Mobile Application Penetration Testing

Mobile Application Penetration Testing is an easy-to-follow guide with full of hands-on examples of real-world attack simulations. Each topic is explained in context with respect to testing, and for the more inquisitive, there are more details on the concepts and techniques used for different platforms.

This book gives you the necessary skills to security test your mobile applications as a beginner, developer, or security practitioner. You’ll start by discovering the internal components of an Android and an iOS application. Moving ahead, you’ll understand the inter-process workings of these applications. Then you’ll set up a test environment for this application using various tools to identify the loopholes and vulnerabilities in the structure of the applications. Finally, after collecting all the information about these security loopholes, we’ll start securing our applications from these threats.

Who this book is written for

If you are a mobile application evangelist, mobile application developer, information security practitioner, penetration tester on infrastructure web applications, an application security professional, or someone who wants to learn mobile application and pursue it as a career, then this book is for you. This book will provide you with all the skills you need to get started with Android and iOS pen-testing.

What you will learn from this book

- Gain an in-depth understanding of Android and iOS architecture and the latest changes
- Discover how to work with different tool suites to assess any application
- Develop different strategies and techniques to connect to a mobile device
- Create a foundation for mobile application security principles
- Learn techniques to attack different components of an Android device and the different functionalities of an iOS device
- Get to know secure development strategies for both iOS and Android applications
- Gain an understanding of threat modeling mobile applications
- Get an in-depth understanding of both Android and iOS implementation vulnerabilities and how to provide counter-measures while developing a mobile app
In this package, you will find:

- The author biography
- A preview chapter from the book, Chapter 1 'The Mobile Application Security Landscape'
- A synopsis of the book’s content
- More information on Mobile Application Penetration Testing
About the Author

Vijay Kumar Velu is a passionate information security practitioner, speaker, and blogger, currently working as a cyber security technical manager at one of the Big4 consultancies based in India. He has more than 10 years of IT industry experience, is a licensed penetration tester, and has specialized in providing technical solutions to a variety of cyber problems, ranging from simple security configuration reviews to cyber threat intelligence. Vijay holds multiple security qualifications including Certified Ethical Hacker, EC-council Certified Security Analyst, and Computer Hacking Forensics Investigator. He loves hands-on technological challenges.

Vijay was invited to speak at the National Cyber Security Summit (NCSS), Indian Cyber Conference (InCyCon), Open Cloud Conference, and Ethical Hacking Conference held in India, and he has also delivered multiple guest lectures and training on the importance of information security at various business schools in India. He also recently reviewed *Learning Android Forensics*, Packt Publishing.

For the information security community, Vijay serves as the director of the Bangalore chapter of the Cloud Security Alliance (CSA) and chair member of the National Cyber Defence and Research Center (NCDRC).
Preface

The adoption of mobile technology has changed the world, smartphones especially have become an integral part of everyone's lives and an extension of the corporate workplace.

With over a billion smartphone users worldwide, mobile applications play a crucial role in almost everything a device can do. Most of the time, the security of these applications is always an afterthought when data is the only asset that one would like to protect.

In short, the purpose of this book is to educate you about and demonstrate application security weaknesses on the client (device) side and configuration faults in Android and iOS that can lead to potential information leakage.

What this book covers

Chapter 1, The Mobile Application Security Landscape, takes you through the current state of mobile application security and provides an overview of public vulnerabilities in Android and iOS applications. It also teaches you the OWASP mobile top 10 vulnerabilities in order for you to establish a baseline for the vulnerabilities and principles of securing mobile applications.

Chapter 2, Snooping Around the Architecture, walks you through the importance of an architecture and dives deep into the fundamental internals of the Android and iOS architectures.

Chapter 3, Building a Test Environment, shows you how to set up a test environment and provides step-by-step instructions for Android and iOS devices within a given workstation.
Chapter 4, Loading up – Mobile Pentesting Tools, teaches you how to build the toolbox within your workstation required to perform an assessment of any given mobile app, and it also teaches how to configure them.

Chapter 5, Building Attack Paths – Threat Modeling an Application, shows you how to build attack paths and attack trees for a given threat model.

Chapter 6, Full Steam Ahead – Attacking Android Applications, shows you how to penetrate an Android application to identify its security weakness and exploit them.

Chapter 7, Full Steam Ahead – Attacking iOS Applications, shows you how to penetrate an iOS application to exploit the weaknesses and device vulnerabilities that affect the application.

Chapter 8, Securing Your Android and iOS Applications, teaches you the practical way of securing Android and iOS applications, starting from the design phase, and how to leverage different APIs to protect sensitive data on the device.
Chapter 1

The Mobile Application Security Landscape

Life is now in the palm of your hands. Risk is real, threats are growing!

With more than 1 billion users worldwide and 2.5 million applications (and still counting) available across Google and Apple digital marketplaces, smartphones have become commonplace. The difference they make to our lives is stark and simple, and is impacting our day to day life in multiple ways — in particular, the way we interact, work, and socialize. The increase in demand from consumer market and processing power and the capabilities of smartphones, such as storage, GPS, camera, displays, and so on, have changed the paradigm of the development of mobile applications. The ability to do online banking, trading, e-mails, airport check-ins, and much more is just a tap away.

Mobile application development is the hottest type of software development right now. New surface area equals dangerous surface area, which means that the uppermost layer of smartphones is mobile apps, which are the potential targets of adversaries.

This chapter will cover the current state of mobile application security. We will discuss some of the public vulnerabilities that are disclosed in various mobile applications in order to provide a context and reasons why security needs to be at the forefront of every mobile application developer’s mind. We will also cover the following topics:

- Android and iOS vulnerabilities
- Key challenges in mobile application security
- The impact of mobile application security
The Mobile Application Security Landscape

- The need for mobile application penetration testing
- The mobile application penetration testing methodology
- The OWASP (short for Open Web Application Security Project) mobile top 10 risks

There is no doubt that mobile applications have emerged as one of the most significant innovations of all time. Statista (for more information, visit http://www.statista.com/), a statistical portal company, reports that there are around 1.6 million applications in Google Play Store, 1.5 million applications in the Apple app store, 400,000 applications in the Amazon app store, 340,000 applications in Windows Phone Store, and 130,000 applications in Blackberry World. These statistics alone reflect the exponential growth in mobile applications over the years.

Numerous applications are introduced in stores every single week. At the same time, thousands of cyber criminals, also known as hackers, keep a tab on these applications by constantly looking for new applications that are published to the stores and try to compromise the user information or embed any malicious programs by various techniques. None of the development frameworks currently used are proven as immune to security issues.

The smartphone market share

Understanding the market share will give us a clear picture about what cyber criminals are after and also what could be potentially targeted. The mobile application developers can propose and publish their applications on the stores, being rewarded by a revenue sharing of the selling price.

The following screenshot referenced from www.idc.com provides us with the overall smartphone OS market, 2015:

<table>
<thead>
<tr>
<th>Period</th>
<th>Android</th>
<th>iOS</th>
<th>Windows Phone</th>
<th>BlackBerry OS</th>
<th>Others</th>
</tr>
</thead>
<tbody>
<tr>
<td>2015Q2</td>
<td>62.8%</td>
<td>13.9%</td>
<td>2.6%</td>
<td>0.3%</td>
<td>0.4%</td>
</tr>
<tr>
<td>2014Q2</td>
<td>64.8%</td>
<td>11.6%</td>
<td>2.5%</td>
<td>0.5%</td>
<td>0.7%</td>
</tr>
<tr>
<td>2013Q2</td>
<td>79.8%</td>
<td>12.9%</td>
<td>3.4%</td>
<td>2.8%</td>
<td>1.2%</td>
</tr>
<tr>
<td>2012Q2</td>
<td>66.3%</td>
<td>16.6%</td>
<td>3.1%</td>
<td>4.9%</td>
<td>6.1%</td>
</tr>
</tbody>
</table>

Source: IDC, Aug 2015

Since mobile applications are platform-specific, a majority of software vendors are forced to develop the applications for all the available operating systems.
The android operating system
Android is an open source Linux-based operating system for mobile devices (smartphones and tablet computers). It was developed by the Open Handset Alliance, which was led by Google and other companies. Android OS is Linux-based, and it can be programmed in C/C++, but most of the application development is done in Java (Java access to C libraries via JNI, short for Java Native Interface).

The iPhone operating system (iOS)
iOS was developed by Apple Inc. It was originally released in 2007 for the iPhone, iPod Touch, and Apple TV. Apple's mobile version of the OS X operating system used in Apple computers is iOS. BSD (short for Berkeley Software Distribution) is Unix-based and can be programmed in the Objective C and Swift languages.

Different types of mobile applications
In the modern realm, mobile applications are also called mobile apps. There are thousands of user-friendly apps on the market for most specific needs, starting from chatting, multi-video conferencing, games, health check-ups, gambling, communities, trading, other financial services, and so on and so forth.

One of the interesting future technologies in the mobile apps space is the development of mobile apps running on iOS and Android devices, where the app can listen for signals from beacons in the physical world and react accordingly, called iBeacon.

The apps are broadly categorized into the following types:

- Native apps
- Mobile web apps
- Hybrid apps
The Mobile Application Security Landscape

Native apps
Native applications that reside in the mobile operating system are pushed/installed through the respective app stores. These apps are typically built using development tools and languages (Xcode and Objective C, Swift for iOS apps, and Android Studio and Java for Android apps) and are designed for a particular platform and can take advantage of all the device features, such as the usage of the camera, GPS, phone contact list, and so on. The following screen capture of a well-known game is a solid example of a native mobile application:

![Temple Run 2](image)

Mobile web apps
Mobile web applications are non-native applications. Most of them are HTML5, JavaScript, and CSS applications with a web interface supporting the native application look and feel. Users first access them as they would access any other web page, and these are mobile-optimized web pages.
These applications became popular when HTML5 came around and people started to utilize the functionality of native applications from browser. The development and testing of these applications are easy since they all have tooling support.

The following screen capture shows one of the banking web applications:

Hybrid apps

Hybrid applications have two definitions. One definition is of a combination of web-based content and native components accessing services on the mobile device, most notably, storing or using storage. Another definition is of a client-server architecture of mobile applications. An example is a mobile enterprise application.
These are web apps built into native mobile framework and take advantage of the cross-compatibility of web technologies, such as HTML5, CSS, and JavaScript. The following is a screen capture of a well-known news mobile application, which is an example of a hybrid app:

![Screen capture of a mobile application](image.png)

**Why does it matter?**

The changes to the programming languages in order to develop applications force developers to maintain multiple code bases. Cyber attackers follow users; the mobile application threat scape has grown significantly grown over the years.
Public Android and iOS vulnerabilities

Before we proceed with the different types of vulnerabilities on Android and iOS, this section introduces you to Android and iOS as operating systems and covers various fundamental concepts that need to be understood in order to gain experience in mobile application security.

<table>
<thead>
<tr>
<th>Year</th>
<th>Android</th>
<th>iOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>2007/2008</td>
<td>1.0</td>
<td>iPhone OS 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>iPhone OS 2</td>
</tr>
<tr>
<td>2009</td>
<td>1.1</td>
<td>iPhone OS 3</td>
</tr>
<tr>
<td></td>
<td>1.5 (Cupcake)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2.0 (Eclair)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2.0.1 (Eclair)</td>
<td></td>
</tr>
<tr>
<td>2010</td>
<td>2.1 (Eclair)</td>
<td>iOS 4</td>
</tr>
<tr>
<td></td>
<td>2.2 (Froyo)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2.3-2.3.2(Gingerbread)</td>
<td></td>
</tr>
<tr>
<td>2011</td>
<td>2.3.4-2.3.7 (Gingerbread)</td>
<td>iOS 5</td>
</tr>
<tr>
<td></td>
<td>3.0 (HoneyComb)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3.1 (HoneyComb)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3.2 (HoneyComb)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4.0-4.0.2 (Ice Cream Sandwich)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4.0.3-4.0.4 (Ice Cream Sandwich)</td>
<td></td>
</tr>
<tr>
<td>2012</td>
<td>4.1 (Jelly Bean)</td>
<td>iOS 6</td>
</tr>
<tr>
<td></td>
<td>4.2 (Jelly Bean)</td>
<td></td>
</tr>
<tr>
<td>2013</td>
<td>4.3 (Jelly bean)</td>
<td>iOS 7</td>
</tr>
<tr>
<td></td>
<td>4.4 (KitKat)</td>
<td></td>
</tr>
<tr>
<td>2014</td>
<td>5.0 (Lollipop)</td>
<td>iOS 8</td>
</tr>
<tr>
<td></td>
<td>5.1 (Lollipop)</td>
<td></td>
</tr>
<tr>
<td>2015</td>
<td></td>
<td>iOS 9 (beta)</td>
</tr>
</tbody>
</table>

The preceding table comprises the operating system releases year after year.
The Mobile Application Security Landscape

An interesting research conducted by Hewlett Packard (HP), a software giant that tested more than 2000 mobile applications from 600+ companies, has reported the following statistics (for more details, visit http://www8.hp.com/h20195/V2/GetPDF.aspx/4AA5-1057ENW.pdf):

- 97% of applications tested access at least one private information source of those applications
- 86% of applications failed to use simple binary hardening protections against modern-day attacks
- 75% of applications do not use proper encryption techniques when storing data on a mobile device
- 71% of the vulnerabilities resided on the web server
- 18% of applications sent usernames and password over HTTP, while another 18% implemented SSL/HTTPS incorrectly

So, the key vulnerabilities to mobile applications arise due to the lack of security awareness, usability versus security trade-off by developers, excessive application permissions, and lack of privacy concerns. Couple this with a lack of sufficient application documentation, and it leads to vulnerabilities that developers are not aware of.

Usability versus security trade-off

For every developer, it is difficult to provide an application with high security and high usability. Making any application secure and usable takes a lot of effort and analytical thinking.

Mobile application vulnerabilities are broadly categorized into the following categories:

- **Insecure transmission of data**: Either the application does not enforce any kind of encryption for the data in transit on the transport layer, or the implemented encryption is insecure.
- **Insecure data storage**: Apps store the data in a plaintext or obfuscated format or hardcoded keys in the mobile device. An example e-mail exchange server configuration on an Android device using the e-mail client stores the username and password in the plaintext format, which is easy to reverse by any attacker if the device is rooted.
Chapter 1

- **Lack of binary protections**: Apps do not enforce any anti-reversing, debugging techniques.
- **Client-side vulnerabilities**: Apps do not sanitize data provided by the client side leading to multiple client-side injection attacks, such as cross-site scripting, JavaScript injection, and so on.
- **Hard-coded passwords/keys**: Apps designed in such way that hardcoded passwords or private keys are stored on the device.
- **Leakage of private information**: Apps unintentionally leaking private information; this could be the use of a particular framework and obscurity assumptions by the developers.

**Rooting/jail-breaking**

Rooting/jail-breaking refers to the process of removing the limitations imposed by the operating system on devices through the use of exploit tools. It enables users to gain complete control of the device operating system.

**Android vulnerabilities**

In July 2015, a security company called Zimperium announced that it has discovered a high risk vulnerability *Stagefright* (Android bug) inside the Android operating system. They deemed it as a *unicorn in the world of Android risk*, and it was practically demonstrated in one of the hacking conferences in the US on August 5, 2015. More information can be found at [https://blog.zimperium.com/stagefright-vulnerability-details-stagefright-detector-tool-released/](https://blog.zimperium.com/stagefright-vulnerability-details-stagefright-detector-tool-released/), and a public exploit is available at [https://www.exploit-db.com/exploits/38124/](https://www.exploit-db.com/exploits/38124/).

This has made Google release security patches for all Android operating systems, which is believed to be 95% of Android devices, an estimated 950 million users. The vulnerability is exploited through a particular library, which can let attackers take control of an Android device by sending specifically crafted multimedia services, such as MMS.

If we take a look at the Superuser and other similar application downloads from Play Store, there are around 10 million to 50 million downloads. It can be assumed that more than 50% of Android smartphones are rooted.

More features that are introduced to the operating system in the form of applications act as additional entry points that allow cyber attackers or security researchers to circumvent and bypass the controls that were put in place.

**iOS vulnerabilities**

On June 18, 2015, a password stealing vulnerability, also known as XARA (Cross Application Resource Attack), outlined for iOS and OS X cracked the Keychain services on jail broken and non-jail broken devices. The vulnerability is similar to the cross-site request forgery attack in web applications. In spite of Apple's isolation protection and its App Store's security vetting, it was possible to circumvent the security controls mechanism. It clearly provided the need to protect the cross-app mechanism between the operating system and the app developer. Apple rolled out a security update week after the XARA research. More information can be found at http://www.theregister.co.uk/2015/06/17/apple_hosed_boffins_drop_0day_mac_ios_research_blitzkrieg/.

The following graph shows the iOS vulnerabilities from 2007 until January 2016. There are around 805 reported vulnerabilities for Apple IPhone OS (http://www.cvedetails.com/product/15556/Apple-Iphone-Os.html?vendor_id=49).
As we can see, year after year, the vulnerabilities kept on increasing. A majority of the vulnerabilities reported are denial-of-service (DoS) attacks. This vulnerability makes the application unresponsive.

Primarily, the vulnerabilities arise due to insecure libraries or overwriting with plenty of buffer in the stacks.

**The key challenges in mobile application security**

Mobile security is not just about code running safely on the mobile device. Starting from the design, it also includes the residual data and data in motion.

Looking at the data and behavior of the application, any interesting mobile application will send back data to the server. Lots of applications use third-party web services. Some prevalent problems associated with data on different layers are mentioned as follows:

- **Network layer**: Data travelling from mobile applications from the device over Wi-Fi and data services
- **Hardware layer**: Baseband attacks, broadband attacks, and RF range attacks that can affect mobile features
- **Operating system layer**: Jailbreaking or rooting vulnerability in mobile platforms
- **Application layer**: API (short for Application Program Interface) of the device without administrative permissions
Since mobile apps are platform-dependent, the key challenges change from the traditional applications; some of the key challenges are as follows:

- **Threat Model**: Mobile applications that have a significantly complicated threat model cannot be the same for different versions of operating systems, devices, and manufacturers. We will discuss this in more detail in Chapter 5, *Building Attack Paths – Threat Modeling an Application*.
- **Third party code**: Developers including code developed by third-parties or open source.
- **Obscure assumptions by developers**: Assumes that the code is inherently secure.
- **Outsourcing**: Intellectual property. Part of the code or entire code is not available since it was outsourced.
- **Privacy of the data**: It is important to comply with regulations and end user's private data. How many third-party API's are integrated? Who collects what data?

**The impact of mobile application security**

Mobile applications put the security and privacy of an individual or corporation at risk. With more vulnerabilities attributed to mobile application flaws than any other category today, security has become a core concern for the business. Several attacks are associated with the way the mobile apps are used and the specific methods the app utilizes to communicate with the user.

Mobile applications can communicate over various services, which increases the attack surface significantly. Some of these services from which applications can obtain input are Bluetooth, **Short Message Service (SMS)**, microphone, camera, and **near field communication (NFC)**, to name a few.

The two primary impacts of mobile application security are **data at rest** and **data in motion**:

- **Data at rest**: Mobile applications are unique in the sense that they reside on the user's phone. As such, threats to these devices are primarily from mobile malware and other applications. Mobile devices are easily susceptible to theft, getting lost, or being acquired and used by someone else. Mobile app developers should also consider the possibility of data recovery using forensics techniques.

- **Data in motion**: Sensitive information disclosure and **man-in-the-middle (MiTM)** attacks are possible risks when the data is not secured in transit.
Other considerations: Mobile app developers should also consider the implications of malicious applications that are installed from various nonstandard app stores. Developers will always have the war game with the latest improvements in mobile malwares, such as Zeus MITMO, Spitmo, Citmo, Tatanga, which have bypassed plenty of mobile security features.

The need for mobile application penetration testing

Today's mobile apps have complex security landscapes; vulnerabilities might occur due to various reasons, starting from misconfiguration to code level bugs.

As the need for mobile applications is increasing, multiple companies ranging, from Fortune 500 to start-ups, are investing lots of money on security programs to protect critical information that is handy for every single individual at their fingertips. Naturally, the companies intend the applications to be secured. Their goal is to identify the loopholes while battling cyber attackers and prevent a serious data breach.

As discussed earlier about the importance of mobile applications, penetration test is one of the most effective ways to identify known and unknown weaknesses and functionality bugs (which will lead to a vulnerability) in these applications. By attempting to circumvent security controls and bypassing security mechanisms, a security tester is able to identify ways in which a hacker might be able to compromise an organization's security. Potentially, it leads to damaging the image of an organization that they have built over a period of time while building trust.

Current market reaction

The need for security in mobile applications has paved the market to create multiple job roles with respect to mobile security. Some of these job roles are as follows:

- Mobile Application Security Expert
- Mobile Security Compliance Specialist
- Mobile Technology Risk Manager
- Mobile Device Management Specialist
- Security Architect – Mobile Application
- Mobile Application Privacy Specialist
- Mobile Application Security Assurance Specialist
The mobile application penetration testing methodology

The mobile application penetration testing methodology is typically based on the application security methodology. The focus shifts from traditional application security, where the primary threat is from multiple sources over the Internet. The key difference is in the client-side security, filesystem, hardware, and network security. Traditionally for mobile applications, an end user is in control of the device.

Everything starts with understanding the risk environment of mobile applications.

Discovery

Information collection is an important point to keep in mind during the penetration testing process:

- **Open Source Intelligence**: It may be possible to find out more information about an application. This includes checking through search engines, third-party libraries that are used, or finding leaked source code through the use of source code repositories, developer forums, and social media.

- **Understanding the platform**: Understanding the platform is a crucial part of application penetration testing. This gives a clear understanding from an external point of view when it comes to creating a threat model for the application.

- **Client side vs Server side scenarios**: It is crucial to understand the type of application (native, hybrid, or web) and work on the test cases.
Analysis/assessment

Mobile applications have a unique way of assessment or analysis, and testers have to check the applications pre and post installation.

- **Static analysis**: Static analysis is performed, without executing the application, on the provided or decompiled source code and accompanying files. Sometimes, you might be provided with just the source code of the application.

- **Archive analysis**: The application installation packages for the Android and iOS platforms will be extracted and examined to review configuration files that have not been compiled into the binary.

- **Local file analysis**: When the application is installed, it is given its own directory in the filesystem. During the usage of the application, it will write to and read from this directory. Files accessed by the application will be analyzed to verify.

- **Reverse engineering**: Reverse engineering will be attempted to convert the compiled applications into human-readable source code. If possible, code review will be performed to understand the internal application functionality and search for vulnerabilities. In the case of Android, the application code may be modified and recompiled to enable access to debug information during dynamic analysis.

- **Dynamic analysis**: Dynamic analysis is performed while the application is running on the device. This includes forensic analysis of the local filesystem, network traffic between the application and server, and assessment of the app's local **inter-process communication (IPC)** surface(s).

- **Network and web traffic**: The device will be configured to route their connection to the server through a test proxy controlled by the security tester. This will enable web traffic to be intercepted, viewed, and modified. It will also reveal the communication endpoints between the application and the server so that they can be tested. Network traffic that is not traversing the Web and is happening at a lower layer in the TCP/IP protocol stack, such as TCP and UDP packets, will also be intercepted and analyzed.

- **Inter-process communication endpoint analysis**: Android mobile apps are composed of the following IPC endpoints:
  - **Intents**: These are signals used to send messages between components of the Android system
  - **Activities**: These are screens or pages within the application
  - **Content providers**: These provide access to databases
The Mobile Application Security Landscape

- **Services**: These run in the background and perform tasks regardless of whether the main application is running
- **Broadcast receivers**: These receive and possibly act on intents received from other applications or the Android system

**Exploitation**

To demonstrate real-world data breach, a properly executed exploitation can happen very quickly:

- **Attempt to exploit the vulnerability**: Acting upon the discovered vulnerabilities to gain sensitive information or perform malicious activities.
- **Privilege escalation**: Demonstration of identified vulnerability to gain privileges and attempt to become a super user.

**Reporting**

Clearly, a thorough mobile application penetration testing methodology involves a great deal of work in data collection, analysis, and exploitation:

- **Risk assessments for the findings**: Analyze business criticality of the application and the security risk posture and categorize the overall risk rating of the assessed application
- **Final report**: Detailed report about the discovered vulnerabilities, including the overall risk rating, description, the technical risk associated, technical impact, the business impact and proof of concept, and recommendations to fix the findings

**The OWASP mobile security project**

OWASP operates as a nonprofit group and does not belong to any particular technology company. It operates as a community of like-minded professionals, so it has its unique position to provide impartial information to individuals and companies. Every document, framework, tool, technique, and other details are made available to Internet users for free. OWASP always supports innovation and encourages experiments for the betterment of secure software development.

Mobile application security problems are as serious as web application security problems. Attackers have begun to focus on mobile application security issues and are actively developing tools and techniques to detect and exploit them. This community has taken the initiative for mobile application security ([https://www.owasp.org/index.php/OWASP_Mobile_Security_Project](https://www.owasp.org/index.php/OWASP_Mobile_Security_Project)) in order to help testers and developers.
The mobile security project aims at providing security insights into development in order to reduce the security impact or the likelihood of exploiting the vulnerability. The project focus is on the mobile application layer, but platform risks are considered as well.

**OWASP mobile top 10 risks**

In 2013, OWASP polled the industry for new vulnerability statistics in the field of mobile applications. The following risks were finalized in 2014 as the top 10 dangerous risks as per the result of the poll data and the mobile application threat landscape:

- **Weak Server Side Controls**: Internet usage via mobile has surpassed fixed Internet access. This is largely due to the emergence of hybrid and HTML5 mobile applications. Application servers that form the backbone of these applications must be secured on their own. The OWASP top 10 web application project defines the most prevalent vulnerabilities in this realm. Vulnerabilities such as injections, insecure direct object reference, insecure communication, and so on may lead to a complete compromise of the application server, and adversaries who have gained control over the compromised servers can push malicious content to all the application users and compromise user devices as well.
• **Insecure Data Storage**: Insecure Data Storage, as the name says, is about the protection of the data in storage. Mobile applications are used for all kinds of tasks, such as playing games, fitness monitors, online banking, stock trading and so on, and most of the data used by these applications is stored in the device itself inside SQLite files, XML data stores, log files, and so on. Or, they are pushed on to cloud storage. The types of sensitive data stored by these applications may range from location information to bank account details. **Application programming interfaces (APIs)** that handle the storage of this data must securely implement encryption/hashing techniques so that an adversary with direct access to these data stores via theft or malware will not be able to decipher the sensitive information stored in them.

• **Insufficient Transport Layer Protection**: All the hybrid and HTML 5 apps work on the client-server architecture; emphasis for data in motion is a must as the data will have to traverse through various channels and will be susceptible to eavesdropping and tampering by adversaries. Controls such as SSL/TLS, which enforce confidentiality and integrity of the data, must be verified for correct implementations on the communication channel from the mobile application and its server.

• **Unintended Data Leakage**: Certain functionalities of mobile applications may place sensitive data of the users in locations where it can be accessed by other applications or even by malware. These functionalities may be there in order to enhance usability or user experience but may have adverse effects in the long run. Actions such as OS data caching, key press logging, copy/paste buffer caching, and implementations of web beacons or analytics cookies for advertisement delivery can be misused by adversaries to gain information about victims.

• **Poor Authorization and Authentication**: As mobile devices are the most personal devices, developers utilize this to store important data such as credentials locally in the device itself and come up with specific mechanisms to authenticate and authorize users locally for the services that the user is requesting via the application. If these mechanisms are poorly developed, adversaries may circumvent these controls and unauthorized actions can be performed. As the code is available to adversaries, they can perform binary attacks and recompile the code to access authorized content directly.

• **Broken Cryptography**: This relates to weak controls that are used to protect the data. The usage of weak cryptographic algorithms, such as RC2, MD5, and so on, that can be cracked by adversaries will lead to encryption failure. Improper encryption key management when the key is stored in locations accessible to other applications or the use of a predictable key generation technique will also break the implemented cryptography techniques.
• **Client Side Injection**: Injection vulnerabilities are the most common web vulnerabilities according to OWASP web top 10. These are due to malformed inputs that cause unintended actions, such as altering database queries, command execution, and so on. In the case of mobile applications, malformed inputs can be a serious threat at the local application level and on the server side as well (such as the risk of **Weak Server Side Controls**). Injections at the local application level that mainly target data stores may result in conditions such as access of paid content locked for trial users or file inclusions, which may lead to abusing functionalities such as SMS, and so on.

• **Security Decisions via Untrusted Inputs**: The implementation of certain functionalities such as use of hidden variables to check the authorization status can be bypassed by tampering them during transit via web service calls or inter-process communication calls. This may lead to privilege escalations and unintended behavior of the mobile application.

• **Improper Session Handling**: The application server sends back the session token on successful authentication with the mobile application. These session tokens are used by the mobile applications to request for services. If these session tokens remain active for a longer duration and adversaries obtain them via malware or theft, the user account can be hijacked.

• **Lack of Binary Protections**: Mobile application source code is available to everyone. An attacker can reverse engineer the application and insert malicious code components and recompile them. If these tampered applications are installed by a user, they would be susceptible to data theft, become victims of unintended actions, and so on. Most of the applications do not ship with mechanisms such as checksum controls, which help in deducing whether the application is tampered or not.

In 2015, there was another poll under the OWASP Mobile security group named the Umbrella Project. This leads us to have M10 to M2; the trends lock binary protection to take over weak server-side controls; however, we will have to wait until the 2015 final list. More details can be found at https://www.owasp.org/images/9/96/OWASP_Mobile_Top_Ten_2015_-_Final_Synthesis.pdf.
Vulnerable applications to practice

The open source community has been proactively designing plenty of mobile applications that can be utilized for practical tests. These are specifically designed to understand the OWASP top 10 risks. Some of these applications are as follows:

- **iMAS**: This is a collaborative research project initiated by the MITRE Corporation (http://www.mitre.org/). It is for application developers and security researchers who would like to learn more about attack and defense techniques in iOS. More information about iMAS can be found at https://github.com/project-imas/about.

- **GoatDroid**: A simple functional mobile banking application for training with location tracking developed by Jack and Ken for Android application security is a great starting point for beginners. More information about GoatDroid can be found at https://github.com/jackMannino/OWASP-GoatDroid-Project.

- **iGoat**: OWASP’s iGOAT project is similar to the WebGoat web application framework. It’s designed to improve the iOS assessment techniques for developers. More information on iGoat can be found at https://code.google.com/p/owasp-igoat/.

- **Damn Vulnerable iOS Application (DVIA)**: This is an iOS application that provides a platform for developers, testers, and security researchers to test their penetration testing skills. This application covers all of OWASP’s top 10 mobile risks and also contains several challenges that one can solve and come up with custom solutions for. More information on this can be found at http://damnvulnerableiosapp.com/.

- **MobiSec**: This is a live environment for the penetration testing of mobile environments. This framework provides devices, applications, and supporting infrastructure. It provides a great exercise for testers to view vulnerabilities from different points of view. More information on MobiSec can be found at http://sourceforge.net/p/mobisec/wiki/Home/.

Summary

In this chapter, we saw the evolution of mobile applications over the years and the need for mobile application security—in particular, the role of penetration testing for mobile applications. Understanding the methodology, common vulnerabilities around iOS and Android are a crucial part of mobile application penetration testing. We covered the current mobile application security landscape and existing methodologies, such as OWASP, along with several concepts and vulnerable applications for testing. We will discuss the different Android and iOS architectures in the next chapter.
Where to buy this book

You can buy Mobile Application Penetration Testing 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.