Setting up a Raspberry Pi[1]

by Kerry Veenstra
December 4, 2012

[1] Raspberry Pi is a trademark of the Raspberry Pi Foundation.

1.0 Introduction

The Raspberry Pi (www.raspberrypi.org) is a new, very popular single-board computer that we can use for basic embedded and networking projects. The board lets us benefit from the work of the countless volunteers who write the software for the complementary GNU (www.gnu.org) and Linux projects. We can use a Raspberry Pi board just as we can use nearly any other GNU/Linux box—although this fact does imply that we need to learn *how* to use a GNU/Linux box! So let's mention that first.

Reading this document will help you get started with a Raspberry Pi board, with the goal of accessing it remotely over a wired local network connection. The document is based in part on another document, the Raspberry Pi's Quick Start Guide, which can be found at www.raspberrypi.org/quick-start-guide . We assume that you have *some* knowledge of using GNU/Linux or Unix. If not, there are other publications that can help you.

In particular, take a look at the new publication *Getting Started with Raspberry Pi* by Matt Richardson and Shawn Wallace (ISBN 1-4493-4421-6, O'Reilly Media, Inc., December 2012). I have a preprint, and I plan to buy the final release. In addition to using a preprint of this book, I have discovered answers to many questions by simply doing a web search for the phrase "raspberry pi *thing-that-I-want-to-do.*"

For more advanced uses of the board, you'll need to begin the process of learning more about GNU/Linux. The Raspberry Pi is an excellent training platform for administering a GNU/Linux system. The best way to learn to administer it is by playing with it! It stores its filesystem on an SD Card, and if disaster strikes, you can start over at any time by reformatting the card.

2.0 Materials

This setup guide will provide you with a minimal system (although in GNU/Linux that actually is quite extensive!). Here is the list of what we are assuming you have:

- ☐ Raspberry Pi board
- ☐ 1-amp USB power supply
- ☐ USB cable
- ☐ 4-GB SD memory card (any size 2 GB or larger should work)
- ☐ USB-to-SD adapter for 4-GB or larger memory cards *or* a computer that can write SD Cards
- ☐ USB keyboard
- ☐ Video monitor (*not* a computer monitor) for either RCA composite or HDMI signals
- ☐ Video monitor cable
- ☐ Computer with Internet access

Although the Raspberry Pi can use a USB mouse, we do not because these instructions do not use the GUI interface.

3.0 Procedure

The setup procedure described here is divided into several sections. After completing the steps in all of the sections, you will have a Raspberry Pi that can access your home wired network. You will not be *finished*, however: you will be ready to *begin*!

3.1 Load GNU/Linux on the SD Memory Card

Some Raspberry Pi vendors will sell you an SD memory card that is preloaded with GNU/Linux. There is nothing wrong with this, and it will save you the steps in this first section, but it also will cost you about $4 additional (at the time of this writing).

Assuming that you need to load GNU/Linux on a blank SD memory card, follow the instructions on the web page

http://elinux.org/RPi_Easy_SD_Card_Setup

For your convenience (actually, for *our* convenience), we've included steps from that web page below for using an Apple computer. Check the web page just mentioned for using other computers.

Note: These commands and actions need to be performed as a computer user that has administrator privileges. Even if you don't know what that means, it's quite possible that you already have such privileges, but a web search may help you understand.

- ☐ (1) Download the Raspberry Pi's SD-Card image. See the URL http://www.raspberrypi.org/downloads . Get the most recent version of the file.

Some commands below are typed into a Mac's terminal program, but most of the commands are typed into the Raspberry Pi's keyboard. To remind you, the Mac commands are prefixed with `mac$`, while the Raspberry Pi's commands are prefixed with just `$` .

- ☐ (2) Start the Mac OS X Terminal application.
- ☐ (3) Optional. Verify the file that you just downloaded. In the terminal, use the `shasum` command to print the file's *Secure Hash Algorithm* checksum and compare it to the SHA checksum on the download website. The example command below underlines the name of the file that I had downloaded. Substitute the name of the file that you downloaded.

```
mac$ shasum ~/Downloads/2012-10-28-wheezy-raspbian.zip
```

- ☐ (4) Assuming that you downloaded the `.zip` file, extract the GNU/Linux image from it. Substitute the name of the file that you downloaded.

```
mac$ unzip ~/Downloads/2012-10-28-wheezy-raspbian.zip
```

The extracted file should have a `.img` extension.

We need to know the name that the operating system uses for the SD Card. We'll determine that by comparing the mounted volumes before and after inserting the card.

- [ ] (5) Ensure that the SD Card reader is **disconnected**.
- [ ] (6) From the terminal run the *Disk Free* command.

```
mac$ df -h
```

We'll use the response of the command momentarily.

- [ ] (7) Connect the SD Card reader with the SD Card inside. Wait a moment for the Mac to notice the card.
- [ ] (8) Rerun the *Disk Free* command.

```
mac$ df -h
```

- [ ] (9) Compare the results of the two `df` commands. Look for the device that appeared after the SD Card reader was inserted. If there is no difference, maybe the operating system didn't notice the SD Card yet, so wait a moment and then run `df` again.
- [ ] (10) Record the device name of the SD Card's partition, e.g. `/dev/disk1s1` .

The SD Card's partition name is `/dev/disk___`

- [ ] (11) Unmount the partition so that you will be allowed to overwrite the SD Card.

```
mac$ sudo diskutil unmount /dev/disk___
```

- [ ] (12) Starting with the device name of the partition `/dev/disk___` work out the *raw* device name for the entire SD Card by omitting the partition name (probably a final `s1`) and replacing `disk` with `rdisk` .

This is very important! You could lose all data on the hard drive of your computer if you accidentally use <u>its</u> name in the steps that follow!

Record the raw device name of the SD Card, e.g. `/dev/rdisk1` .

The SD Card's raw device name is `/dev/rdisk_`

- [ ] (13) Verify that the device name is the name of the whole SD card as described above and not just a partition of it (for example, `rdisk3`, not `disk3s1`). Be sure that you don't have another SD drive like `rdisk2` or `rdisk4`. If you have any doubts, recheck by using the `df -h` command both before & after you insert your SD Card reader into your Mac.

- [ ] (14) Write the GNU/Linux `*.img` file to the SD Card using the command below. The parameter `if=____.img` represents the name of the GNU/Linux image file that you downloaded and extracted earlier. The parameter `of=/dev/rdisk_` represents the name of the SD Card's raw device name, which you just recorded and verified. Substitute the appropriate names in the command. It is very important to use the correct `of=` parameter because it specifies the disk that gets overwritten.

  ```
  mac$ sudo dd bs=1m if=____.img of=/dev/rdisk_
  ```

  Note that dd will print an error if it is unsuccessful, but it will print nothing if it is successful. If you are curious, while dd is running, you can instruct the command to report its progress by typing Ctrl-T. (This is called sending a SIGINFO signal to the process.)

The dd command will take about five minutes to complete.

- [ ] (15) Type this command

  ```
  mac$ sudo diskutil eject /dev/disk___
  ```

  This command doesn't physically eject the card. It just tells Mac OS X that you are *about* to eject the card.

  Note: it is *super tempting* to take a look at the card's contents before ejecting it. Resist temptation! The Mac OS X operating system will write some hidden files on the drive if you access it using the Finder, and these files can interfere with its intended use under GNU/Linux. Just eject the SD Card immediately using the command above.

- [ ] (16) Remove the card physically from the SD Card reader.

We'll talk about how to transfer other files onto the SD Card later. Next, let's boot the Raspberry Pi and configure it for basic operation!

3.2 Basic Configuration

The first time you boot a Raspberry Pi that uses your newly formatted SD Card, it will run a utility called raspi-config. Below we tell you how to use this configuration utility.

Note that we do not want to boot the Raspberry Pi with connection to a network yet because its security is fairly low by default. Let's fix that first.

- [ ] (17) Ensure that the Raspberry Pi is **unpowered**.
- [ ] (18) Insert the SD Card into its SD Card socket.
- [ ] (19) Ensure that the network cable is **unplugged**.
- [ ] (20) Power up the Raspberry Pi by connecting it to the USB power supply using the USB cable (the board has no power switch). The raspi-config utility runs.

- [ ] (21) As it is now, the SD Card image is 2 GB. Run the option **expand_rootfs** to expand the GNU/Linux root file system to use the entire card. Use the downarrow key to move select the option, and press the Enter key to run it. Press Enter again to close the **<Ok>** screen that follows.

The GNU/Linux image assumes that you are using a UK-based keyboard instead of a US-based keyboard. Why is this important? Because you want to be able to type things like a dollar sign! Se let's tell the computer that you are using a US keyboard.

- [ ] (22) Select the **config_keyboard** option.
- [ ] (23) Select **Generic 104-key PC** (I'm assuming that that's what you have).
- [ ] (24) For the keyboard layout select **Other**.
- [ ] (25) Select **English (US)**.
- [ ] (26) Select **English (US)** at the top of the next list.
- [ ] (27) Select **The default for the keyboard layout**.
- [ ] (28) Select **No compose key**.
- [ ] (29) Regarding Ctrl-Alt-Backspace, select **<No>**.

We are going to remove the `pi` user later, but changing its default password is a good idea anyway.

- [ ] (30) Select the **change_pass** option and enter a new password for the `pi` user.

The "locale" of the computer determines its character set. By default, it uses UK English, but we want to use US English. In the steps below, you will add US English, but you will *leave UK English selected* because it is used by the running raspi-config utility.

- [ ] (31) Select the **change_locale** option.
- [ ] (32) Leave the option **en_GB.UTF-8 UTF-8** selected, and select the option **en_US.UTF-8 UTF-8** as well.
- [ ] (33) Tab to **<Ok>** and press Enter.
- [ ] (34) Select **en_US.UTF-8** for the default locale and press Enter.

The Raspberry Pi sets its clock from the network every time it boots (it lacks a battery-powered clock), and it needs to know your timezone. There are two Pacific timezones. I don't know why. I selected "Pacific.".

- [ ] (35) Select **change_timezone** and do the steps to enter your time zone.

I think that SSH already is enabled, but it doesn't hurt to be sure.

- [ ] (36) Select the **ssh** option and do the steps to enable SSH.

By default the Raspberry Pi starts a graphical user interface. I don't use it.

- [ ] (37) Select **boot_behaviour**. Select **<No>** so that the desktop does **not** start on boot.

We are done, but we don't want to boot just yet.

- [ ] (38) Using the Tab and Enter keys exit raspi-config but do *not* reboot. To do this you will select **<Finish>** followed by **<No>**.

If necessary, you can fix the video display's borders.

- [ ] (39) Edit the boot configuration file.

```
$ sudo vi /boot/config.txt
```

  You may not have seen the `sudo` command before. It acts like the `root` user and runs the rest of the command line, which in this case is the `vi` text editor. The first time you use `sudo`, it will prompt you for your password. It's just verifying that you really want to do this command.

  I hope that you already know the basics of the `vi` text editor! If not, then you'll need to learn it. It's used often when administrating a GNU/Linux system.

- [ ] (40) Set the left border to 6 pixels. You'll be editing four existing lines to look like these by removing the initial # character and changing the numbers after the = signs. If needed later, you can repeat these steps to adjust the borders.

```
overscan_left=6
overscan_right=0
overscan_top=0
overscan_bottom=0
```

Next let's turn up the network security before rebooting.

3.3 Basic Network Security

The `pi` user login probably is known by everyone who learns about Raspberry Pi. So for security, let's create a new user, confirm that we can log in as the new user, and then delete the `pi` user.

- [ ] (41) Choose a new user name. In this example, the name is `testbed`, but you can make it anything you want. Create that user with the following command. You can just press the Enter key for most questions, but you must choose a password.

```
$ sudo adduser testbed
```

☐ (42) Give the new user administrative privileges.

```
$ sudo visudo
```

☐ (43) Using arrow keys, go to the end of the file and add a line like this for *your* user.

```
testbed ALL=(ALL) ALL
```

Save the file with **Ctrl-O Enter** and exit with **Ctrl-X**.

☐ (44) Test the new user by logging out and then logging again as the new user.

```
$ logout
raspberrypi login: testbed
```

☐ (45) Remove the `pi` user.

```
$ sudo deluser -remove-home pi
$ sudo visudo
```

☐ (46) Using arrow keys and the Delete key, go to the end of the file remove the line

```
pi ALL=(ALL) NOPASSWD: ALL
```

Save the file with **Ctrl-O Enter** and exit with **Ctrl-X**.

3.4 Update GNU/Linux

The GNU/Linux image that you downloaded probably does not have all of the security updates. Usually it's best to get the updates.

☐ (47) Plug in the network cable.
☐ (48) Reboot the Raspberry Pi. This step will take a few minutes as the operating system finalizes the configuration changes. (The `-r` option tells the command to reboot the board instead of just shutting it down.)

```
$ sudo shutdown -r now
```

☐ (49) Log in as `testbed` (or whatever your new user is called).
☐ (50) The first step is to "update":

```
$ sudo apt-get update
```

The udpate step takes about 2 minutes.

☐ (51) The next step is to “upgrade”. After about a minute you’ll be asked a question. Answer “Y”.

```
$ sudo apt-get upgrade
Y
```

The upgrade step takes about 40 minutes.

At this point, if you need to install other modules, you can do so with the `apt-get` command. If you don’t need to install other modules, skip ahead to step 54.

☐ (52) Get the module that you desire. You may be asked a confirmation question.

```
$ sudo apt-get install module-name
```

☐ (53) At this point it is prudent to confirm that the Raspberry Pi still can boot. Check that now.

```
$ sudo shutdown -r now
```

Assuming that the board reboots correctly, log in again.

If there are problems, we know that they were caused by the update, the upgrade, or a module that you installed in steps 50, 51, or 52. You *could* start over with the `dd` command that formatted the SD Card and reboot after steps 50 and 51 to narrow down the location of the problem, but the operating system has log files that store diagnostic information. One of those log files may give you a hint for the cause of the problem. (Now is the time to start Googling “raspberry pi log files”!)

That said, if you don’t really *need* to use the latest upgrades, but there are problems booting after the upgrades, then just repeat the steps of this document skipping the upgrades (steps 50 through 53 above).

3.5 Decide How to Use the Serial Port

Your Raspberry Pi board has a TTL-level (not RS-232) serial port. You can use it to log into the board over a serial terminal, or you can use it for other communication (or you can just ignore it). The steps below cover the software portion of the configuration. You still will need to determine whether you should get a hardware TTL/RS-232 converter.

3.5.1 Terminal Login

The serial port already is set up for logging in. If you want it for that (or if you don’t want to use it at all) then you are done here. Skip ahead to step 58.

3.5.2 Other Communication

If you want to use the serial port to monitor data or to control a peripheral, then you should disable its login messages. The steps below are from the URL http://www.hobbytronics.co.uk/raspberry-pi-serial-port .

- [ ] (54) To disable the login prompt, edit the `inittab` file:

```
$ sudo vi /etc/inittab
```

- [ ] (55) Search for `ttyAMA0`, and comment out its `getty` line. You will prefix the appropriate line with a # character. Probably you will edit the last line of the file. When you are done, the line will start with a # .

```
#T0:23:respawn:/sbin/getty -L ttyAMA0 115200 vt100
```

  Save the file and exit.

- [ ] (56) To suppress bootup info on the serial port, edit the `cmdline.txt` file:

```
$ sudo vi /boot/cmdline.txt
```

- [ ] (57) Remove all references to `ttyAMA0`. You will delete the portion of the file that looks like this:

```
... console=ttyAMA0,115200 kgdboc=ttyAMA0,115200 ...
```

  Save the file and exit.

3.6 Copying Files Onto Your Raspberry Pi

One way to copy files onto your Raspberry Pi is to write the SD Card using another computer. But on a Mac (which I'm using) this is not a good idea. The Mac OS X operating system writes hidden files onto the SD Card for Mac purposes. These files may trouble the GNU/Linux operating system. So instead, it's best to leave the SD Card plugged into the Raspberry Pi and transfer files over the network connection.

- [ ] (58) Display the network IP address of your Raspberry Pi. (Realize that this address is assigned when you reboot the board, and so it may change.)

```
$ ifconfig eth0
```

  Look through the command's output for a string like one of these, with digits instead of # characters. The phrase `inet addr` means *internet address*.

```
inet addr:10.###.###.##
inet addr:172.##.###.##
inet addr:192.168.##.##
```

The string of digits and periods following the colon (for example `192.168.1.3`) is the IP address of the board's network connector. Record this address for the following steps.

On a Mac, you will use the `scp` program to copy files between your computer and the board. Let's start with copying a file from your computer to your Raspberry Pi board.

- ☐ (59) *Mac to Raspberry Pi:* On your Mac, in the terminal, go to the directory that contains the file. Let's say that it's called `file` and that your Raspberry Pi's login name is `testbed` and that the IP address of the Raspberry Pi's network connector is `192.168.1.3` . Then to copy the file to the login directory of user `testbed`, you type this into your Mac's terminal:

```
mac$ scp file testbed@192.168.1.3:.
```

Then you can log into your Raspberry Pi and confirm that the file is in the login directory of the `testbed` user. If desired, you can move the file anywhere else that it needs to be.

Now let's copy a file from the login directory of the Raspberry Pi board to your computer.

- ☐ (60) *Raspberry Pi to Mac:* On your Mac, in the terminal, go to the directory that should receive the file. Once again, let's say that the file is called `file` and that your Raspberry Pi's login name is `testbed` and that the IP address of the Raspberry Pi's network connector is `192.168.1.3` . Then to copy the file from the login directory of user `testbed`, you type this into your Mac's terminal:

```
mac$ scp testbed@192.168.1.3:file .
```

The commands that we've just shown copy files between the current directory of a Mac computer and the login directory of a Raspberry Pi user.

3.7 Turning Off Your Raspberry Pi

How often do you boot your computer? If you have a modern computer, it probably sleeps when you walk away from it. When you *do* turn it off, it automatically shuts down the operating system before turning off its own power. But think back to a time years ago when you needed to remember to *shut down* your computer before turning it off.

Well, welcome back to those times. You must shut down your Raspberry Pi before removing its power!

- [ ] (61) We've already seen the command that shuts down the Raspberry Pi. You used it to reboot. Here's how to shut down the board and make everything safe for you to remove its power.

  ```
  $ sudo shutdown -h -P now
  ```

  The `-h` option says to halt the processor (I think it actually enters an infinite do-nothing loop). The `-P` option tells the board to power down.

  Now you can remove the board's power. Remember, the board has no power switch. I keep the USB power supply plugged into a switched power strip.

4.0 Last Words

At this point, you have your own tiny GNU/Linux computer! Now what? The publication that I mentioned in the Introduction, *Getting Started with Raspberry Pi* by Richardson and Wallace, looks like it will cover the necessary steps to interact with the Raspberry Pi board as an embedded processor. But you don't *need* to do that.

The Raspberry Pi board will function as a fine web server, for example! (Not a high-performance one, but a working one.) Mine runs a MySQL client and routes data from its serial port into a database server on the local network. The Raspberry Pi can do many of the things that an ordinary GNU/Linux computer can do; using the Raspberry Pi like a GNU/Linux computer will guide you to learn about system administration.

All of the O'Reilly books about Linux are available for use, but since Linux changes over time, you'll want to read about the distribution and release that you have. At the time of this writing, the Raspberry Pi recommends the Debian distribution and the *wheezy* release (the releases are named after the characters of the *Toy Story* animated films). So assuming that you have that, when you are Googling or reading books about Linux, be sure to watch for information that is specific to *wheezy* and Raspberry Pi.

And there you have it! Your GNU/Linux computer is ready for exploration.