Permission Management in Android Based System

Jincy Joseph
UG Student, Department of Computer Science and Engineering
Rajarajeswari College of Engineering
Bengaluru, India

Meghana P
UG Student, Department of Computer Science and Engineering
Rajarajeswari College of Engineering
Bengaluru, India

Kavya G
UG Student, Department of Computer Science and Engineering
Rajarajeswari College of Engineering
Bengaluru, India

Poonam Kumari
Assistant professor, Department of Computer Science and Engineering
Rajarajeswari College of Engineering
Bengaluru, India

***Abstract*—Smartphones are becoming an integral part of everyday life with users using it to store many private and sensitive data. However, the mobile applications have access to the sensitive data and may expose users to high security and privacy risks. Hence, permission management plays a very important role. The permission manager app will allow user to enable or disable the permission that are requested by the app. This provides the control in the hands of the user to decide which resource can be accessed by the app.**

Keywords*— *Smart phone, permission manager, application, privacy, security.

I. INTRODUCTION

Nowadays, smartphones that are developed have very high computational and communication capabilities. The application developers are taking advantage of these capabilities to provide enhanced services to their applications. Resolving the tension between the fun and utility of running third-party mobile applications and the privacy risks they pose is a critical challenge for smartphone platforms. When users install third-party applications on their smartphones, several security concerns may arise. For instance, malicious applications may access user's emails, SMS and many other confidential data stored in the smartphone exposing users to high security and privacy risks if applications use them inappropriately and without the user's knowledge.

The standard security model in Android based Smartphones provide only coarse-grained controls for regulating whether an application can access private information, but provide little insight into how private information is actually used. For example, if a user allows an application to access their location information, the users have no way of knowing if the application will send their location to a location-based service, to advertisers, to the application developer, or to any other entity. As a result, users must blindly trust that applications will properly handle their private data.

Hence users must be able to have a better control over their device capabilities by reducing certain application privileges. To achieve this, smartphone systems must allow mobile users to control their device usage of system resources and application privileges.

In this paper, we propose Permission manager App for Android systems that allows smartphone users to set restrictions over their applications' usage of device resources and services. Through this app, users can, for example, set restricted privileges for device applications when using the device at work, and provide the original privileges when the device is used at home. The user is also allowed to report about any malicious apps present in their smartphones.

II. TECHNOLOGICAL ISSUES

A. Java

Java is a programming language originally developed by James Gosling at Sun Microsystems and released in 1995.The language derives much of its syntax from C and C++ but has a simpler object model and fewer low-level facilities. Java applications are typically compiled to bytecode (class file) that can run on any Java Virtual Machine (JVM) regardless of computer architecture. Java is a general-purpose, concurrent, class-based, object-oriented language that is specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere". Java is currently one of the most popular programming languages in use, and is widely used from application software to web applications.

B. Android

Android is a software stack for mobile devices that includes an operating system, middleware and key applications. Google Inc. purchased the initial

developer of the software, Android Inc., in 2005.Android's mobile operating system is based on a modified version of the Linux kernel. Google and other members of the Open Handset Alliance collaborated on Android's development and release. The Android Open Source Project (AOSP) is tasked with the maintenance and further development of Android. The Android operating system is the world's best-selling Smartphone platform.

Android has a large community of developers writing applications ("apps") that extend the functionality of the devices. There are currently over 150,000 apps available Google Play Store , though which apps can be downloaded from third-party sites. Developers write primarily in the Java language, controlling the device via Google-developed Java libraries.

1) *Permission System:* The Android permission system controls which application has the privilege of accessing device resources and data. Application developers that need access to protected Android APIs need to specify the permissions they need in the AndroidManifest.xml file which, if inaccurately assigned, can increase the risks of exposing the users' data and increase the impact of a bug or vulnerability. Each application declares the permissions listed in its AndroidManifest.xml file at the time of installation, and users have to either grant all the requested permissions to proceed with the installation, or cancel the installation. The Android permission system does not allow users to grant or deny only some of the requested permissions, which limits the user's control of application's accessibility.

C. Eclipse

Eclipse is a multi-language software development environment comprising an integrated development environment (IDE) and an extensible plug-in system. It is written mostly in Java and can be used to develop applications in Java and, by means of various plugins, other programming languages. The IDE is often called Eclipse ADT for Ada, Eclipse CDT for C/C++, Eclipse JDT for Java, and Eclipse PDT for PHP.

Initial codebase originated from Visual Age. In its default form it is meant for Java developers, consisting of the Java Development Tools (JDT). Users can extend its abilities by installing plug-ins written for the Eclipse software framework, such as development toolkits for other programming languages, and can write and contribute their own plug-in modules. Released under the terms of the Eclipse Public License, Eclipse is free and open source software. It was one of the first IDEs to run under GNU Classpath and it runs without issues under IcedTea.

III. SYSTEM DESIGN

A. System Architecture

Using the Permission Manager App, users can configure the application restrictions by enabling or disabling the permissions granted to the app. When an application requests a resource or service, the Access Controller verifies at run-time whether the application request is authorized and forwards the request to the Policy Executer. If the request is authorized, the Policy Executer then checks if there is restrictions that corresponds to the application request. If it exists, the Policy Executer compares the permission restrictions with database and if it matches, it enforces the restrictions by reporting back to the Access Controller to apply those restrictions on the application request.

Fig 1: The system architecture of permission manager app

IV. IMPLEMENTATION

As shown in fig 2, the user can view all the installed apps or view the apps based on the list of permission. The app list shows a list of all installed apps as shown in fig 3. The user can then select the required app for which the permission has to be restricted. If a user clicks on a particular app, then the list of permissions of the app is displayed as shown in fig 4.

Fig 2:Welcome screen of the Permission Manager App

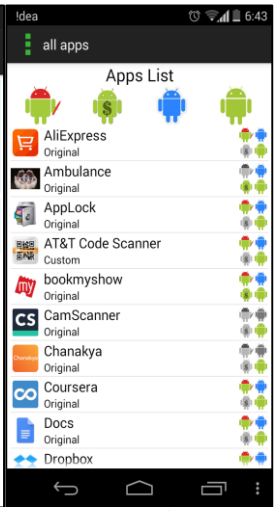

Fig 3:Display of installed apps

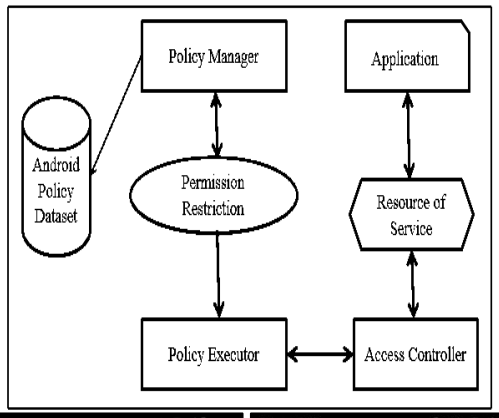

Fig 4: Permission list of Chankya app permission

Fig 5: Chanakya app after disabling

The user can uncheck the checkbox to disable the required permission. For example, if the user un-checks internet permission of an app named "Chanakya", the ads in the app will not be seen. This is shown in fig. 4.4 where below previously the ads would be displayed at the bottom but since it has no access to internet it's not displayed.

V. CONCLUSION

In this paper, we proposed an Android app called the "Permission Manager App" that allows user restrict applications from accessing specific data and/or resources. The restrictions get updated in the database and enforced during resource request. The user can later grant the permissions they had revoked before by enabling the permissions. However the app needs to uninstalled and reinstalled to enforce the permission restriction. Hence, in future enhancements this drawback needs to be overcomed.

ACKNOWLEDGEMENT

The documentation and implementation of this paper would not be succeeded without the kind support from Dr. Bhagyashekar M.S, Principal, RajaRajeswari College Of Engineering, Bengaluru, who always gives us valuable advice and kind assistance to complete this paper. We are also grateful to Dr. Usha Sakthivel, Professor and HOD, Department Of Computer Science and Engineering, RajaRajeswari College Of Engineering, Bengaluru, for giving us the great knowledge.

Finally, we would like to thank our parents who constantly supported us since the beginning till the end of the paper.

REFERENCES

[1] W. Enck, P. Gilbert, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth , "Taintdroid: an information-flow tracking system for real time privacy monitoring on Smartphone's," in Proceedings of the 9th USENIX conference on Operating systems design and implementation, ser. OSDI'10, Berkeley, CA, USA, 2010.

[2] J. Ligatti, B. Rickey, and N. Saigal, "Lopsil: A location-based policy-specification language."

[3] Y. Zhou, X. Zhang, X. Jiang, and V. W. Freeh, "Taming information-stealing Smartphone applications (on android)," in Proceedings of the 4th international conference on Trust and trustworthy computing, ser. TRUST'11. Berlin, Heidelberg: Springer-Verlag, 2011

[4] F. Roesner, T. Kohno, A. Moshchuk, B. Parno, H. J. Wang, and C. Cowan, "User-driven access control: Rethinking permission granting in modern operating systems," in Proceedings of the 2012 IEEE Symposium on Security and Privacy, ser. SP '12.

[5] M. Backes, S. Gerling, C. Hammer, M. Maffei, and P. von Styp-Rekowsky, "AppGuard—Real-Time Policy Enforcement for Third-Party Applications," Technical Report A/02/2012, SaarlandUniv., 2012

[6] Alexandre Bartel, Jacques Klein, Yves Le Traon and Martin Monperrus "Automatically Securing Permission-Based Software by Reducing the Attack Surface: An Application to Android"

[7] David Barrera, H. Günes Kayacık, P.C. van Oorschot, Anil Somayaji "A Methodology for Empirical Analysis of Permission-Based Security Models and its Application to Android," Carleton University, Ottawa, ON, Canada ACM CCS(2010)

[8] Yury Zhauniarovich, Giovanni Russello, Mauro Conti, Bruno Crispo, and Earlence Fernandes ,"MOSES: Supporting and Enforcing Security Profiles on Smartphones" in proceedings of the IEEE transactions on dependable and secure computing, May-June 2014