JES
Journal of Engineering Sciences

Vol 13, Issue 06, June/2022

ISSN NO:0377-9254

Security Threats To Mobile Multimedia Applications

1J SRIVIDYA, 2PHANI CH, 3ANIKETH D, 4ROHITNIVAS B
1Assistant Professor, 234B.Tech Student
DEPT OF CSE
CMR TECHNICAL CAMPUS, Hyderabad

ABSTRACT: Today's mobile smartphones are very powerful, and many smartphone applications use wireless multimedia communications. Mobile phone security has become an important aspect of security issues in wireless multimedia communications. As the most popular mobile operating system, Android security has been extensively studied by researchers. However, few works have studied mobile phone multimedia security. In this article, we focus on security issues related to mobile phone cameras. Specifically, we discover several new attacks that are based on the use of phone cameras. We implement the attacks on real phones, and demonstrate the feasibility and effectiveness of the attacks. Furthermore, we propose a lightweight defense scheme that can effectively detect these attacks.

I.INTRODUCTION

Since 2007, the Android operating system (OS) has enjoyed an incredible rate of popularity. As of 2013, the Android OS holds 79.3 percent of global smartphone market shares. Meanwhile, a number of Android security and privacy vulnerabilities have been exposed in the past several years. Although the Android permission system gives users an opportunity to check the permission request of an application (app) before installation, few users have knowledge of what all these permission requests stand for; as a result, they fails to warn users of security risks. Meanwhile, an increasing number of apps specified to enhance security and protect user privacy have appeared in Android app markets. Most large anti-virus software companies have published their Android-version security apps, and tried to provide a shield for smartphones by detecting and blocking malicious apps. In addition, there are data protection apps that provide users the capability to encrypt, decrypt, sign, and verify signatures for private texts, emails, and files. However, mobile malware and privacy leakage remain a big threat to mobile phone security and privacy.

Generally, when talking about privacy protection, most smartphone users pay attention to the safety of SMS, emails, contact lists, calling histories, location information, and private files. They may be surprised that the phone camera could become a traitor; for example, attackers could stealthily take pictures and record videos by using the phone camera. Nowadays, various types of camera-based applications have appeared in Android app markets (photography, barcode readers, social networking, etc.). Spy camera apps have also become quite popular. As for Google Play, there are nearly 100 spy camera apps, which allow phone users to take pictures or record videos of other people without their permission. However, believe it or not, phone users themselves could also become victims.

Attackers can implement spy cameras in malicious apps such that the phone camera is launched automatically without the device owner’s notice, and the captured photos and videos are sent out to these remote attackers. Even worse, according to a survey on Android malware analysis [1], camera permission ranks 12th of the most commonly requested permissions among benign apps, while it is out of the top 20 in malware. The popularity of camera usage in benign apps and relatively less usage in

malware lower users' alertness to camera-based multimedia application attacks.

1.1 OBJECTIVE:

Nowadays, people carry their phones everywhere; hence, their phones see lots of private information. If the phone camera is exploited by a malicious spy camera app, it may cause serious security and privacy problems. For example, the phone camera may record a user's daily activities and conversations, and then send these out via the Internet or multimedia messaging service (MMS). Secret photography is not only immoral but also illegal in some countries due to the invasion of privacy. Nevertheless, a phone camera could also provide some benefits if it is con-trolled well by the device owner. For example, when the owner wants to check if someone has used his/her phone without permission, the phone camera could be used to record the face of an unauthorized user. Besides, it can also help the owner find a lost phone.

In this article, we first conduct a survey on the threats and benefits of spy cameras. Then we present the basic attack model and two camera based attacks: the remote-controlled real-time monitoring attack and the passcode inference attack. We run these attacks along with popular antivirus software to test their stealthiness, and conduct experiments to evaluate both types of attacks. The results demonstrate the feasibility and effectiveness of these attacks. Finally, we propose a lightweight defense scheme.

II. EXISTING SYSTEM:

Several video-based attacks targeted at keystrokes have been proposed. The attacks can obtain user input on touch screen smart phones. Maggi *et al.* implement an automatic shoulder surfing attack against modern touch-enabled smart phones. The attacker deploys a video camera that can record the target screen while the victim is entering text. Then user input can be reconstructed solely based on the keystroke feedback displayed on the screen.

DRAWBACKS IN EXISTING SYSTEM:

- It works only when visual feedback such as magnified keys are available.
- Mobile malware and privacy leakage remain a big threat to mobile phone security and privacy

III.LITERATURE SURVEY:

TITLE : Dissecting Android Malware: Characterization and Evolution
AUTHOR : Y. Zhou and X. Jiang,
YEAR : 2012

DESCRIPTION

The popularity and adoption of smart phones has greatly stimulated the spread of mobile malware, especially on the popular platforms such as Android. In light of their rapid growth, there is a pressing need to develop effective solutions. However, our defense capability is largely constrained by the limited understanding of these emerging mobile malware and the lack of timely access to related samples. In this paper, we focus on the Android platform and aim to systematize or characterize existing Android malware. Particularly, with more than one year effort, we have managed to collect more than 1,200 malware samples that cover the majority of existing Android malware families, ranging from their debut in August 2010 to recent ones in October 2011. In addition, we systematically characterize them from various aspects, including their installation methods, activation mechanisms as well as the nature of carried malicious payloads. The characterization and a subsequent evolution-based study of representative families reveal that they are evolving rapidly to circumvent the detection from existing mobile anti-virus software. Based on the

evaluation with four representative mobile security software, our experiments show that the best case detects 79.6% of them while the worst case detects only 20.2% in our dataset. These results clearly call for the need to better develop next-generation anti-mobile-malware solutions.

TITLE : A Fast Eavesdropping Attack against Touchscreens.

AUTHOR : F. Maggi, et

YEAR : 2011.

DESCRIPTION

The pervasiveness of mobile devices increases the risk of exposing sensitive information on the go. In this paper, we arise this concern by presenting an automatic attack against modern touchscreen keyboards. We demonstrate the attack against the Apple iPhone - 2010's most popular touchscreen device - although it can be adapted to other devices (e.g., Android) that employ similar key-magnifying keyboards. Our attack processes the stream of frames from a video camera (e.g., surveillance or portable camera) and recognizes keystrokes online, in a fraction of the time needed to perform the same task by direct observation or offline analysis of a recorded video, which can be unfeasible for large amount of data. Our attack detects, tracks, and rectifies the target touchscreen, thus following the device or camera's movements and eliminating possible perspective distortions and rotations In real-world settings, our attack can automatically recognize up to 97.07 percent of the keystrokes (91.03 on average), with 1.15 percent of errors (3.16 on average) at a speed ranging from 37 to 51 keystrokes per minute.

IV.PROPOSED SYSTEM:

In this article, we first conduct a survey on the threats and benefits of spy cameras. Then we present the basic attack model and two camera based attacks: the remote-controlled real-time monitoring attack and the passcode inference attack. We run these attacks along with popular antivirus software to test their stealthiness, and conduct experiments to evaluate both types of attacks. The results demonstrate the feasibility and effectiveness of these attacks.

ADVANTAGES IN PROPOSED SYSTEM:

- The attacker needs considerable effort in translating central processing unit puzzle software to its functionally equivalent GPU version such that the translation cannot be done in real time.
- Moreover, we show how to implement puzzle in the generic server-browser model. To outsourcing any business onto a cloud.
- By using this Applications, we can easily be avoided by selecting the time to launch attack.
- The malicious camera app can periodically check the screen status and run the stealthy video recording only when the screen is off, which means that the user is not using the phone and the camera device is idle.

V.IMPLEMANTATION

The seriousness of the DoS/DDoS problem and their increased frequency has led to the advent of numerous defense mechanisms. In this paper, we are particularly interested in the countermeasures to DoS/DDoS attacks on server computation power. DoS and DDoS are effective if attackers spend much less resources than the victim server or are much more powerful than normal users.

There are five modules for the Software Puzzle.

5.1MODULES

- USER INTERFACE DESIGN
- GPU-INFLATED DOS ATTACK
- PUZZLE GENERATION
- CODE PROTECTION

- SECURITY ANALYSIS

- **User Interface Design:**

This is the first module of our project. The important role for the cloud user is to move login window to cloud user window. This module has created for the security purpose. In this login page we have to enter login user id and password. It will check username and password is match or not (valid user id and valid password). If we enter any invalid username or password we can't enter into login window to user window it will shows error message. So we are preventing from unauthorized user entering into the login window to user window. It will provide a good security for our project. So server contain user id and password server also check the authentication of the user. It well improves the security and preventing from unauthorized user enters into the network. In our project we are using JSP for creating design. Here we validate the login user and server authentication.

- **GPU-Inflated Dos Attack:**

In order to elaborate software puzzle, we recap its rival GPU-inflated DoS attack in advance. When a client wants to obtain a service, she sends a request to the server. After receiving the client request, the server responds with a puzzle challenge *x*. If the client is genuine, she will find the puzzle solution *y* directly on the host CPU, and send the response *(x, y)* to the server. However, as shown in Fig. 1, by using the similar mechanism in accelerating calculation with GPU, a malicious user who controls the host will send the challenge *x* to GPU and exploit the GPU resource to accelerate the puzzle-solving process.

Since the virtual keyboard in a touch screen smartphone is much smaller than computer keyboards, the virtual keys are very close to each other. Based on measurement of a Galaxy Nexus 4 phone, even an offset of 5 mm could result in touching the wrong key. Hence, when typing, users tend to keep a short distance to the screen, which allows the phone (front) camera to have a clear view of a user's eye movements. A user's eyes move along with the keys being touched, which means that tracking the eye movement could possibly tell what the user is entering. Thus, it is of great importance to investigate whether an attacker could obtain a phone user's passcode by tracking the eye movements.

- **Puzzle Generation:**

In order to construct a puzzle, the server has to execute three modules: puzzle core generation, puzzle challenge generation, puzzle encrypting/obfuscating.

1) **Puzzle Core Generation***:* From the code block warehouse, the server first chooses *n* code blocks based on hash functions and a secret, e.g., the *j* th instruction block bi_j , where $i_j = H1(y, j)$, and $y = H2(key, sn)$, with one-way functions $H1(\cdot)$ and $H2(\cdot)$, *key* is the server's secret, and *sn* is a nonce or timestamp. All the chosen blocks are assembled into a puzzle core, denoted as $C(\cdot) = (bi1 ; bi2 ; \cdots ; bin)$. As an illustrative example, Table III in the appendix shows an example puzzle core *C* generated from AES operation blocks stored in warehouse **S**.

2) **Puzzle Challenge Generation***:* Given some auxiliary input messages such as IP addresses, and in-line constants, the server calculates a message *m* from public data such as their IP addresses, port numbers and cookies, and produces a challenge $x = C(y,m)$, smiliar to encrypting plaintext *m* with key *y* to produce ciphertext *x*. As the attacker does not know the puzzle core $C(\cdot)$ (or equivalently the puzzle function $P(\cdot)$) in advance, it can not exploit GPU to solve the puzzle $C0x$ in real time using the basic GPU-inflated DoS attack addressed in Subsection III-A. Nonetheless, if the puzzle is merely constructed as aboveit is possible for an attacker to generate the GPU kernel by mapping the CPU instructions in $C0x$ to the GPU instructions one by one, i.e., to automatically

translate the CPU software puzzle *C0x* into its functionally equivalent GPU version.

- **Code Protection:**

  Intuitively, code obfuscation is able to thwart the above translation threat to some extent. Though there are no generic obfuscation techniques which can prevent a patient and advanced hacker from understanding a program results in show that obfuscation does increase the cost of reverse-engineering. Thus, although code obfuscation may be not satisfactory in long-term software defense against hacking, it is suitable for fortifying software puzzles which demand a protection period of several seconds only.
  A puzzle consists of instructs, and each instruction has a form (opCode, [operands]), where opCode indicates which operation (*e.g.*, addition, shift, jump) is, while
  the operands, varying with opCode, are the parameters (*e.g.*, target address of jump instruction) to complete the operations. As a popular obfuscation technology, code encryption technology treats software code as data string and encrypts both operand and opCode.

- **Security Analysis:**

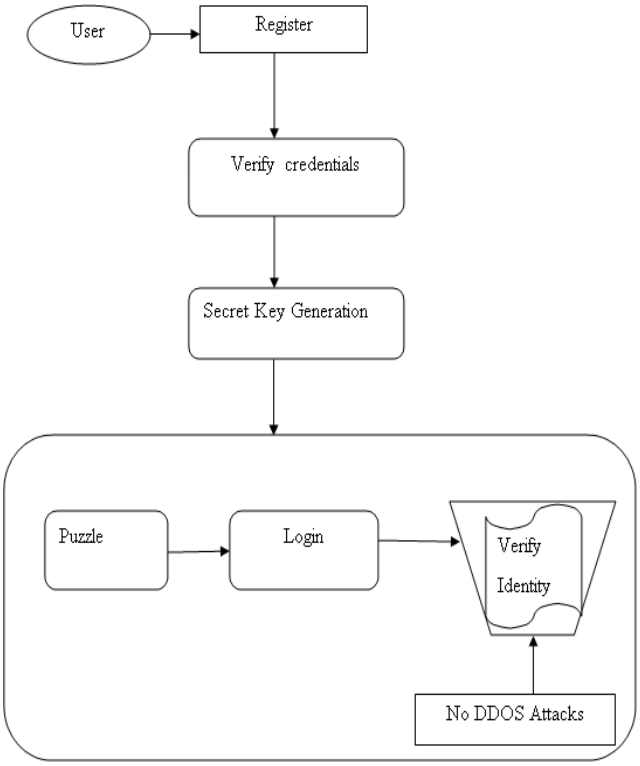
In this module puzzle aims to prevent GPU from being used in the puzzle-solving process based on different instruction sets and real-time environments between GPU and CPU. Conversely, an adversary may attempt to deface the puzzle scheme by simulating the host on GPU (Subsection V-A), cracking puzzle algorithm (Subsection V-B), re-producing GPU-version puzzle (Subsections V-C ~ V-E), or abusing the access priority in puzzle-solving (Subsection V-F).

5.2 SYSTEM TECHNIQUES:

Algorithm: Cracking Data Puzzle Algorithm

The practical strategy of the attacker is to accelerate the brute force process by exploiting the parallel computation capability of GPU cores. We classify client puzzles into two types. If a puzzle functions *P*, as all the existing client puzzle schemes, is fixed and disclosed in advance, the puzzle is called a data puzzle; otherwise, it is referred to as a software puzzle.

VI.SYSTEM ARCHITECTURE



EXPLANATION:

The systems architect establishes the basic structure of the system, this we know about that the practical strategy of the attacker is to accelerate the brute force process by exploiting the parallel computation capability of GPU cores. We classify client puzzles into two types. If a puzzle functions *P*, as all the existing client puzzle schemes, is fixed and disclosed in advance, the puzzle is called a data puzzle; otherwise, it is referred to as a puzzle. To ensure

challenge data confidentiality and code security for an appropriate time period**.** After receiving the puzzle sent from the server, a client tries to solve the software puzzle on the host CPU, and replies to the server, as the conventional client puzzle scheme does.

VII.APPLICATION

A Network application is any application running on one host and provides a communication to another application running on a different host, the application may use an existing application layer protocols such as: HTTP(e.g. the Browser and web server), and may be the application does not use any existing protocols and depends on the socket programming to communicate to another application. So the web application is a type of the network applications.

We can distinguish between **network** and **stand-alone** applications. For example, if you use Microsoft Word to write a letter and save it on your PC, both the program and the data are stored on your computer. Since your computer does not have to be connected to a network, this is an example of a stand-alone application.

VIII.FUTURE ENHANCEMENTS

GPU-inflation attack, its idea can be extended to thwart DoS attackers which exploit other inflation resources such as Cloud Computing. For example, suppose the server inserts some anti-debugging codes for detecting Cloud platform into software puzzle, when the puzzle is running.

IX.CONCLUSION

In this article, we study camera-related vulnerabilities in Android phones for mobile multimedia applications. We discuss the roles a spy camera can play to attack or benefit phone users. We discover several advanced spy camera attacks, including the remote-controlled real-time monitoring attack and two types of passcode inference attacks. Meanwhile, we propose an effective defense scheme to secure a smartphone from all these spy camera attacks. In the future, we will investigate the feasibility of performing spy camera attacks on other mobile operating systems.

REFERENCES

1.Y. Zhou and X. Jiang, "Dissecting Android Malware: Characterization and Evolution", IEEE Symp. Security and Privacy 2012, pp. 95-109, 2012.

2.R. Schlegel, "Soundcomber: A Stealthy and Context Aware Sound Trojanfor Smartphones", NDSS, pp. 17-33, 2011.

3.N. Xu, "Stealthy Video Capturer: A New Video-Based Spyware in 3g Smartphones", Proc. 2nd ACM Conf. Wireless Network Security, pp. 69-78, 2009.

4.F. Maggi, "A Fast Eavesdropping Attack against Touchscreens", 7th Int'l. Conf. Info. Assurance and Security, pp. 320-25, 2011.

5.R. Raguram, "ispy: Automatic Reconstruction of Typed Input from Compromising Reflections", Proc. 18th ACM Conf. Computer and Commun. Security, pp. 527-36, 2011.

6."Android-eye", 2012.

7."Nanohttpd".

8.A. P. Felt and D. Wagner, "Phishing on Mobile Devices", Proc. WEB 2.0 Security and Privacy, 2011.

9.D. Li, D. Winfield and D. Parkhurst, "Starburst: A Hybrid Algorithm for Video-Based Eye Tracking Combining Feature-Based and Model-Based Approaches", IEEE Computer Soc. Conf. Computer Vision and Pattern Recognition Workshops, pp. 79, 2005.

10.P. Aldrian, "Fast Eyetracking", 2009.