IISJ CZ-WNS
INTERNATIONAL INTERDISCIPLINARY SCIENTIFIC JOURNAL CZ-WNS
internationalscientificjournalnews.eu
Volume 4, pp. 35-44

Zhalgasbek Iztaev, Sandugash Mombekova, Kulaysh, Ayhinbay
South-Kazakhstan State University im. M. Auezova
Shymkent, Kazakhstan
e-mail address: san.mom@inbox.ru

ANALYSIS OF SAFETY AND VULNERABILITIES OF THE LEVELS OF THE INFRASTRUCTURE AND APPLICATIONS ANDROID

Abstract

In this article, we will focus on analyzing the security and vulnerabilities of infrastructure and application levels. Security mechanisms, such as the sandbox and Android permission systems, exist at the infrastructure level, while malware scanners protect the application layer. However, there is room for improvement in both mechanisms. For example, it is known that the Android permissions system is implemented irregularly and not sufficiently tested for vulnerabilities. The application layer is also focused mainly on the detection of malicious applications, while in the application markets there are various types of malicious applications.

The purpose of this article is to address these security gaps by analyzing vulnerabilities on mobile platforms and identifying policy-breaking applications. As a result of our analysis, we find various vulnerabilities at the level and can run serious validation concepts on Android platforms. We also offer mechanisms for detecting policy violation and disguised applications. It is shown that our methods improve the security of mobile systems and have several implications for the mobile industry.

Keywords: applications, system operations, platforms, analysis of system applications, unsecured receivers.

Introduction

Mobile systems typically consist of three levels of software: an application layer where third-party applications are installed, an infrastructure level where application programming interfaces (APIs) open, and a kernel level where low-level system operations are performed.

Mobile devices play an important role in modern life, and there are billions of mobile users worldwide. Platform providers have a great responsibility to ensure the security and privacy of mobile users. In this article, we study the security

of the application layer and the level of the infrastructure of mobile systems.

1. Access rights to important resources on mobile devices

First, platform providers include several security mechanisms, such as application sandboxes and access control, at the mobile device platform level. The sandbox engine isolates code execution and storage of application data on mobile devices to minimize the damage that can be caused by malicious applications. At the same time, access control mechanisms allow applications with the appropriate permissions to access important resources on mobile devices.

Second, at the application level, platform providers check applications when they are loaded into application repositories. Therefore, they remove malicious applications after they are detected. There are several drawbacks to these mechanisms:

Currently, there is no standard way to analyze vulnerabilities for Android platforms. The lack of framework-specific vulnerability analysis tools makes security testing difficult, which can lead to various vulnerabilities. On the other hand, securing mobile frameworks is not trivial due to the wide variety of API types and the extensive presence of the API.

Platforms are also constantly being modified by various platform providers at a very fast pace mobile environment. [1]

Security control in mobile applications has focused on malware, which accounts for only a small percentage of mobile applications in the markets.

There has been negligence in dealing with bad applications that are less aggressive than malware, but still violate developer policies, such as intellectual property rights violations. In this article, we take the first step to systematically analyzing vulnerabilities in mobile environments and discover policy-breaking applications. We consider a third-party application as an attacker. This attacker gains access to mobile resources, bypassing the security mechanisms of the mobile infrastructure.

The results of these two works are attacks to confirm concepts that can be performed on mobile platforms. In this paper, we conduct an empirical analysis of applications that violate policies, and create detection mechanisms for all applications that violate Google Play policies. After that, we will focus on detecting masked applications that are also part of policy-breaking applications.

2. Identify vulnerabilities in the Android Framework

Android requires third-party applications to request permissions when accessing critical mobile resources, such as personal user information and system operations. For example, only applications with Android permission. CAMERA get access to the cameras of the phone. In this article, we present attacks that can be launched without permissions. We integrate the Android APIs into three categories: system services, system applications, and dynamically register translations.

To identify all the vulnerabilities, we perform interprocedural analysis of the call graph for system services and discover all the interfaces of the Android Interface Definition Language (AIDL), which are not protected by any authorization checks or Linux ID verification mechanisms. Then we perform component analysis of system applications to find unprotected receivers, actions, and services.

After that, we conduct in-process data flow analysis to detect unprotected dynamically registered broadcast messages from both system services and system applications.

The result of our analysis is a systematic review of unsecured Android APIs. These insecure APIs provide a way to access resources without any permissions. Then we use the selected unprotected APIs and launch a series of attacks on Android phones. In particular, we launch attacks

using Java reflections, attacks using broadcasts, a hijacking attack, an attack to launch malicious actions, an activity-capture attack, an attack to launch malicious services, and an attack on service seizures. [2]

We find that without requesting any permissions, an attacker can gain access to the device ID, telephone status, SIM status, Wi-Fi and network information, as well as information about user settings, such as the aircraft, location, NFC, USB and power modes for mobile devices. An attacker could also disrupt Bluetooth discovery services and block incoming emails, calendar events and Google documents.

Moreover, an attacker can set the volume of devices and trigger alarms and ringtones that users personally set for their devices. An attacker can also launch camera, mail, music, and phone applications, even when devices are locked.

We compare our study of two versions of Android and find that as platform providers implement more APIs, the number of unprotected APIs increases and new attacks become possible. This is contrary to the popular belief that the security of the new version should improve, as many of the security flaws in the old version are reported and corrected.

To ensure the quality and reliability of mobile applications in the Google Play store, we apply developer policies covering various aspects, including intellectual

property rights, spam and advertising. As soon as the application reports suspicious behavior that violates the application’s policies, it is removed from the repository to protect users.

Currently, the Google Play store uses reviews from mobile users to identify violations. Our work takes the first step towards understanding these declared applications by performing an empirical analysis of real-world application samples. We scan 302 Android applications that violate the policies reported on the Reddit forum by mobile users, and then removed from the Google Play store.

Our empirical analysis shows that many behavioral disorders have not been well studied by industry or research communities. We found that 53% of claimed applications either copy popular applications, or violate copyrights or trademarks of brands such as Adobe, Disney, Minion, Despicable and Pikcachu.

In addition, 49% of the applications claimed violate advertising rules by sending push notifications, adding a desktop icon and changing browser settings. Many applications also exhibit behaviors similar to malware, such as downloading malicious files to users' mobile phones, redirecting users to other applications on the market, and accessing PayPal's user accounts.

Based on the results of our empirical analysis, we extracted 208 functions that distinguish bad applications from ordinary applications. Our functions include the use of brand names and other keywords, third-party libraries, network operations, metadata, permissions, and suspicious API calls from third-party libraries.

The first three groups of functions are based on an empirical analysis of samples of our applications, and the last three groups of functions are based on their bad behavior. We applied 10 machine learning classifiers to the extracted functions to detect declared bad applications. Our experimental result shows that we can detect them with 86.80% true positive indicator and 13.6% false negative indicator.

Our work highlights the problem of policy-breaking applications and suggests revising the current strategy for maintaining high-quality mobile application markets. [3]

Detection of disguised applications. Application plagiarism or application cloning is a new threat in the mobile application markets. This reduces the profits of the original developers and sometimes even damages the security and privacy of users.

Conclusion

In this article, we present a new concept, called hidden applications, into which external functions of mobile applications are copied, such as icons, screenshots, names or descriptions of applications. We then offer a scalable detection environment that can find these suspiciously similar disguised applications.

To do this, we use text-based search methods and image-based image search methods in our environment. Our framework is implemented and tested with 30,625 Android applications from the official Google Play market.

Experimental results show that even the official market consists of 477 potential victims in disguise, which cover 1.56% of the tested samples. Our work emphasizes that these disguised applications not only pose potential security threats, but also degrade the quality of mobile application markets. Our work also analyzes the behavior of detected masked applications and calculates the frequency of false positives of the proposed structure.

References

1. Goloshchapov A. A. Google Android: programming for mobile devices. – SPb, BHV Petersburg 2014, p. 163.
2. Sokolov V.V. Computer equipment and information technology. Mobile application development. Tutorial. – M, Yurayt 2016, p.176.
3. Vale E. HTML5. Mobile application development. – SPb, Peter 2015, p. 225.