Building a Home Security System with Raspberry Pi

Building a Home Security System with Raspberry Pi starts off by showing you the Raspberry Pi and how to set up the Linux-based operating system. It then guides you through connecting switch sensors and LEDs to the native GPIO connector safely, and how to access them using simple Bash scripts. As you dive further in, you’ll learn how to build an input/output expansion board using the I2C interface and power supply, allowing the connection of the large number of sensors needed for a typical home security setup.

In later chapters, we’ll look at more sophisticated topics such as adding cameras, remotely accessing the system using your mobile phone, receiving intrusion alerts and images by e-mail, and more.

By the end of the book, you will be well versed in the use of Raspberry Pi to power a home-based security system that sends message alerts whenever it is triggered and will be able to build a truly sophisticated and modular home security system.

Who this book is written for

This book is for anyone who is interested in building a modular home security system from scratch using a Raspberry Pi board, basic electronics, sensors, and simple scripts. It would be great if you have some basic soldering skills in order to build some of the interface modules.

What you will learn from this book

- Understand the concepts behind alarm systems and intrusion detection devices
- Connect sensors and devices to the on-board digital GPIO ports safely
- Monitor and control connected devices easily using Bash shell scripting
- Build an I/O port expander using the I2C bus and connect sensors and anti-tamper circuits
- Capture and store images using motion detectors and cameras
- Access and manage your system remotely from your mobile phone
- Receive intrusion alerts and images by e-mail
- Build a sophisticated multi-zone alarm system

In this package, you will find:

- The author biography
- A preview chapter from the book, Chapter 1 'Setting Up Your Raspberry Pi'
- A synopsis of the book’s content
- More information on Building a Home Security System with Raspberry Pi
About the Author

Matthew Poole is a systems engineer based near Southampton on the south coast of England with over 20 years of industry experience. After graduating in electronics and communications engineering, he went on to train as and to become an air traffic engineer for the UK Civil Aviation Authority, working on microprocessor-based control and communications systems.

Later, he became a software architect and mobile technology specialist, and worked for several consultancies and global organizations. He is now a partner at UK Mobile Media, a boutique systems consultancy focused on designing Bluetooth and other wireless systems, taking ideas from concept to prototype. He is also the director of technology for Mobile Onboard, a leading UK-based transport technology company that specializes in on-bus connectivity and mobile ticketing.
Preface

The Raspberry Pi is a powerful, low-cost, credit-card sized computer, which lends itself perfectly as the controller of a sophisticated home security system. Using the available on-board interfaces, the Raspberry Pi can be expanded to allow the connection of a virtually infinite number of security sensors and devices. The Raspberry Pi has the processing power and interfaces available to build a sophisticated home security system but at a fraction of the cost of commercially available systems.

*Building a Home Security System with Raspberry Pi* starts off by showing you the Raspberry Pi and how to set up the Linux-based operating system. The book then guides you through connecting switch sensors and LEDs to the native GPIO connector safely, and it also shows you how to access these using simple Bash scripts. As you dive further in, you'll learn how to build an input/output expansion board using the I2C interface and power supply, allowing the connection of the large number of sensors needed for a typical home security setup.

The book features clear diagrams and code listing every step of the way to allow you to build a truly sophisticated and modular home security system.

What this book covers

*Chapter 1, Setting Up Your Raspberry Pi*, starts out by taking our Raspberry Pi out of its box and preparing it for being the centerpiece of our home security system. Along the way, we will install and set up the operating system, connect our Pi to the network, and access it remotely. We'll also secure our Pi and make sure it can keep the right time.

*Chapter 2, Connecting Things to Your Pi with GPIO*, explores the GPIO port and the various interfaces it features. We'll look at the various things we can connect to the Raspberry Pi using the GPIO including switches and sensors as we start to build our home security system.
Chapter 3, Extending Your Pi to Connect More Things, looks at ways of expanding the number of things we can connect to our Raspberry Pi, overcoming the limitation of having just the eight digital pins available to us on the GPIO by tapping into other interfaces on the GPIO and building our own input/output expansion board.

Chapter 4, Adding a Magnetic Contact Sensor, starts to actually connect things to our home security system, such as magnetic sensors and other types of contact devices. You will learn how to program our I2C expansion port using Bash scripts so that we can read the state of our sensors and switch on warning LEDs. We'll also start to develop the control scripts for our system that will allow us to arm and disarm the system and add delay timers.

Chapter 5, Adding a Passive Infrared Motion Sensor, looks at passive infra-red motion detectors, how they work, and how we can connect wired and wireless types to our home security system. We'll also learn how to create log files based on events using Bash scripts so that we can maintain a history of detector states as they change.

Chapter 6, Adding Cameras to Our Security System, teaches you how to connect both Raspberry Pi camera modules and USB cameras to our Pi board in order to take image and video captures when required by our home security system. We'll also learn how to overlay our images with informative text and have the files immediately emailed to us.

Chapter 7, Building a Web-Based Control Panel, gets down to the business of starting to put together modules by building a mobile-optimized web-based control panel for our home security system. You'll learn how to set up a web server on our Raspberry Pi and manipulate files using our web control panel, meaning we'll start to explore how all of the elements so far will come together as part of our final system.

Chapter 8, A Miscellany of Things, looks a few other bits and pieces, such as adding other sensors to our home security system that are not necessarily related to intruder detection. We'll also look at how we can administer our entire Raspberry Pi system remotely using a web browser in addition to accessing our home security control panel.

Chapter 9, Putting It All Together, is the moment we've all been waiting for; we're going to take all of the elements and concepts from the previous chapters and put together our full system comprising the elements we want to feature. The star of the show will be our Bash scripts, which will glue together all of these elements and provide the control logic for the entire system.
Setting Up Your Raspberry Pi

Before we can get into the realms of building our home security system, there's a bit of preparation work needed to get our Raspberry Pi up and running. So, we're going to go through the initial steps needed to get our Pi ready to be worked on.

In this chapter, we will cover the following topics:

• Exploring the different versions of the Raspberry Pi that are available
• Choosing the right Raspberry Pi version for your system
• Preparing the SD Card with the Raspbian Operating System
• Learning how to remotely access the Raspberry Pi over your home network
• Updating our operating system with the latest packages
• Exploring the time-keeping options on the Raspberry Pi
• Setting the user and root passwords to secure our Raspberry Pi

Which flavor of Pi?

Since the Raspberry Pi was released in 2012, there have been several versions of the mini-PC board released. I'll go though each of the versions released with their respective features so that you can choose which one is suitable for your particular project.
Setting Up Your Raspberry Pi

The good news is that it doesn't really matter which version you use in terms of power, as our home security system doesn't necessarily need loads of processing power, depending on what you want your system to do, of course. You might have an older board kicking about that will work for you.

The other piece of good news is that the GPIO interface pin layouts are the same across all the versions. The later versions have more pins, but the original 26 pins still remain in the same place.

The latest Raspberry Pi Version 2
Raspberry Pi Model A

The baby of the family is the Model A; it was released as a lower-cost version of the Model B, shown in the following section. Its main differences from the Model B are that it features just 256Mb of memory and has no Ethernet port; so if you want to connect this board to a network, you are limited to using a USB Wi-Fi dongle.
Raspberry Pi Model B

This was the first version of Raspberry Pi to be released; an updated revision, which improved the power system and USB port protection, came later. It features 512Mb of memory and has an Ethernet port for connecting to your network. This is probably the most common version used, and having the Ethernet port is incredibly useful, especially to get up and run quickly in order to set up and configure your Pi without the need for a keyboard and monitor.

The Raspberry Pi Model B Layout

Raspberry Pi Model B+ and Model 2

In 2014, the Raspberry Pi Foundation released a new version of the board that had some fundamental changes as compared to the previous version. The most fundamental changes were the board layout, form factor, and mounting points—much to the dismay of the many enclosure and accessory manufacturers out there.

In fact, we were in the middle of prototyping an enclosure for a commercial product that we were developing based on the Raspberry Pi—fortunately we caught wind of the board change in the nick of time and were able to change our enclosure to support the upcoming model B+.
The main electronic changes to this board are the addition of 2 more USB ports that can deliver more power to peripherals, an expanded GPIO interface, and the removal of the composite video port that is now consolidated into the audio jack. It also now uses a micro SD card with a better card slot.

In February 2015, a more powerful Raspberry Pi was released: the Raspberry Pi Model 2. It's similar to the Model B+ in terms of form-factor and interfaces, but is now reportedly 6-times faster than the Model B/B+ with its upgraded ARM processor and 1Gb of memory.

At the same low cost of less than £30, it's a fantastic little board and a great powerhouse for embedded systems.

Model comparison table

<table>
<thead>
<tr>
<th></th>
<th>Model A</th>
<th>Model B</th>
<th>Model B+</th>
<th>Version 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor</td>
<td>ARM1176JZF-S 700 MHz processor, VideoCore IV GPU</td>
<td>Quad-core ARM Cortex-A7 CPU and a VideoCore IV dual-core GPU</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Memory</td>
<td>256Kb</td>
<td>512Kb</td>
<td>512Kb</td>
<td>1Gb</td>
</tr>
<tr>
<td>USB Ports</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>
Setting Up Your Raspberry Pi

<table>
<thead>
<tr>
<th></th>
<th>Model A</th>
<th>Model B</th>
<th>Model B+</th>
<th>Version 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ethernet</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>No. GPIO Pins</td>
<td>26</td>
<td>26</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>Storage</td>
<td>SD Card</td>
<td>SD Card</td>
<td>Micro SD Card</td>
<td>Micro SD Card</td>
</tr>
</tbody>
</table>

So which one?
Essentially, any version of the Raspberry Pi will work with the modules presented in this book, but if you want to exploit features such as the camera, which may require more processing power and memory, or want to have an Ethernet connection, you’ll need to use the Model B.

If you want to start plugging additional stuff into the USB port, such as a GSM modem, then I recommend that you use the Model B+ as it delivers more power to those kinds of devices without the need for additional USB hubs.

If you want to do more processing with video and images from an attached camera, or want to connect lots of things, then go for the latest Model 2 board. I’m going to assume that’s the one you have chosen for this project, and that’s the one I’ll be using throughout this book; just be aware of any limitations if you choose to use an earlier model.

The Raspberry Pi Foundation site has more detailed information about each model. You can visit it at https://www.raspberrypi.org/products.

Preparing the SD card
The Raspberry Pi only boots from an SD card (or micro SD card for the B+ and v2 models), and cannot boot from an external drive or USB stick (well that’s not strictly true, but is beyond the scope of this book).

It’s recommended that you use a Class 10 SD card for performance, but a Class 4 or 6 card will be fine for this project. You’ll need to have a minimum card size of 4Gb.

Now that we have our Raspberry Pi board and SD card to hand, we need to prepare the SD Card specifically for our home security system. We’re going to use the standard Raspbian operating system as there really is no reason to use any other distribution; it’s the de facto choice for the Raspberry Pi.
Chapter 1

Downloading the Raspbian image

Download the Raspbian OS ZIP file containing the image to your PC.

At the time of writing, the latest version was Raspbian Jessie version 4.1 (2015-09-24-raspbian-jessie.zip).

Once downloaded, unzip the file and you'll have the file, 2015-09-24-raspbian-jessie.img.

The next thing to do is burn this image to your SD card...

Using Microsoft Windows
On a Windows PC, the best way to burn the image to your SD card is to use the Win32 Disk Imager utility. This can be downloaded from http://sourceforge.net/projects/win32diskimager.

The current version, at the time of writing, is 0.9.5.

It doesn't have an installer, and launches directly from the EXE file.

Now, it's time to create your SD card image:

1. Insert your SD card into the PC and launch the Win32 Disk Imager.
2. Select the SD card device drive letter (make sure it's right!).
3. Choose the Raspbian image file you've just downloaded.
4. Click on the **Write** button to create the SD card image.

Using Linux

On a Linux PC, you'll need to use the **gparted** and **dd** utilities to burn the image on your SD card.

Carry out the following steps to create your SD card image:

1. Extract 2015-09-24-raspbian-jessie.img to your **Home** folder.
2. Insert your SD card into the PC.
3. If you're not already in a shell terminal window, open one (you can use Ctrl + Alt + T on most graphical-based desktop systems).
4. Type the following command in the shell terminal:
   ```
   $ sudo fdisk -l
   ```
   In the list check, your SD card appears as a drive device (for example, /dev/sdb). It's crucial that you ensure you use the right device in the next step. We'll assume that your device is /sdb.
5. To burn the image to the SD card, type the following command:
   ```
   $ sudo dd if=2015-09-24-raspbian-jessie.img of=/dev/sdb
   ```
6. Hit **Enter** and go make a cup of tea or coffee as this will take a while. You'll know that it's finished when the command ($) prompt re-appears.
7. When the command prompt does re-appear, type the following command:
   ```
   $ sudo sync
   ```
8. Once that command has finished, you can remove the SD card from the PC.
Booting your Pi

You're now ready to boot up your Raspberry Pi. Pop your shiny new SD card into it and plug in the power.

Assuming that you have a monitor attached to your Pi, you should see your system booting up nicely. Although you could wait for it to boot up and connect to it via a terminal session (we'll look at that later), I recommend that you connect a monitor to it, at least in the first instance, just to make sure everything is working correctly.

In the new **Jessie** version of Raspbian, you'll boot straight into a desktop GUI, which is a major change from previous versions, where you'd be taken to the `raspi-config` utility, the first time the system is run, where you'd set up your Pi, and importantly, expand the file system to use the entire space available on your SD card.

![Debian Jessie boots into the GUI by default](image)

**Expanding the file system**

When you first create your Raspbian SD card, you'll only be left with about 200Mb of space in the file system, regardless of the size of your SD card. This is not much use, so we want to expand the file system so that it uses all of the available space on the card.

Fortunately, this is very easy on the Raspberry Pi now, as this function is available in the Raspberry Pi Configuration Tool on the desktop.
Setting Up Your Raspberry Pi

To access the new configuration tool, go to Menu and select Preferences | Raspberry Pi Configuration.

The new Raspberry Pi Configuration Tool

Goodbye GUI
Most of our work is going to be done in the command-line interface (CLI). Therefore, before we reboot the system in a minute, let’s change the Boot option by selecting To CLI, as shown in the previous screenshot, so boot into the command line going forward.

Anyway, now we click on the Expand Filesystem button, and in a couple of seconds, you'll see a confirmation message. The filesystem will be expanded when the system next reboots.

Using the raspi-config utility
If you have an older version of Raspbian, or you're not using the desktop GUI, then you'll need to use the raspi-config utility (which is still better than the old days when we had to do this manually in the shell). The first time you boot up, you'll be taken straight to the raspi-config utility.
The first option is the Expand Filesystem option; select this and you'll see various commands scrolling up the screen. Once it's finished, you'll see the following message:

Root partition has been resized.
The filesystem will be enlarged upon the next reboot

Click on OK.

Select Finish on the config screen and reboot your Pi when prompted.

After your Pi reboots with its fuller file system, you'll be taken straight to the shell prompt where you can log in with the default user and password.

**Login:** pi

**Password:** raspberry
Setting up your Pi

When you boot into the shell and have the Ethernet connected, hopefully the Pi will have connected to your home network and acquired an IP address from your router. If this is the case, you should see the **IP address** that has been issued just before the login prompt, as shown in the following screenshot:

As you can see from my screenshot, it's given me the IP address, **192.168.0.118**. This is good because I can now access the Pi remotely, using a secure shell (SSH) client to connect to it from the comfort of my laptop. This is particularly useful when my Pi is in the office and I want to sit on my sofa in front of the telly but still work on it, which I often do when I'm feeling lazy.
Downloading the example code

You can download the example code files from your account at http://www.packtpub.com for all the Packt Publishing books you have purchased. If you purchased this book elsewhere, you can visit http://www.packtpub.com/support and register to have the files e-mailed directly to you.

To do this, download PuTTY: a utility that allows you to connect to shell terminals remotely over the network. You can download it from http://www.putty.org.

Install and launch PuTTY and you're ready to connect to your Pi remotely from the comfort of your sofa.
Type the IP address of the Raspberry Pi into the Host Name box and click on Open. You'll be connected to your Pi in a remote terminal window. Once you've logged in, you can do pretty much everything on your Pi, as if you were sitting in front of it.

```
$ ssh pi@192.168.0.125
```

We’ll assume from now on that most of the work we do will be through a remote shell session, unless highlighted otherwise.

If you want to use the command line to launch the Raspberry Pi remote shell—for example, from another Linux system—use the following command from your terminal window:

```
# ssh pi@192.168.0.125
```
You'll then be prompted for the Pi's password and taken into a shell session.

![Shell session](image)

The Pi shell session launched from a Debian desktop terminal window

**Getting up to date**

Something that you should get into the habit of doing is updating the operating system regularly; even though you may have the latest image installed, it's very likely that there are updated packages available. To update your OS, enter the following command:

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

After this, enter the following one:

```bash
$ sudo apt-get dist-upgrade
```

This may take a while, depending on the number of updates required.

**Getting the right time**

The Raspberry Pi doesn't have on-board real-time clock hardware. This is one of the deliberate omissions to keep the cost of the board down. Instead, the Pi gets its time when it boots up from time servers on the Internet using the **Network Time Protocol (NTP)**. However, if there is no Internet connection at the time of booting up, then the time will be wrong.

In our security system, it's important that the time is kept accurate so that timestamps on log files and images are correct.
Setting Up Your Raspberry Pi

fake-hwclock
The fake-hwclock package is included in the latest Raspbian distributions, but in other past versions it wasn’t. If you need to install it, use the command that follows:

$ sudo apt-get install fake-hwclock

fake-hwclock is used by the Raspberry Pi to try and keep time when there is no network connection. It will regularly save the current time and restore it at boot-up. The obvious problem with this is that if the Pi has been switched off a few days, then the time will be set to the last time that it was on, using fake-hwclock.

If you want to see what time it last logged, type the following command:

$ cat /etc/fake-hwclock.data

ntp
The Network Time Protocol (NTP) is used when there is an Internet connection available and it can request the latest most accurate time from one or more time servers on the Internet.

By default, the ntp service is enabled on the latest Raspbian distribution, but it will initially get its time at boot-up from fake-hwclock if there is no Internet connection. There may be times when it’s necessary to force the ntp service to update from the Internet—for example, if the Internet connection is restored sometime after boot-up.

To force the ntp service to update from the Internet, use the following commands:

$ service ntp stop
$ ntpd -gq
$ service ntp start

Talking of security…
There’s no point in having a security system if the system itself is not secure. So, now we’ll change the default password for the pi user.

From the prompt, type the following command:

$ sudo passwd pi

pi@raspberrypi ~ $ sudo passwd pi
Enter new UNIX password:
Retype new UNIX password:
pwd: password updated successfully
What is this sudo thing anyway?

You'd have noticed that we've been putting `sudo` at the start of each command that we run in the terminal window. This is so that commands are run as the root user—the highest security level. This elevated security is required to perform many operations. `sudo` actually means super do.

If you can't be bothered to type `sudo` every time, then you can switch to the super user by typing the following:

```
sudo su
```

You'll see that the prompt changed from a `$` to a `#`, which indicates that you are now running as the root user.

So, this might be a good time to change the root user password too! To do this, type the following:

```
# passwd
root@raspberrypi:/home/pi# passwd
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
root@raspberrypi:/home/pi#
```

Connecting via Wi-Fi

You can also connect your Raspberry Pi to your network using Wi-Fi by plugging a USB dongle into it. There are additional configuration steps required to make this work, which are beyond the scope of this chapter, but there are many resources available covering this subject.

Summary

In this chapter, we took our Raspberry Pi out of its box and prepared it to be the centerpiece of our home security system. Along the way, we installed and set up the operating system, connected our Pi to the network, and accessed it remotely. We also secured our Pi and made sure it could keep the right time.

In the next chapter, we're going to explore the GPIO port and the various interfaces it features. We'll look at the various things we can connect to the Raspberry Pi using the GPIO port, including switches and sensors, as we start to build our home security system.
Where to buy this book

You can buy Building a Home Security System with Raspberry Pi from the Packt Publishing website.

Alternatively, you can buy the book from Amazon, BN.com, Computer Manuals and most internet book retailers.

Click here for ordering and shipping details.