Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 5.258

IJCSMC, Vol. 5, Issue. 5, May 2016, pg.204 – 208

Survey on Security Issues with Possible Solutions of Android

Shifa Behal, Tarun Bagga
CSE & Kurukshetra University, India
CSE & Kurukshetra University, India
behal.shifa@gmail.com, tarunbagga07@gmail.com

Abstract— Android framework enforces a permission-based security policy. Android has been a major target how to detect and keep the malapps out of the app markets. Android permissions has many attacks and security issues. A state-based model which determine the behavior specification of permissions authorization and the interactions between application components. Algorithms SVM, Decision Tree and Random Forest helpful to detect malapps. Static analyses are capable of analyzing the Android framework rather than dynamic analysis. The main objective of this paper is to achieve security enhancement, permission security, security from attacks and malicious applications of android.

I. INTRODUCTION

Android: Android is an open platform for mobile devices whose application framework enforces the permission-based security policy that allows installed applications to access other parts of the system when they are explicitly permitted to do so. The permission is granted by user when an application is installed on that device. This scheme is simple and easy to understand. The platform source codes are public. Anyone can write and deploy an application without having being certified by Google, Inc. or Open Handset Alliance.

Security Architecture Of Android: An Android application has four types of application component:
Activity: The Activity runs in the foreground interacting with users via the screen.
Service: Service is running in the background.
Content Provider: The Content Provider provides data storage for the components.
Broadcast Receiver: The Broadcast Receiver helps others inter-communication.

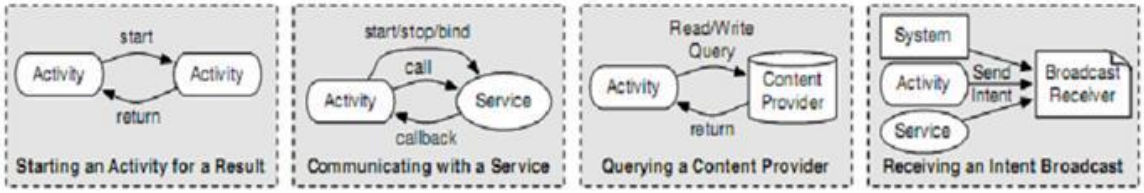

Fig: ICC between the different components

Android Manifest file: The Android application package is an organization of an application, which follows an XML format descriptor file, called the Android manifest file. The manifest file composed of the following authorization-related information:

1. List of application sub-components: The application is composed of a set of XML sub-elements for activity component, service component, content provider component, and broadcast receiver component.
2. List of permission declarations: (permission) element is used to declare a permission in an application and permission is added to a system when an application is installed.
3. List of permissions expected to be granted: An application lists the permissions needed to perform its task, using (uses permission) element. The permissions are requested at the time of installation, and are listed on the screen.
4. List of permissions used for protection: The android: permission attribute is used for the protection.

Protection level: The protection level attribute of a permission determines how the permission is granted. The current security architecture of Android have four protection levels: normal, dangerous, signature, and signature Or System.

Dangerous-level: When an application is installed then these permissions are listed on the screen.

Normal-level: The permissions of normal level are hidden in a folded menu on the screen.

Signature Or System-level: The process of granting the permissions for the signature level or signature Or System-level permission requires certificate comparison. The signature-level permission is granted only when the application that requests the permission and the application that declares the permission are signed with the same certificate. The signature Or System-level permission is granted as the signature-level permission.

Android Malware and Security Issues: There are hence four types of malware categorized by Troy Vennon, namely, (8)

Virus:- A virus is defined as a malicious program that lacks the capacity to self-reproduce.

Worm:- worm is a malicious code that can control a system or a network in order to automatically duplicate to another system.

Trojan:- A Trojan allows an attacker to obtain unauthorized access to a system while it seems to be executing a required operation.

Spyware:- This destructive application cover up itself from the user while it collects.

Authors Kamran Habib Khan & Mir Nauman Tahir have categorized the attacks on mobile phones as below (4):-

Ad-ware:- This category includes the advertisements on cell phones.

Direct Payoff:- This category consists of malwares that send SMS without the permission of the user.

Destructive:- The erasure of phonebook entries without the user's knowledge is an example of these kind of attacks.

Information Scavengers:- This type includes the checking of cookies, address books and passwords without user's permission.

Premeditated Spyware:- This attack represents remote listening and location tracking.

II. LITERATURE REVIEW

Wook Shin.et.al(2010) This paper represents a flaw in the permission scheme of Android. The security of the framework depends on the owner of a device since the authorization decisions are mainly made by the user. As a result, the permission scheme enforces administrative burden on the user instead of keeping it simple. Moreover, the framework does not apply no naming rule or constraint for a new permission declaration; therefore two different permissions can have the same name. These features results in a security flaw.

Toshiaki Tanaka.et.al(2010) This paper proposes a formal model of the Android permission scheme. This scheme specify entities and relationships, and also provides a state-based model which represents the behavior of permission authorization and the interactions between components of application. The proposed model can be used as a reference model when the permission scheme is implemented in a different embedded platform, or when the current scheme is extended with additional constraints or elements. This is the first formalization of the permission scheme enforced by the Android framework.

Yajin Zhou.et.al(2012) This paper focus on the Android platform to characterize existing Android malware. More than one year efforts, it collect about 1,200 malware samples that cover the existing Android malware families. In addition, this characterize them from various aspects, like their installation methods, activation mechanisms as well as the nature payloads. The evolution-based study of these families reveal that they are evolving rapidly detection from existing mobile anti-virus software. The best case detects 79.6% of them while the worst case detects only 20.2% in our dataset.

Danyang Jiang.et.al(2012) This paper presents the analysis of Permission-based security model and the permissions features of Android applications. The permission model quantify the functional characteristics of the application, and then provides an assessment method which use the network visualization techniques and clustering algorithm which determines whether the testing application is malicious application or not. Then test the assessment method on 873 applications available online and analysis the results to find that this method can do efforts in finding potentially malicious applications.

Don Manuel.et.al(2013) This paper proposes Flow Permissions. Flow Permissions allow users to examine and grant explicit information flows within an application (for e.g., a permission for reading the phone number and sending it over the network) and also implicit information flows across multiple applications (for e.g., a permission used for reading the phone number and sending it to another application that is already installed on the user's phone).Flow Permissions are used to provide visibility into the behavior of the applications installed on a user's phone.

Wei Wang.et.al(2014) This paper explore the permission-induced risk in Android applications on three levels in a systematic manner. First, analyze the risk of an individual permission and the risk of a group of collaborative permissions. Second, then evaluate the usefulness of risky permissions for malapp detection with support vector machine, decision trees and random forest. Third, analyze the detection results and discuss the feasibility and the limitations of malapp detection based on permission requests.

Alexandre Bartel.et.al(2014) This paper propose the Static Analysis for Extracting Permission Checks. A common security architecture is based on the protection of resources by permission checks. The analysis of permission-based framework requires a precise mapping between API methods of framework and the permissions they require. This paper show that static analysis fails when applied with stored components on the Android framework. Then it presents an advanced class-hierarchy and field-sensitive set of analysis to extract this mapping. These advanced static analysis are capable of analyzing the Android framework.

S. NO.	AUTHOR	YEAR	TITLE	MERITS	CONCLUSION
1	Wook Shin	2010	A Small but Non-negligible flaw in the Android Permission Scheme	Naming or constraint rule helps in finding security flaw	found the flaw, exploitation was successful in several versions of android SDK
2	Toshiaki Tanaka	2010	A Formal Model to Analyse the Permission Authorization and Enforcement in the Android Framework	model can be used as a reference model when the permission scheme is implemented in a different platform	The security of the system is mathematically verifiable with the help of a formal model.
3	Yajin Zhou	2012	Dissecting Android Malware: Characterization and Evolution	Characterize their installation methods, activation mechanisms and nature payloads.	the best case detects 79.6%, worst case detects only 20.2%.
4	Danyang Jiang	2012- 2013	Assessment Method For Android Applications Based On Permission Model	provide an assessment method which use clustering algorithm to determine whether application is malicious or not.	can identify a part of malicious applications which belong to the same category.
5	Don Manuel	2013	Flow Permissions for Android	user can examine and grant explicit information and implicit information.	Visibility into the behavior of the applications installed on a user's phone.
6	Wei Wang	2014	Exploring Permission-Induced Risk in Android Applications for Malicious Applications Detection	several algorithms SVM, Decision Tree and Random Forest helpful to detect malapps	risky permissions are effective for the detection of malapps
7	Alexandre Bartel	2014	Static Analysis for Extracting Permission Checks of a Framework	Provide class-hierarchy and field-sensitive set to analyze framework	static analysis result better than dynamic analysis for analyzing Androids permissions

III. CONCLUSIONS

This paper has been studied on security enhancement. A Formal Model to Analyze the Permission Authorization can be used as a reference model when the permission scheme is implemented in a different platform. Dissecting Android Malware Characterize their installation methods, activation mechanisms and nature payloads and the best case detects 79.6%, worst case detects only 20.2%. Algorithms SVM, Decision Tree and Random Forest helpful to detect malapps. Static Analysis is used for Extracting Permission Checks of a Framework. Assessment method which use clustering algorithm to determine whether application is malicious or not.

References

[1] W. Shin, S. Kiyomoto, K. Fukushima, and T. Tanaka, "A formal model to analyze the permission authorization and enforcement in the android framework," in International Symposium on Secure Computing (SecureCom-10) (to appear), 2010.
[2] D. F. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn, and R. Chandramouli, "Proposed NIST standard for role-based access control," ACM Trans. Inf. Syst. Secur., vol. 4, no. 3,
pp. 224–274, 2001.
[3] (2011) Smartphone Shipments Tripled Since '08. Dumb Phones Are Flat. http://tech.fortune.cnn.com/2011/11/01/smartphone-shipments-tripled-since-08-dumb phones-areflat.
[4] William Enck, M. O., and Patrick McDaniel (2009). "Understanding Android Security." IEEE: 8.
[5] A. P. Felt, H. J.Wang, A. Moshchuk, S. Hanna, and E. Chin, "Permission re-delegation: attacks and defenses," in Proceedings of the 20th USENIX conference on Security, SEC'11, 2011.
[6] A. P. Felt, H. J.Wang, A. Moshchuk, S. Hanna, and E. Chin, "Permission re-delegation: attacks and defenses," in Proceedings of the 20th USENIX conference on Security, SEC'11, 2011.
[7] L. O. Andersen, "Program analysis and specialization for the C programming language," Ph.D. dissertation, Dept. Comput. Sci., Univ. Cophenhagen, Kbenhavn, Denmark, 1994.

