International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 04 Issue: 04 | Apr -2017 www.irjet.net p-ISSN: 2395-0072

Anti-Hijack: Real-Time Detection and Prevention of Attack on Android

[1]LOKESH KUMAR MISHRA, [2]RAHUL DEVKAR, [3]TUSHAR DESALE, [4]RAHUL BIDKAR,
[5] JAYANT WAGHALE, [6]PROF. S.R. BHAMARE

[123456]*Dept. of Computer Engineering, NDMVP's Karmaveer Baburao Thakare College of Engineering, Nashik, Maharashtra, India*

Abstract—*Examining the latest studies, Android have achieved the highest market value share in Smartphones. The Android applications are increasing numerously. Hence threats and attacks on Android technology are also rising. There are many applications which bypass user by hiding their functionalities and send user private and important information and data across the network. . In this paper, we are summarizing detection of two fatal malware attacks i.e. intent based hijacking and authenticated session based hijacking. We are using the honey-pot concept for detection of these two authentication hijacking attacks. To achieve this, we have tested various apps and their cooperation with the honey-pot maintained by android device i.e. smartphone or an emulator. We have designed benevolent app as a honey framed app. We quibble that hijacking malware would be detected with higher accuracy using our methodology at real-time by comparing the traditional machine learning strategies. Our detection method Anti-Hijack is a helping hand in this platform which provides light weight, device operated real-time detection at hijacking malware.*

Key words— **Vulnerabilities, Smartphone, Security, Honey-net, Spoofing Attack, Anti-spoofing, Android, Permissions.**

I. INTRODUCTION

One of the most widely used mobile OS these days is ANDROID. Android is a bunch of software comprising not only operating system but also middleware and key applications. People are contracting with the growth of mobile technology. As the number of users is increasing, also facilities are increasing. With old regular handsets which were used just for phone calls, mobiles have changed life and have become member of it. Now they are not used just for making calls but they have innumerable uses and can be used as a Camera, Music player, T.V., Web-browser etc. and with the new technologies, new software and operating systems are required. Specifically from some malware samples, we find that some of them are repackaged versions of legitimate applications with malign payloads, which indicates the policing need of detecting repackaged applications in the current Android Markets.

II. PROBLEM STATEMENT

This application is useful for the users to provide security to the mobile phones. In current time, there has been many cases where personal and confidential information of a user can be accessed by many other hijackers. This usually happens when we install a new application to our system and then all our personal data can be seen by others. Also, while doing banking transactions, we have experienced many cases where confidential data is shared to an unknown person (hijacker). For this, we have proposed a new version of security system, whether it checks the downloaded application or redirected URL is safe to access or not.

Thinking on the drawbacks and inadequacies of existing process, there is a great need of an efficient system. The

proposed system rectifies the demerits and existing process defects to a large extend.

III. LITERATURE SURVEY

Android is a Linux-based open-source operating system, with a layered structure of services including core native libraries and application frameworks. Android natively separates applications and provides safety through operating system primitives and environmental features such as type safety each software package is sandboxed that employs privilege separation as a matter of course. Applications request access to system resources using special application level permissions, such as which must be granted by the user. Android's permission model requires each application to explicitly request the right to access protected resources. The permission model is intended to prevent the unwarranted intrusion by an application on the user's data and the data of other applications, as well as limit access to features that directly or indirectly cause financial harm. Before installing an application, the user is presented with a list of all1 the permissions requested by that application which they must accept before installation begins Unprivileged application attacks can take advantage of the complexity of the Android permission model and the Android market to persuade users to install malicious software and grant applications the permissions required to deliver a harmful payload. Applications advertised in the Market must be signed by the author, but the user has no means of verifying that a signature is associated with any particular entity. Even users that carefully monitor permission requests may not fully understand the semantics of a particular permission or the framework behavior surrounding a permission.

SMS application will have the ability to process the incoming SMS message first, including the ability to prevent the broadcast from being observed by any other application. Malware employing covert communication channels in this manner has already been seen. When a user grants permissions based on prompts that show a RECEIVE_SMS permission, such behavior is likely unexpected. Similarly, the only means a user has of understanding the capabilities of an application is through the list of permissions presented at install time.

IV. PROPOSED SYSTEM

It is clear that intent attack are performed on benevolent app by seizing their intent and session. Therefore an enhancement of honey net functionality fit here. While testing our methodology we have assumed that honey-pots as real devices or an emulated environment which is used to monitor blocked system call and permissions between test app and our specially designed honey framed benevolent app. All the actions are saved in a log. This log file (SysCall) is used for auditing, detecting and testing vulnerable Android apps in complete functionalities of Anti-Hijack. We have initiated monkey-runner tool to record system call and permissions separately for testing app and designed benevolent app. After installing SDK, we have used test app, honey framed benevolent app, monkey runner tool, targeted android API in a single form. All these makes a system deployed as honey-pot. After using single system as honey-pot, we have initiated monkey runner tool to record system call separately for tested app and benevolent app.

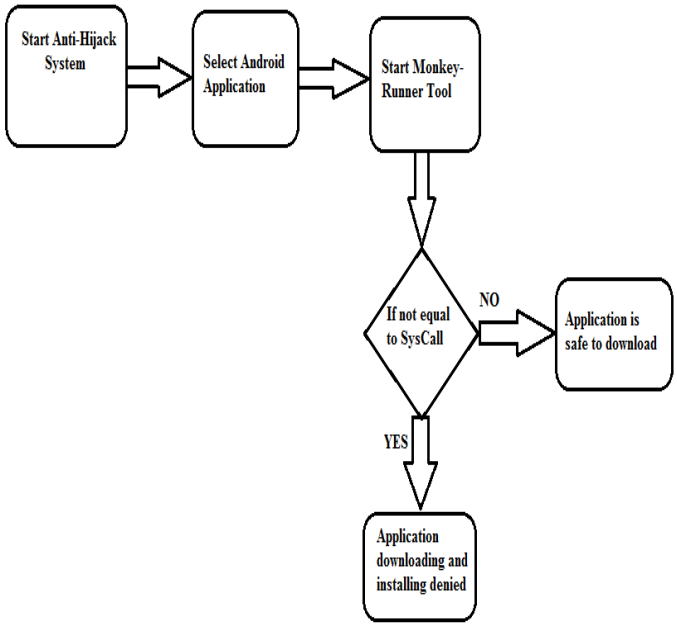

Figure 1: System Architecture (1)

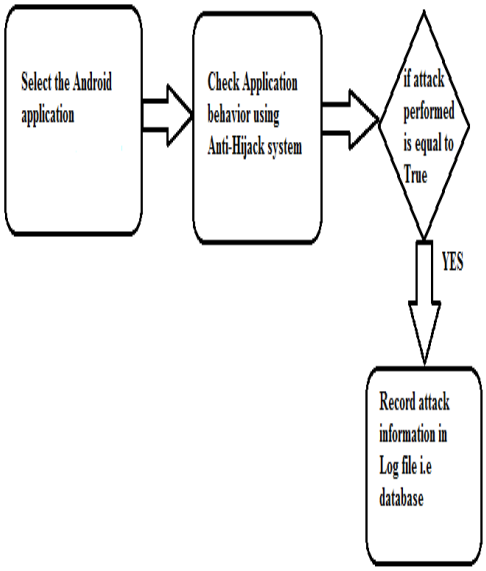

Figure 2: System Architecture (2)

1.1 Anti-Hijack System: This system contain harmful permission and attack dataset. Initially this system must be turn on in mobile.

1.2 Monkey-Runner Tool: 1) It is a functional tool for testing the android applications at a time of development phase of their own.

2) This monkey-runner tool is checking the permissions and system calls of the testing application with the benign application which doesn't contain any malicious permission and system call.

3) If monkey-runner tool found any suspicious permission or system call that do not match with benign application then it prevents the downloading of the application.

The figure 2 gives analysis of capturing the attack and saving the attack information into log file which is our database.

Attack information includes name of attack e.g. session attack, intent attack, password hacking etc. as well as date and time when attack is performed.

1.3 Harmful Permissions: Figure 3 have data of some permissions which are called as harmful by Google. These permissions may hack the private data of the user. Hackers use such permissions to steal the private data from mobile.

Permission Group	Permissions
CALENDAR	• READ_CALENDAR • WRITE_CALENDAR
CAMERA	• CAMERA
CONTACTS	• READ_CONTACTS • WRITE_CONTACTS • GET_ACCOUNTS
LOCATION	• ACCESS_FINE_LOCATION • ACCESS_COARSE_LOCATION
MICROPHONE	• RECORD_AUDIO
PHONE	• READ_PHONE_STATE • CALL_PHONE • READ_CALL_LOG • WRITE_CALL_LOG • ADD_VOICEMAIL • USE_SIP • PROCESS_OUTGOING_CALLS
SENSORS	• BODY_SENSORS
SMS	• SEND_SMS • RECEIVE_SMS • READ_SMS • RECEIVE_WAP_PUSH • RECEIVE_MMS
STORAGE	• READ_EXTERNAL_STORAGE • WRITE_EXTERNAL_STORAGE

Figure 3: Dangerous permissions and permission groups.

V. CONCLUSION

By implementing this project there would be improved security and reduce vulnerabilities and malicious activities which are performed on android devices. Proposed approach is a strong estimated framework for exploitation vulnerabilities using real time analysis. Main aim of honey-pot based model is to allow malign apps to do its activities. This information is used for security of benign apps and detection of malicious apps. In this paper, we made two key contributions, one is detecting intent hijacking and second is detecting session hijacking using our framework Anti-Hijack. We would like to implement some more approaches to detect root exploits and scripts which violate sandbox to gain special privileges in next approach. Anti-Hijacking System can detect malicious activities which are performed on android devices. We also introduce the concept of Anti-Hijack, by accepting, introducing as well as protecting our devices from threats so by considering all these feature our application make most user friendly, and also efficient and easy to use.

VI. REFERENCES

1. T. B`Ieasing, L. Batyuk, A.-D. Schmidt, S. A. Camtepe, and S. Albayrak, An android application sandbox system for suspicious software detection, In 5th International Conference on Malicious and Unwanted Software (MALWARE 2010), pages 55–62, Nancy, France, October 2010. IEEE Conference Publications.

2. O. C.C, Oladej, F.A.Benjamin, B.C.Alakiri, and H.A.Olisa, Penetration testing for android Smartphone's, October 2013

3. F. D. Cerbo, A. Girardello, F. Michahelles, and S. Voronkova, Detection of malicious applications on android os, In H. Sako, K. Franke, and S. Saitoh, editors, ICWF, volume 6540 of Lecture Notes in Computer Science, pages 138–149. Springer, 2010, ISBN 978-3-642-19375-0

4. M. Chandramohan and H. B. K. Tan, Detection of mobile malware in the wild, http://www.infoq.com/articles/detection-of-mobile-malware, 2014.

5. E. Chin, A. P. Felt, K. Greenwood, and D. Wagner, Analyzing inter-application communication in android, In Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services, MobiSys '11, pages 239–252.

6. A. Cozzette, Intent spoofing on android, http://blog.palominolabs.com/2013/05/13/android-security/, 2013.

7. 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, OSDI'10, pages 1–6

8. R. Hay, A new vulnerability in the android framework: Fragment injection, URL https://securityintelligence.com, (Online; Last Accessed January 21, 2015).

9. D. W. Kelly Casteel, Owen Derby, Exploiting common intent vulnerabilities in android applications, In MIT CSAIL Computer Systems Security Group, December 2012

10. J. Lessard and G. C. Kessler, Android forensics: Simplifying cell phone examinations, September 2010.

11. S. Mansfield-Devine, Android architecture: attacking the weak points, In Network Security, 2012(10):5–12, 2012