

Wait about 2 minutes and then connect to the network. You use whatever you have programmed. In my example I use as my SSID nurse123 with the passcode nurse123

When you want to shut down your Raspberry Pi be sure to use shutdown **before** powering down the Pi, so you don't corrupt your SD card. Use:

```
sudo shutdown now
```

You can log in via a terminal program using SSH through port 22 at 192.168.4.1 as user pi@192.168.4.1 You can use the browser to see the webpage (index.php or index.html) at http://192.168.4.1

- Place your webpages (html or php) in the directory /var/www/html
- Your browser will always go to index.php or index.html as the home page.
- Create sub directories for images or downloads.
- Be sure to give yourself pi permission to use this directory.
- For those using CGI scripts/programs the CGI directory is at /usr/lib/cgi-bin

To transfer files using the following on any FTP client:
Host/IP/URL the IP address for your Pi, use **`hostname -I`** to find this
Your username/password. Your username should be "pi" unless you've changed it.

Port 22 (same as for SSH)
Connection type is SSH/SFTP

Assuming you have not changed your user name from "pi" do the following:

```
mkdir /home/pi/FTP
mkdir /home/pi/FTP/files
sudo chmod a-w /home/pi/FTP
```

When you connect with SFTP go to the /home/pi/FTP/files directory. The above commands give you permission to transfer files as "pi"

Have fun, be brave, and be creative.

Minor corrections made to this document on: 13 January 2019.