Information: The Foundation of an App

The basis of all meaningful applications is information, and we design and build applications to exchange, create, or store it. Mobile applications are no different. In today’s well-connected mobile landscape, information exchange is the name of the game. To illustrate this point, imagine an Android phone without mobile network or WiFi coverage. While there would still be uses for such a phone, you would have lost access to some of the more important applications on your device. For example, e-mail, instant messaging, web browsing, and any other application that require the Internet would now be nonfunctional.

In later chapters, we will focus our efforts on examining information in transit and how to secure it. In this chapter, we will focus mostly on what happens to information that is stored.

Securing Your Application from Attacks

When created or received, data needs to be stored somewhere. How this information is stored will ultimately reflect on how secure your application really is. Releasing your application to the public should be approached with the same caution and paranoia as launching a website on the Internet. You should assume that your application will be either directly or indirectly attacked at some time and that the only thing standing between your end user’s privacy and data protection is your application.

Indirect Attacks

As dramatic as that last sentence sounds, it is not without basis. Before we go further, let’s take a look at whether my fear mongering is justified. In the latter part of 2010 and early 2011, two vulnerabilities were discovered in Android versions 2.2 and 2.3, respectively. The vulnerability is essentially the same one, in which an attacker can copy any file that is stored on the device’s SD
Card without permission or even without a visible cue that this is happening. The vulnerability works as shown in Figure 2-1.

Figure 2-1. Data theft vulnerabilities

The following are the most noteworthy points:

1. A user visits a malicious website hosting a file, such as evil.html.
2. Due to one part of the vulnerability, the evil.html file is downloaded and saved to the device SD Card without prompting the user.
3. Due to another part of the vulnerability, the saved file can be made to execute JavaScript code as soon as it is saved. Once again, there is no prompt to the end user.
4. Due to the final part of this vulnerability, the executed JavaScript from the preceding point, because it is running under the “local” context of the device, will have full access to upload files stored on the SD Card to a website of the attacker’s choosing.

For the sake of argument, assume that your application writes all saved information to the SD Card for storage under its own directory. Because of the vulnerability just discussed, the data used by your application is at risk of being stolen. Any Android device that runs your application and the vulnerable firmware versions poses a risk of data theft to its end user. This is an example of an indirect attack on your application.

How vulnerable your application is to an indirect attack depends largely on how much effort you put into architecting and considering security aspects before you begin writing a single line of code. You may ask the question, “I’m just a small app developer planning to sell my app for a low price online, so do I really need to waste time doing so much planning beforehand?” And I would answer you with a resounding, “Yes!” Whether you are part of a team of thirty developers or an individual working from home, a well-architected application is something you should always strive to create. I hope that this is what you will learn from this book.