VETTING THE SECURITY OF MOBILE APPLICATIONS (REVIEW PAPER)

[1]Ahmed Khan, [2]Aaliya Sarfaraz
[1] Department of Information Security, NUST, Pakistan.
[1,2] Department of Computer Science, COMSATS, Pakistan.
[1]{ahmed.khan, [2]aaliya.sarfaraz}@comsats.edu.pk

ABSTRACT: *As the invasive use of digital communication technologies in all the areas, making everything digital also bring with it the threats of intruding, information modification and fabrication by some counterfeit source and badly affect the confidentiality and integrity of information. The security issues and confidentiality of the sensitive information has become the prime and necessary concern, as various events of global terrorism energized the requirement for better techniques for securing the machines and the data they store, transform and transmit. Malware refers to the malicious software that can execute and harm the computer on the any network or another. There are many dangerous malware have created for the destruction of mobiles for some personal purpose or for some beneficial gain. This paper illustrates the weaknesses of the mobile application and the threats from the malicious activities. Therefore, we proposed a solution regarding the issues findings.*

INTRODUCTION

a) Mobile application security motivation?

Security plays an important role in case of software application development. Although, there are many applications that require security layer, but mobile applications (android, iPhones) need special concentration because these devices provides offline and online facilities to user which might be vulnerable to application information. In this perspective, mobile applications are also building that always sync the mobile data to online using web services. To make application safe, application needs to be secure [1].

Due to its great feature of providing access to its subscribers everywhere all over the world, the mobile communication has become very attractive among its subscribers, service providers and operators. In addition to its several advantages, the mobile communication also has many security issues [2]. In GSM, GPRS the architecture has three main nodes; mobile station, visitor location register and home location register. These three nodes perform encryption and decryption of data and provide authentication of the subscriber in GSM, GRPS [3]. Symmetric encryption has been in use, which includes any form ranging from simple substitution to more complex ciphers [4]. However, it has some limitations like random num generation and encryption function [5].

b) Why We Need Security Breaches Detection?

Protecting ad-hoc networks needs to be a multi-pronged strategy. Intrusion prevention in the form of strong identification and authentication mechanisms alone are not sufficient [6]. A malicious intruder can still launch attacks from both outside and inside the network environment that can weaken and compromise the network integrity resulting in serious consequences.

Recognition of unauthorized attempts is done manually or automatic systems that create logs of information regarding the behavior of the network. Humans are much interested in automation of such system because manually it takes too much time and effort. In IDS, firstly system must be observable and present different behaviors graphically if distinction found on the network.

The reason why we need security breaches detection mechanisms is that encryption and certification reduce the probability of information leakage. These schemes make the system form against vulnerable attacks.

In this paper [1], the main focus is on the information security related to mobile apps. Different scenarios are observed to prevent the mobile apps from modification and fabrication performed by attacker intrusion.

App testing in organizations is done by app sending by admin to analyzers. Security is a very important feature to keep app confidential and authentic. Information contained by app is related to author copyrights and codes to handle big task, depending upon the app.

Issues are also raised if administrator sends app electronically to analyzer over an unencrypted network. This may allow an unauthorized access to an attacker.

After app receiving, if analyzer will store it in unsecured system and database, then unauthorized users can access the app. This access has possibility that unauthorized user get copy rights, codes and another app is replaced by an attacker.

After assuring that app is secured from the unauthorized access, modification and fabrication, analyzer will test app susceptibility that disobeys app security requirements. In this section, requirements and the testing approaches are discussed which is used to detect breaches that contravene the requirements.

In this paper our focus is on two parameters. First, the maturity required to make mobile apps more secure by experienced team during the software development life cycle. Maximum security checks in every chunk of code are required. A special team required to deal with apps critically. Its advantage is that apps will get more and more attention of the team. Expertise of the team also matters in this regard. Maturity can also be the part of the apps development if team uses some strategic mechanism to make the apps instead to code and fix method. There must be a condition on the permission to use it. Permission regarding the security point of view is also required on every app.

The team has to analyze the nature of the apps and code in it. This checking is important that either apps code uses data from the other apps or redirecting the secret information to some other address. Apps needs to be self defended. Apps have to back trace the malware source.

A second point of attention is that awareness of the user regarding usage of the apps are required [7]. In fact, people find easy to them and trust on free available apps of app

stores instead of paid apps.

Furthermore, section 2 is about the literature being discussed in the paper. Section 3 is related to the problems found and solutions of those problems. The conclusion is given in section 4.

LITERATURE REVIEW

1. *General Requirement*

Following factors are discussed to implant the security and secure the systems from unauthorized access, modification and fabrication. Security could only be the part of the application if it embed at every phase of software development. Goal of setting the software security requirements are to develop such apps that are not vulnerable to attacks. Requirement gathering is a part of requirement engineering. This stage of app development phase could be much mature if security involves in it. To set track on the gaps and their solutions, it is important to have a record in documentation. This documentation records all the gaps, their vulnerability and their solutions.

a) Enabling certified functionality: This functionality means that all graphical user interfaces and GUI components like buttons and text fields work properly. Gracefully, error catching conditions must be handled via exception handling as functionality is not available [8].

b) Avoid illegal functionality: Illegal functioning by malware must not be supported like data leakage performed.

c) Limiting permissions: Applications must allow other apps or functionality in bounds to interact with it.

d) Guard sensitive data: Applications should provide the confidentiality, integrity and authentication to the software and the systems that store and transmit the data to avoid the unauthorized access. This requirement guards the personal information and code of te author.

e) Lessen code reliance: Apps must ensure that the libraries used to provide the features, not be fake and harmful to code [9].

f) Apps update testing: A rapid application development system releases new versions of the apps very quickly. Application updates must be tested to prevent the other software and system from harmful resources using finding the weaknesses.

1.1 Enabling Authorized Functionality

An imperative piece of affirming approved usefulness is trying to display the portable user interface (UI), which can shift significantly for diverse screen sizes and resolutions. The rendering of pictures and buttons may not be right because of these distinctions if applicable, portrait and landscape mode also available for UI display. Telephony usefulness incorporate a wide assortment of system calls that an application can utilize if given the best possible authorizations, including making telephone calls, transmitting SMS and recovering unique telephone identifier data [10]. Numerous applications utilize the telephone number (hard coded) as a unique identifier. However, this is a terrible coding practice which should be noted by a company. This is a block used to carry out the exceptions like this [11].

If(email.getText.equals(“ali@yahoo.com”) && pass.get Text.equals(“123”))

// login page redirecting via hard coded information

1.2 Preventing Unauthorized Functionality

Most mobile applications are malevolent and functionally perform secure behavior. Illegal functioning by malware must not be supported like data leakage performed or tracking locations. Sensitive information can also be hacked by injecting the bogus website or add-ons (link) into user’s browsers as shown in fig 1 below. These attacks are not detected by the mobile antivirus, called phishing attack [12]. Publically available antivirus can recognize known and unknown malware. These tools are also used as mobile application to provide protection to mobile apps. If these apps have not identified the virus or malware in the apps, then no one could know that there is some malware in the mobile. This can be prevented by using the authorize antivirus to provide CIA to the users [13].

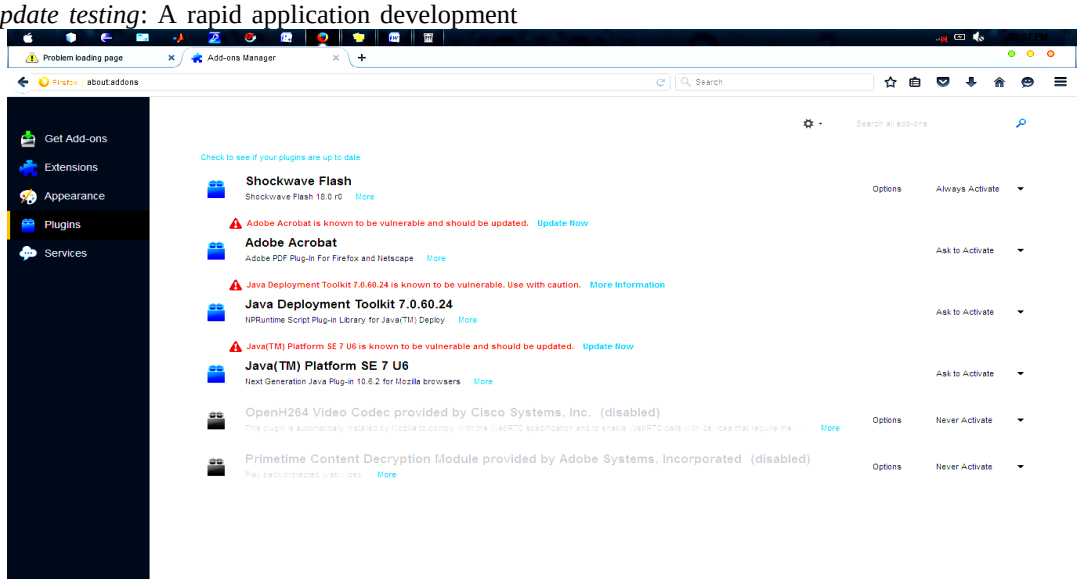

Fig 1. Malicious add-ons are in the browsers

1.3 Limiting Permissions

Some of the mobile applications have agreement are not reliable in order to exhibit secure behavior of the application. Similarly, most of the applications have permissions are not in use. It is important to identify app's permissions that either it is permission or not because it can be malware which is represented by the permission. Unnecessary permissions can manifest like:

a) File (I/O) and removable storage: Security is in risk when I/O of a file using a removable device. Malware can be activated after connecting the device (if it is a requirement that activation can only be performed after event performs the device connection). Usually, an autorun virus is activated and embeds itself in every file and directory of the hard disk [14].

b) Privileged commands: Mobile applications can have the ability to perform the low level commands that can affect the hardware and utility software.

c) APIs: Application programming interface provide the abstraction level that helps the developer to build program easily and rapidly. Android provide the ADT "http://developer.android.com/guide/developing/eclipse-adt.html" to facilitate the developer; no other toolkit should be used with the reason that they can be malicious and unauthorized.

1.4 Protecting Sensitive Data

Many mobile applications assemble, pile up, and transmit data, like credit card, personal and login data. Cryptography helps to keep up CIA with users to protect the user information. However, firm cryptography codes are few that resist attacks and some are biased in nature.

Some reasons are there that compromises security threats over cryptography models like improper cryptography key managing and less complex algorithms with hard coded information.

Another reason of data leakage is the stack overflow; it is performed when a developer doesn't care the most and common exception "IndexOutofBoundException". This exception raised when memory in arrays and vector is not sufficient and information stored in other memory location. This can be vulnerable to the attacks from the intruders and information leaks. This is block used to carry out the exceptions like this [2,3].

```
try{
// statements
}catch (IndexOutofBoundException e){
        e.getMessage();
}
```

1.5 Securing App Code Dependencies

Dependency on the libraries can lead the system to unsecure body which building the apps using built-in functions. Apps must ensure that the libraries used to provide the features, not be fake and harmful to code.

a) Dynamic Behavior: During mobile applications executing, they perform different kind of behavior. The important thing to know is that where data used by the application begin and where to utilize. Critically, it is difficult to analyze that downloaded stuff from an exterior resource is hazardous.

b) Inter-Application Communications: Mobile applications that converse with other apps provide capability and efficiency upgrading, but there is existence of security hazard. Like, Android communications are allowed "intents" which contains the view information and interact with the android operating systems.

1.6 Testing App Updates

In testing the new version of mobile applications, unintended weaknesses, finding is the main goal in the phase of quality assurance to provide long time guarantee. Mobile applications should examine before next release to authenticate that the application regarding security risk like developers unconsciously implanted weaknesses that makes system vulnerable to attacks.

App stores are available to download mobile applications of the user choice. Ideally, every application and its updates ought to be scrutinize prior to download [15].

2 Testing Approaches

In detection of fulfillment or destruction of requirement, tester tests the mobile apps for the occurrence of software weakness via following points.

2.1 Correctness Testing

It is the process with the intent of finding errors in executing the program to improve quality and with goal of fill out all the security holes. Recommend this testing whenever it is required because there is a firm pairing between quality and security, functionality and trustworthiness of software.

2.2 Binary Code Compatibility

It refers to byte-code. Like, android and Java Virtual Machine (JVM) compile code to generate byte-code which directly executed on the mobiles.

2.3 Automated Tools

Automated tools are used by testers for applications inspection process will to find out the software vulnerabilities and report be made at the end of phase to analyze risk assessment. Following tools are the kind to automated tools [6,9].

a) Simulators: It is the virtual visuals of the actual device like (android simulators) used to view, the view for testing and before launching in the original device as shown in fig 2.

b) Remote Device Access: Distant analyzer tools are used to investigate and understand mobile applications.

c) Automated Testing: There are several softwares that are used to automate the testing process of mobile application to provide CIA to the user. Following are the tools:

- **User Interface-Driven Testing:** Pixel verification used to verify this type UI-specific testing.
- **Data-Driven Testing:** Data-driven authentication utilizes text to recognize the mobile application page. Its tests are used to take notice on the leakage of information of session changing after login and logout page [15].
- **Fuzzing:** Automated creation of analysis inputs, randomly or pattern information is called Fuzzing. Tools used for wrong information by hacker are called "fuzzers."

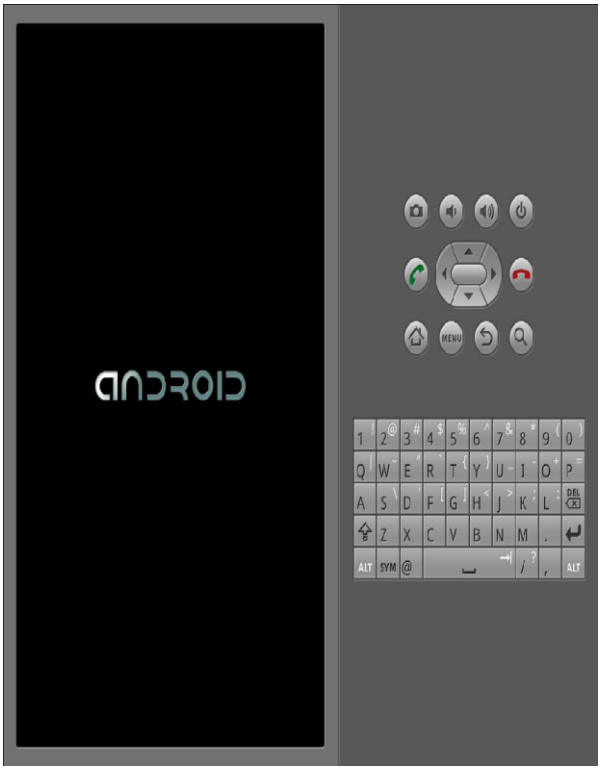

Fig 2. Android Simulator

- **Network-Level Testing:** Incursion test tools used by hackers to access the system and network via simulations are called fuzzers. So, network traffic can be viewed more closely to determine the nature of the attack.

d) Test Automation: Following are the tools used to mechanize the reiteration of tests [1].

- **Static Tools:** Software behavior can be analyzed by providing specific requirements using static tools to find software susceptibility. These tools also used to evaluate the correctness against weaknesses.
- **Metrics Tools:** These tools are used to evaluate the complexity of software, that's how the system is fast. These tools are usually used in RAD (Rapid Application Development) to analyze the weaknesses. Quantifying different measurements are to be taken to observe the system behavior.

These tools clarify the constraints like time and money to save the company assets.

PROBLEM STATEMENTS AND SOLUTIONS

Problem 1: In this paper, it is observed that mobile application has problems more then the desktop and web applications. Because it is always sync the device and web information of every application. Intruders interact with the mobile application on network level with harmful aims.

Solution 1: To provide CIA to users, mobile application developer's needs to handle the exception very well and tester should test the apps carefully with every aspect. Applications build by the experienced developers have to be sectioned. Especially open source apps needs to be tested well.

Problem 2: There is no tracing mechanism of the intruders after it is known that apps are being used by some unauthentic mean. People having personal information on system, does not care on the security meters.

Solution 2: Awareness is required to the people and solutions have to be made so that unauthentic user is required to redirect in the section (especially design to catch the source and send back the malware). Policy needs to be design for this security implanting to provide hard security checks.

CONCLUSION

In this paper the mobile application is analyzed with different level of interest. Mobile applications are very sensitive ones because apps are on the target both by the development team and the unauthorized people. Furthermore, interest of the users regarding the involvement of unauthorized people in their systems is zero. So, we have proposed the solution in contrast of finding the above issues. A policy required to implant a mechanism that provides complete CIA to users. Furthermore, quick response is also required that back trace the intruder and return back the warm to their system. In order to save the time, automated systems are required to build that resists the malware and guard the personal information using mobile apps.

REFERENCE

[1]. Voas, J. (2015). Vetting Mobile App Vendors. *Computer*, (6), 69-71.

[2]. Shah M, Khan A. (2016). "Implementing User Authentication Service for Cloud Network", *Science International*, 28(6), Dec (2016), pg: 5301-5306. (ISSN 1013-5316) (Impact Factor: 0.75) (ISI Indexed Journal)

[3]. Khan. A, Sohaib. M, Amjad. F. M. (2016). "High-Capacity Multi-layer Framework for Highly Robust Textual Steganography", *Science International*, 28(5), Oct (2016), pg: 4451-4457. (ISSN 1013-5316) (Impact Factor: 0.75) (ISI Indexed Journal)

[4]. Khan. A., (2015) "Comparative Analysis of Watermarking Techniques", *Science International*, 27(6), Dec (2015), pg: 6091-6096. (ISSN 1013-5316) (Impact Factor: 0.75) (ISI Indexed Journal)

[5]. Khan. A, Tariq U, Shabbir J, Hassan S. (2016). "Cloud Security Analysis for Health Care Systems". *International Journal of Computer and Communication System Engineering*, 3(1), 1-8. (Impact Factor: 0.374)

[6]. Khan. A. (2016). "Joint Ownership Verification for Digital Text". *Advances in Computer Science and its Applications* (ACSA), 3(4), 525-531. (Global Impact Factor: 0.678)

[7]. Khan. A. (2015). "Robust Textual Steganography". *Journal of Science* (JOS), 4(4), 426-434. (Impact Factor: 1.34)

[8]. Azeem S, Khan. A, Qamar E, Tariq U, Shabbir J. (2016). "A Survey: Different Loss-less Compression Techniques." (IJTNR) *International Journal Of Technology & Research*., 4(1), 1-4.

[9]. Khadim U, Khan. A, Ahmad B, Khan A. (2015). "Information Hiding in Text to Improve Performance for Word Document." (IJTNR) *International Journal of Technology and Research*, 3(3), 50-55.

[10]. Khan. A., Naqvi, N., & khan, A. (2015). "Survey to Improve SQA in Developing Countries". (IJTNR)

International Journal Of Technology & Research., 3(1), 1-6.

[11]. Khan. A. (2015). "Novel Fair Scheduling of Processing Unfairness". (IJTNR) *International Journal Of Technology & Research*., 3(2), 39-41.

[12]. Tariq U, Shabbir J, Hassan S, Khan. A. (2015). "Comparative Analysis of Java and C# Compilers Code Optimization". (IJTNR) *International Journal Of Technology & Research*., 3(4), 1-4.

[13]. Khan. A. (2014a). "Comparative Analysis of Java And C++ History, Similarities & Differences, Syntax And Design Issues." (IJTNR) *International Journal Of Technology & Research*., 2(4), 131-140.

[14]. Khan. A. (2014b). "Quality Software & Security Testing And Its Impact On Software Product." (IJTNR) *International Journal Of Technology & Research*., 2(4), 114-117.

[15]. Khan. A. (2014c). "Streaming File Sharing Media Communication and Intranet Chatting System." (IJTNR) *International Journal of Technology & Research*., 2(3), 81-87.