Mobile Applications -Vulnerability Assessment Through the Static and Dynamic Analysis

name: vishal sanjay patil
intern id : CA/JL3/13813
pvishu570@gmail.com

Abstract

In the recent day's mobile applications usage is increasing by banking and financial institutes, health and hospital management systems such as mobile banking apps, e-commerce apps, news feeds, inpatient and outpatient information, social networking apps and game apps etc. All these mobile applications require support for security properties like authentication, authorization, data confidentiality, and sensitive information leakage etc. The mobile applications have seen rapidly growth in the last couple of years. These applications has provided suitable to banks, credit card data, personally identified information, travel applications etc., the enterprise mobile applications extend corporate networks beyond the perimeter devices and thus potentially expose these organizations to the new types of security threats. Security risks associated with these applications can often be identified and mitigated by subjecting them to security testing. Compared to desktop or web applications, mobile applications are hard to test for security. At the same time, these applications are not necessarily more secure then desktop or web applications.

Keywords:Application Security, Mobile Security, Penetration testing, Information Security, Security Testing, VAPT.

1. Introduction

In this paper, we discuss the importance of mobile applicationsvulnerability assessment and penetration testing techniques to identify different mobile application threats and evaluate our contribution on the different flavors of mobile development platforms such as iOS for iPAD and iPhone apps, Android apps, Windows apps and BlackBerry applications.Currently most of the banking, financial, insurance,health systems and games for their mobile applications rely on underlying device operating system for their security.

The mobile platform has become everywhere for conducting business and engaging with their consumers. The always-connected, portable devices give direct access to systems and data, anywhere a cellular or wireless connection can be established.

But, with mobile application flexibility comes with complexity and insecurity. While the mobile OS vendors or providers have attempted to build a more secure mobile operating systems compared to their desktop families, mobile applications can still be built using insecure coding practices, on top of developmentplatforms, and with insecure features and functions. As security is often an afterthought in application development, the tradeoff between delivering a secure application late and delivering a functional application on time often gets made at the expense of security. However, this doesn't have to be the case; selecting the right platform can enable secure applications to be built across multiple platforms, delivering them with full functionality, on time. Also we will discuss mobile Top-10 vulnerabilities in the next sections.

Figure 1: Smart phones sales and their usages

2. Mobile Application

A mobile application also called as mobile app or smart phone app is a software application designed to run on smartphones, tablet computers and other mobile devices.

They are usually available through application distribution platforms, which are typically operated by the owner of the mobile operating system, such as the Apple App Store, Google Play (Android platform), Windows Phone Store and BlackBerry App World. Some apps are free, while others must be bought. Usually, they are downloaded from the platform to a target device, such as an iPhone, BlackBerry, Android phone or Windows Phone.

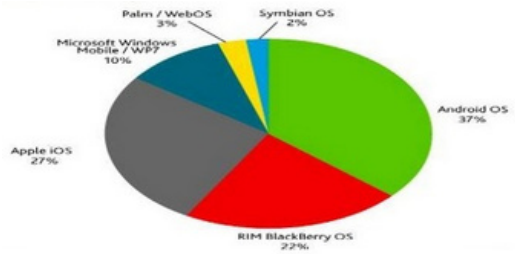

Figure 2: Popular mobile platforms in the recent market

The term app called as application has become very popular. Mobile apps were originally offered for general productivity and information retrieval, including email, calendar, contacts, and stock market watch and weather forecast information. However, public demand and the availability of developer tools drove rapid expansion into other categories, such as mobile games, automation, GPS and location-based services, mobile banking, order-tracking, and ticket purchases. The eruption in number and variety of applications made discovery a challenge, which in turn led to the creation of a wide range of review, recommendation, and curation sources, including blogs, magazines, and dedicated online application discovery services.

Figure 3: An iOS Architecture

3. Mobile Application Vulnerabilities

The popularity of mobile applications (for smart phones) has continued to rise, as their usage has become increasingly prevalent across mobile phone users [1]. We can see the risks at every layer as follows:

- Mobile Network Level: interception of data over the air.
  - Mobile Wi-Fi has problems same as problems as laptops.
  - GSM has shown some cracks.
- Device Hardware Level: called baseband layer attacks
  - Memory corruption defects (buffer overflow) in firmware used to root device.
- Operating System Level: defects in OS kernel code or vendor supplied system code.
  - IPhone or android jailbroken devices are usually exploitingthese defects.
- Application Level: Mobile apps with vulnerabilities and malicious code have access to user's sensitive data and device sensors.
  - Device isn't rooted but all your email and pictures are stolen, your location (geo tagging attack) is tracked, and phone bill is much higher than usual.

In this paper we will discuss only the common application security risks (Top-10) found in different mobile applications irrespective of mobile platform providers. We see these security risks in mobile applications every day. When we see them they often show up as vulnerabilities in the applications we are assessing. However, this is a good starting point for a security or development teams looking to understand the most common mobile application security issues[1] [2] and fix them appropriately. This paper focuses on these risks and also will discuss mobile threat model.

1. Insecure or unnecessary client-side data storage
2. Lack of data protection in transit
3. Personal Data Leakage
4. Failure to protect resources with strong privilege authentication
5. Failure to implement least authorization policy
6. Client-side injection
7. Client-side DOS
8. Malicious third-party code
9. Client-side buffer overflow
10. Failure to apply server-side controls

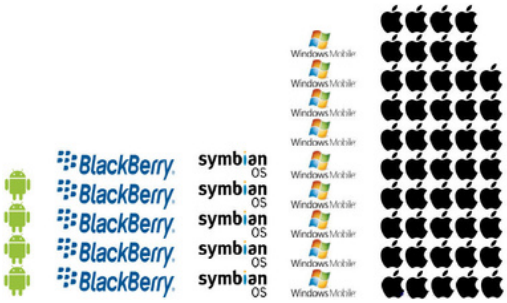

Figure 4: Vulnerability count by platform

3.1. Insecure or unnecessary client-side data storage

This is security risk addresses the understandable concern of sensitive or PII(personally identifiable information) data being stored on physical mobile devices such as iPhone, android, windows or black berry. All application developers must prudently consider storing a piece of data on a mobile device's storage area is entirely critical to the application's functionality. If the data is required to an application, and it is sensitive, it must be protected in such a manner. Data protection means encryption. For iOS or android developers, this is not adifficult problem. Mobile OS provides a very easy to use and secure data protection mechanism. The mechanism is suitably referenced as data protection. Apple iOS provides two different methods to achieve this and it is very straight forward to use.

a. NSDataWritingFileProtectionComplete:This is sub- class of NSData and it is used to set the content protection attribute of the file when writing it out. In this case, the file is stored in an encrypted layout and may be read from or written to only while the physical device is unlocked and at all other times, attempts to read and write the file result in failure.

b. NSFileManager: The NSFileManager class enables to perform generic file system operations and protects an app from the underlying file system. To protect the file at rest you should not access the device.To get the best results when the user has a locked their device and app can only be ready for the user to protect them.

3.2. Lack of data protection in transit

This risk belongs to the data while transmitting. After the data has been secured on the physical device the next major concern is protecting the communications between the mobile application and the application server. The most common protocol is HTTP (S). This can achieve application developers by using the NSURL or NSURLConnection in iOS. By default,NSURL or NSURLConnection will fail with an error in the event of an SSL issue. Integrated development environments do not have a valid SSL certificate and which creates a problem. NSURL and NSURLConnection behavior is changed to accept invalid certificates to development without hassle.Implementing these methods also gives us the opportunity to warn the user in the event of an invalid SSL certificate.

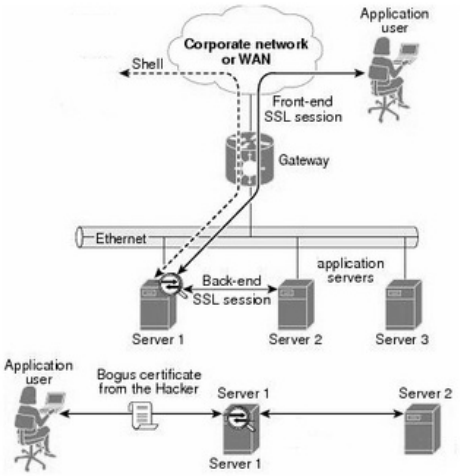

Figure 5: SSL certificate security issue

3.3. Personal Data Leakage

This is another common threat in the mobile applications. From risk perspective the priority of the issue is less.Application developers must take care to secure their user's private information such as usernames, passwords, credit card numbers, emails id's, DOB and address,.Applications providers must protect their user's personal data. We can use the similar data protection mechanism [3] described in the previously discussed section in insecure or unnecessary client-side data storage to protect personal user data. Beyond that this it is an application design decision about how the application will handle the app user's personal data. Personal data privacy policy has become a hot topic and users are becoming much more aware that their private data may be at risk in mobile applications. The following figure [6] shows, personal data leakage on an average.

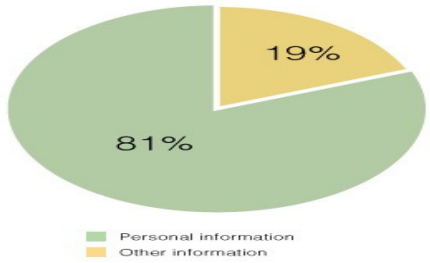

Figure 6: Personal data leakage

3.4. Failure to protect resources with strong authentication

This issue is belongs to a client and server issue. Very little authentication is typically performed on a mobile device in the figure [7].

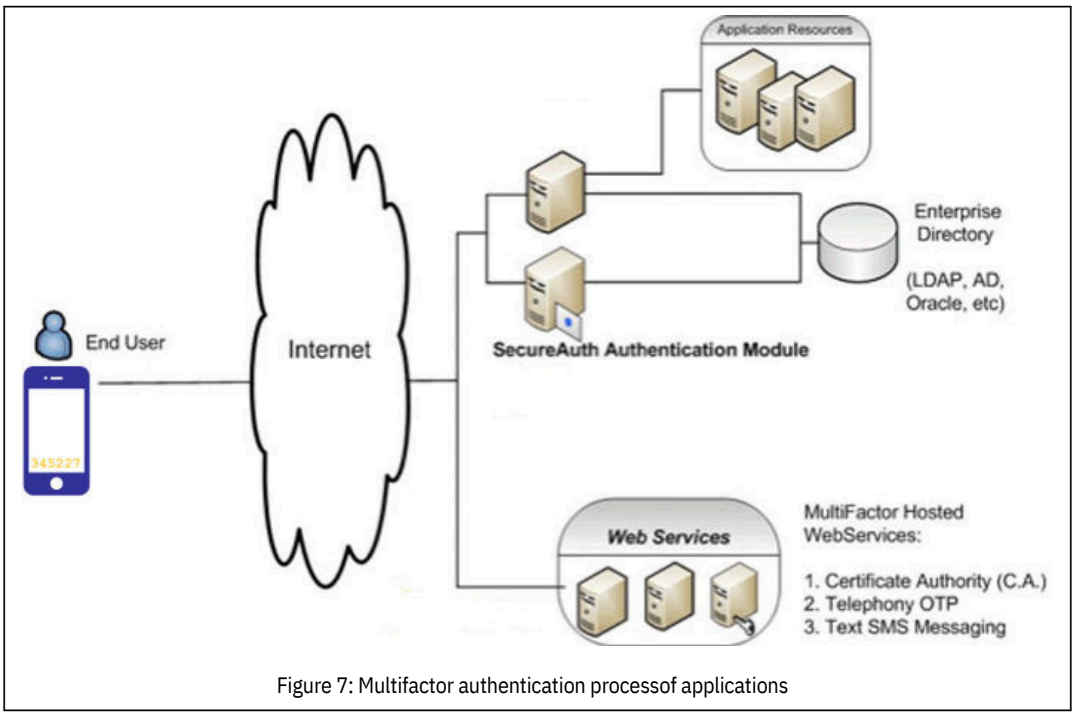

Figure 7: Multifactor authentication processof applications

The majority of the authentication mechanisms that uses mobile devices are server authenticating. Applications on mobile devices hardly authenticate other services directly on internet. If this does occur an application is usually being asked to share a resource, such as a photograph or some other piece of data managed by the application, with a server.

The main concern is that mobile applications properly authenticate [4] to servers and that they implement strong authentication that uniquely identifies each mobile application user to the server. One of the main concerns for application developers is to never embed client side secrets such as tokens (Oauth tokens like consumer token or access token) in their application and then use those as an integral part of an authentication method. 3.5. Failure to implement least privilege authorization policy

Applications should request permissions whenever they requiredthe permissions absolutely they need. Does your mobile application really need access to the user's GPS (global positioning system) or geo tagging location? Is this is protecting user's privacy data. Be practical with what resources user attempt to access.The other part of this issue is application server issue.

The key issue is how to handle that applications, particularly thick client based applications, may contain a great application functionality that may not available to lower privilege users. The application server responsible for checking and validating that a user can perform a requested action. Even if the functionality is accessed via the application, the app server must not allow lower privileged users to access and execute higher privileged server side business functionality. This is called vertical privilege escalation and it is a constant risk to server side applications.

Horizontal privilege escalation [8] flaw allows users of mobile applications to easily bypass authorization controls and can access the data of other registered usersat the same privilege level. In, this situation, proper care must be taken and allow only a mobile application to access server side data that belongs to the currently authenticated and authorized user.

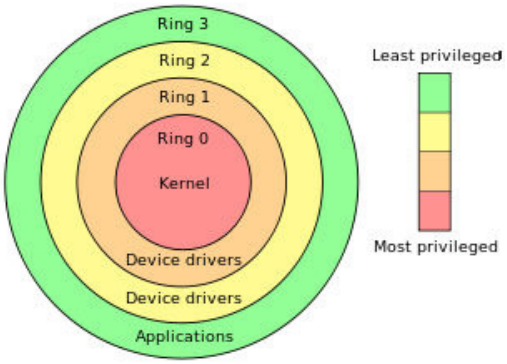

Figure 8: Privilege escalation from least to most

3.6. Client-side attack

Client side injection attack is serious security issue and it is an interesting problem that can lead to a variety of security issues depending on the application and how it operates. Many mobile applications utilize SQLite database to store data at the device memory, which means that some of the applications may be vulnerable to SQL injection. Frequently the significances of SQL injection against a client side application are minimal impact. Also there are another kind of attack called cross site scripting (XSS) attack.

Figure 9: XSS attack on mobile application.

This attack illustrates the possibilities of rich user environments and the concerns they can have when an application implicitly trusts users input. The vulnerability was a simple cross site scripting attack that allowed remote code execution. Similar issues can surface in it and applicable to all mobile applications if the application uses UIWebView or web view kit or other rich environments and it does not carefully check users input.

Any specific advice given here would be specific to be useful in a general way to application developers. Make sure application data confirms to the expected length, range, data type and format. Data length is obvious, range is for data that has expected numerical ranges (it might positive integers only), data type is for any sort of integer or other data structure being read and format is for the actual data formatting.

Data type is an interesting part of the application input. Any kind of objects are inherently insecure and should never be trusted from a remote source. If a user can directly input objects into the application system and manipulate them a variety of difficult to catch security issue can result.

Another interesting issue is with string format in the iOS based applications. Objective-C development environment it hardly feels like you are writing application code that gets compiled to the native programming instructions. Drop down to a library, like SQLite and that illusion is quickly shattered. Format strings using the%@ formatter are vulnerable to a variety of interesting attacks: () such as buffer overflow attacks.

3.7.Client-side DOS

This security issue is honestlyunderstandable. Make sure you are being a defensive application programmer. This can result from development errors and bad application programming logic. New and modern mobile applications do a lot of data parsing of formats such as XML and JSON objects. Defects in these parsers or in the way they are used can result in unexpected DoS (denial of service) attack.

3.8.Malicious third-party code

This security issue applies only iOS based applications. Assume that the device is jailbroken (unlocked). A malicious user can do anything on jailbroken devices. Also, even on appropriate, non jailbroken devices there are remarkable problems that can arise related to the URL/Protocol handling mechanism. iOSapps is not very great on inter process communication. One of the few mechanisms available to iOS application developers to achieve this is the custom URL implementation scheme. Use these with extreme caution. The 3rd party of the code might leads to the application dead state or reads data or sensitive information and send it malicious user. In the mobile application world especially Android platform [5], there are huge number fraud applications because of its open source mobile platform and there is no control to the 3rd code as well as developers compare to the iOS application developers.

Figure 10: Third party application with malicious code installation

3.9. Client-side buffer overflow

This is common application vulnerable and it'sstill quite possible in the native languages such as C, C++. Since Objective-C is a strict superset of C programming and there is no limit to the depth in which an iOS can get itself in trouble with old C programming issues. This attack can easily exploitable.Mobile application developers should take care most careful and attention to string formatting such as NSLog, etc. It is necessary to stick to the NS* class hierarchy when at all possible. Abstract away your C code to the smallest amount possible and be extremely cautious with any C string and memory operations especially iOS applicationsdevelopment.

3.10. Failure to apply server-side controls

This is one the most of the common and serious security issue in both web and mobile applications. For example, whenever accessing the application, the proper controls should be taken care as we discussed in the earlier sections. It includes, authorization and authentication check, read or write permissions, also should check cross talk security issues and also consider the above security issues.

4. Mobile Threat Model

Threat modeling [9] is a systematic process developed by Microsoft Corporationthat begins with a clear understanding of the system or an application. It is necessary to define the following areas to understand possible threats to the application as shown in the figure 11.

a) Mobile Application Architecture: This area describes how the application architecture is designed from device specific features and functionalities used by the application, wireless and data transmission protocols, data transmission mediums, interaction with hardware components and other applications of the organization.

b) Mobile Data: What kind of data does the mobile application store and process? What is the business purpose of this data and what are the data workflows etc.

c) Threat Agent Identification: What are the threats (or evens) to the mobile application and who are the threat agents. This area also outlines the process for defining what threats apply to the mobile application such as entry and exit points of the application.

d) Methods of Attack: What are the most common attacks utilized by different threat agents. This area defines these attacks, so that application or server controls can be developed to mitigate attacks.

e) Controls: What are the different application and server controls to prevent attacks? This is the last area to be defined only after previous areas have been completed by the application development team.

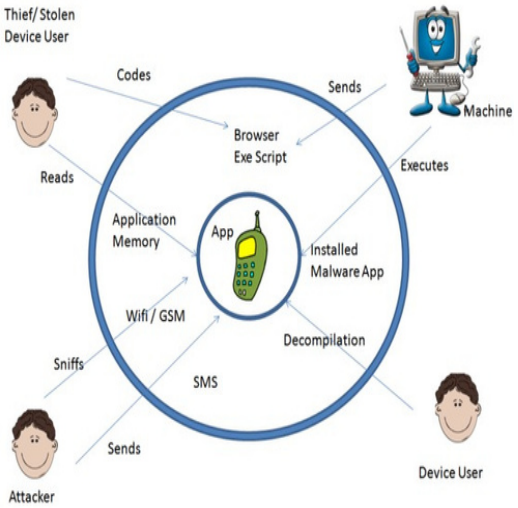

Figure 11: Mobile threat agent identifier and its types

5. Conclusion

Mobile application developers and the organizations intending developing and deploy mobile applications in secure manner and must plan their security testing called

penetration testing and they follow strategy across manual and automation tests approaches for efficient and error-free application delivery. In addition to actual devicebased testing, emulators should be included as an integral part of the security testing program. Enterprise application require special pen tensting techniques because they deal with lot users sensitive data. Outsourcing to vendors who are operating an independent testing practice may be a viable option to manage the expertise, scalability, and quality assurance requirements for mobile application delivery in secure manner.