International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 02 | Feb 2020 www.irjet.net p-ISSN: 2395-0072

DETECTING THE ADVERTISEMENT BEHAVIOR IN ANDROID APPLICATIONS

D.Vetriselvi[1], V.Reethiga[2], A.Vasantha[3]

[1]Assistant Professor, Dept. of Computer Science and Engineering, Jeppiaar SRR Engineering College, Chennai.
[2,3]Final Year Student, Dept. of Computer Science and Engineering, Jeppiaar SRR Engineering College, Chennai.

Abstract - *Mobile phones in recent years have many advantages and a few disadvantages in usage. There are many applications such as Social media and games get access of user's permission to use some privileges. Usually when the user install any application they used to grant permission to access all the privileges. Applications installed in our smart-phones have some tie-up with some advertisement company. Application developer allow those advertisements will run in their application based on the advertisement classification . Advertisements can behave maliciously some times. This behavior extracts personal information like user's contact and files and other important information by using the permission access policy of particular applications. This theft may happen without the knowledge of user. The unauthorized advertisements will upload the user's personal information to the desired advertisement server. To overcome this issue, a third party Server is proposed to validate the advertisement before the advertisements get loaded into the applications. This server will check the malicious codes of the advertisements and restrict their access to the user information and pop up the required access permission to the user. The user can either block the advertisement or allow the advertisement to run in their application.*

***Key Words*: Third party server, Advertisement server, Android Phone, Advertisement Provider.**

1. INTRODUCTION

Advertisement industry is the fastest growing industry in this modern world. We used to see to advertisement in our daily life. Advertisement plays a vital role in trade and business to launch their product or service in the market economy. Traders use strategy to advertise their product and commodities. The purpose of advertisement is to communicate with the consumer and to grab their attention and make consumer buy their product or service. Many years ago, advertisement providers advertise their product through various media, such as newspaper, magazine, radio, television. Nowadays Advertisement providers advertise their product through online. Online advertising is the strategy carried out by advertisement providers to make consumer to know the product and provide offers to sell their product or service. Sometimes these advertisements behave maliciously and get their income by loading their advertisement in related applications. In 2019, 15% of advertisement is advertised through newspaper, 25% through radio, 82% through television, 40% through online and 49% through mobile application. Revenue is about 800 dollars.

Mobile application is the software program developed to run in smart phones and tablets. Mobile application fall under various categories such as application related to healthcare, food, automobiles, etc. Applications are developed in manner that when the user installs any application their details will be accessed and stored in developer server. An application such as you Tube will allow unauthorized advertisement and enable the access permission to advertisement agency. Advertisement can also access the user information through their link.

Google play is the application store for android application. Here there are 2.9 million applications are placed. Among these applications 68.8% of applications is provided for free and user can use those applications without getting paid. An application developer sometimes integrate with the advertisement company. To facilitate the integration, service providers offer software development kits or libraries to application developers. An Application developers integrate these libraries with minor efforts, sometimes adding only small amount lines of code. After that, advertisement libraries interface with advertisement network and retrieve the ad content. Advertisement providers render their advertisement to the application developer to get loaded in their related application. An application developer allow those advertisements to get loaded in their application and get paid by accessing the user's personal information. This is advantageous to both the application developer and advertisement provider.

2. RELATED WORKS

Xiaonan Zhu, Jinku Li, Yajin Zhou, and Jianfeng Ma [1] "Adcapsule : Practical Confinement of Advertisements in Android Applications"- IEEE Transaction on Dependable and Secure Computing,March 2018. Nowadays, app developers tend to integrate advertisement libraries (or ad libraries) into their apps to get revenue from ad networks. However, researches have shown that both ad libraries and ad contents could raise serious security and privacy concerns.In this paper, we propose AdCapsule, a user-level solution to practically confine advertisements, including ad libraries and adcontents. Our solution does not need to change the Android framework, nor requires the root privilege, thus can

be readily deployed. Specifically, we propose the permission sandbox, which isolates the permissions used by ad libraries from the host app, and the file sandbox, which separates the file operations of advertisements. The ad library and ad content cannot read or write any file outside this sandbox. We have implemented a prototype of AdCapsule. Our evaluation results indicate that AdCapsule can successfully enforce security policies to block attempts of accessing private information or manipulating files of the host app, and the performance overhead introduced by AdCapsule is low.

Kathy Wain Yee Au, Yi Fan Zhou, Zhen Huang and David Lie [2] "PScout: Analyzing the Android Permission Specification"- Proceedings of the 2012 ACM conference on Computer and communications security,October 2012. Modern smart phone operating systems (OSs) have been developed with a greater emphasis on security and protecting privacy. One of the mechanisms these systems use to protect users is a permission system, which requires developers to declare what sensitive resources their applications will use, has users agree with this request when they install the application and constrains the application to the requested resources during runtime. As these permission systems become more common, questions have risen about their design and implementation. In this paper, we perform an analysis of the permission system of the Android smart phone OS in an attempt to begin answering some of these questions. Because the documentation of Android's permission system is incomplete and because we wanted to be able to analyze several versions of Android, we developed PScout, a tool that extracts the permission specification from the Android OS source code using static analysis. PScout overcomes several challenges, such as scalability due to Android's 3.4 million line code bases, accounting for permission enforcement across processes due to Android's use of IPC, and abstracting Android's diverse permission checking mechanisms into a single primitive for analysis. We use PScout to analyze 4 versions of Android spanning version 2.2 up to the recently released Android 4.0.

Bin Liu, Bin Liuy, Hongxia Jin, Ramesh Govindanz [3] "Efficient Privilege De-Escalation for Ad Libraries in Mobile Apps"- Proceedings of the 13th Annual International Conference on Mobile Systems,Applications,and Services,May 2015. The proliferation of mobile apps is due in part to the advertising ecosystem which enables developers to earn revenue while providing free apps. Ad-supported apps can be developed rapidly with the availability of ad libraries. However, today's ad libraries essentially have access to the same resources as the parent app, and this has caused significant privacy concerns. In this paper, we explore efficient methods to de-escalate privileges for ad libraries where the resource access privileges for ad libraries can be different from that of the app logic. Our system, PEDAL, contains a novel machine classifier for detecting ad libraries even in the presence of obfuscated code, and techniques for automatically instrumenting by tecode to effect privilege de-escalation even in the presence of privilege inheritance. We evaluate PEDAL on a large set of apps from the Google Play store and demonstrate that it has a 98% accuracy in detecting ad libraries and imposes less than 1% runtime overhead on apps.

Yajin Zhouy, Kunal Pately, Lei Wuy, Zhi Wangz, Xuxian Jiang [4] "Hybrid User-level Sandboxing of Third-party Android Applications" -Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security,April 2015. Users of Android phones increasingly entrust personal information to third-party apps. However, recent studies reveal that many apps, even benign ones, could leak sensitive information without user awareness or consent. Previous solutions either require modifying the Android framework thus significantly impairing their practical deployment, or could be easily defeated by malicious apps using a native library. In this paper, we propose App Cage, a system that thoroughly confines the run-time behavior of third-party Android apps without requiring framework modifications or root privilege. AppCage leverages two complimentary user-level sandboxes to interpose and regulate an app's access to sensitive APIs. Specifically, dex sand box hooks into the app's Dalvik virtual machine instance and redirects each sensitive framework API to a proxy which strictly enforces the user-defined policies, and native sandbox leverages software fault isolation to prevent the app's native libraries from directly accessing the protected APIs or subverting the dex sandbox. We have implemented a prototype of App Cage. Our evaluation shows that App Cage can successfully detect and block attempts to leak private information by third-party apps, and the performance overhead caused by App Cage is negligible for apps without native libraries and minor for apps with them.

Mengtao Sun, Gang Tan [5] "NativeGuard: Protecting Android Applications from Third-Party Native Libraries"- Proceedings of the 2014 ACM Conference on Security and privacy in wireless & mobile networks, July 2014.Android applications often include third-party libraries writ- ten in native code. However, current native components are not well managed by Android's security architecture. We present Native Guard, a security framework that isolates native libraries from other components in Android applications. Leveraging the process-based protection in Android, Native Guard isolates native libraries of an Android application into a second application where unnecessary privileges are eliminated. Native Guard requires neither modifications to Android nor access to the source code of an application. It addresses multiple technical issues to support various interfaces that Android provides to the native world. Experimental results demonstrate that our framework works well with a set of real-world applications, and incurs only modest overhead on benchmark programs.

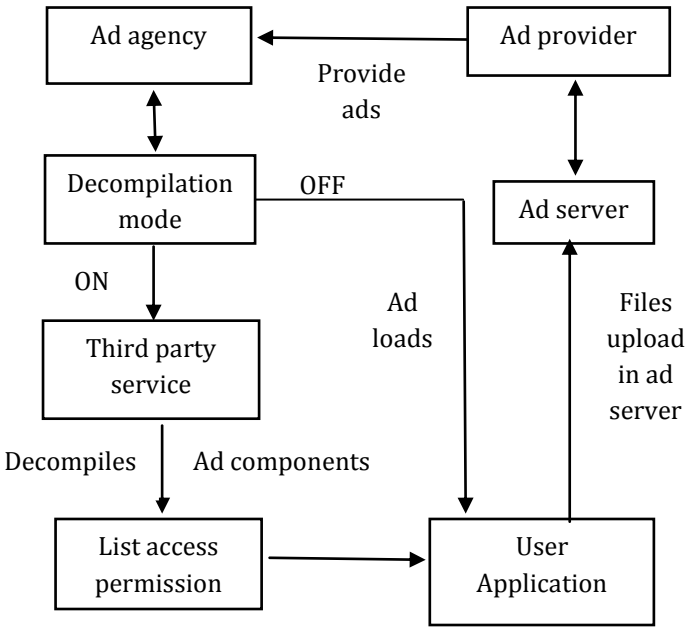
Jaebaek Seo, Daehyeok Kim, Donghyun Cho, Taesoo Kimy, Insik Shin [6] "FLEXDROID: Enforcing In-App Privilege Separation in Android"-Conference on Network and Distributed System Security Symposium ,February 2016. In

this work, we propose FLEXDROID, a new Android security model and isolation mechanism that provides dynamic, fine-grained access control for third-party libraries. With FLEXDROID, application developers not only can gain a full control of third-party libraries (e.g., which permissions to grant or not), but also can specify how to make them behave after detecting a privacy violation (e.g., providing a mock user's information or kill). To achieve such goals, we define a new notion of principals for third-party libraries, and develop a novel security mechanism, called inter-process stack inspection that is effective to JNI as well as dynamic code execution.

3. PROPOSED SYSTEM

A third party Server is proposed to monitor the advertisement library. Here the user whenever install any new application the third party server will start checking the incoming advertisements. So before loading a new advertisement into the application our system will analyze the triggering components inside advertisements and their behavior. If there is any chance of accessing the user details or files or contacts, thus an advertisement can get unauthorized access permissions from the hosted application. Our system is capable of detecting that kind of advertisements and restricts their access, then pop up the required access permission list to the user. Only when the user accepts that permission it allows the ad to run inside the hosted applications. Otherwise, our system will block the background process of the malicious advertisements.

4. PROPOSED ARCHITECTURE



5. METHODOLOGY

5.1. K-MEANS CLUSTERING

K Means Clustering is an unsupervised learning algorithm used to cluster the data based on their similarity. Unsupervised learning means that outcome will not be predictable. In k means clustering, there are specified number of clusters that we want to group into. The algorithm observes data, assign it to the cluster and find the centroid of the cluster. Here this algorithm is used to cluster those advertisements related to their category.

5.2. DECOMPILATION

Decompilation is to convert the object code into executable code, so that the code can be understandable by human. Decompiler is just opposite of compiler which compiles the code in reverse. The purpose of using decompilation is to understand the code, retrieving the source code for analyzing viruses and debugging programs. Decompilation can also be used to retrieve the source code without the permission of the programmer. It also analyzes the variable used in the source code and the code used in the function. Here decompilation is to analyze the source code of the advertisements to prevent against unauthorized access.

6. MODULES AND DESCRIPTION

6.1. ADDING APPLICATIONS IN ADVERTISEMENT AGENCY

Users need to do user registration. After Ad Agency user Authentication the users or application developers will add their applications to the Ad-Agency portal. While adding an application they have to give basic information about the app like the number of downloads, number of daily user logins, ratings, and App ID is mandatory. After that when an Ad-Agency admin sign-in into the portal the requested applications for advertisements will be shown. Here the admin has the privilege to fix amount for the requested applications.

6.2. ADVERTISEMENT PROVIDER ACCESS TO APP BUNDLE

After the Ad Provider, User registered into Ad Agency, The Provider as well as Agency has to generate new bank account using our Bank application. After accounts created successfully they can do money transfer for any desired accounts. Then the Ad Provider has to sign-in the Ad-Agency page and look for applications satisfying their criteria. They can select the applications in a bundle by applying filters based on their type of advertisement content, ratings, cost, number of downloads and number of daily users. Then an

app bundle will be formed with Agency's applied offers. So now the Ad Provider has to purchase the app bundle.

6.3. HOSTING ADVERTISEMENT IN APP BUNDLE:

After bundle purchase the Provider has to upload the suitable ad content which is of types transport, education or health. After uploading the ad content into Agency, it will be automatically loaded into the end applications whose app id is there in the selected bundle. For each category of application we will be showing a related category Android Application. For example, we run a health- related app, Once an advertisement gets hosted into that application the ad content will start initiating its malicious behavior. If the application uses permission like messages or contacts the ad-content will make use of it and send all user's messages or contacts to their ad-Server automatically without user's knowledge in background.

6.4. THIRD-PARTY SERVICE PROVIDER:

We implement a Third Party Service Provider in Ad-Agency side to verify the malicious contents of incoming advertisements. So whenever a new ad component is loaded into the empty spaces allocated for specific ads. The ads get loaded into certain locations inside the host applications. We need to do a preprocessing technique in our proposed Third party Service Provider. So when the advertisement is loaded first into the application the Service provider will check or process all the running components using decompilation and find out the possible permission access that the ads to get from the hosted application. Then the Third party application will send a request to the application user to grant permission for authorized access. A popup will be shown to the application user and get the authorized access. Hereby we can restrict the ad components from access to user's sensitive data. Thus, we can handle the ads from misbehavior.

7. RESULTS

User will receive the list of access permission when the advertisement get loaded in the application. If the user allow the advertisement to run in their application,then the user details will get loaded in the Ad Server, Otherwise third party server will block the back ground process of the advertisements.

8. CONCLUSION

In this work, we propose a third party server to analyze the advertisement libraries, and the advertisement contents. We proposed this to provide safe and security for user's personal information. This prototype will provide the intimation to the user about the access policy list of the advertisements loaded in their mobile application. This third party sever will restrict the access, and also paved the way that the user will know what are the information that the advertisement is going to access. It also makes it easier to support an application in a device without unauthorized advertisements.

REFERENCES

[1] Xiaonan Zhu, Jinku Li, Yajin Zhou, and Jianfeng Ma "Adcapsule : Practical Confinement of Advertisements in Android Applications"-IEEE Transaction on Dependable and Secure Computing,2018.
[2] Kathy Wain Yee Au, Yi Fan Zhou, Zhen Huang and David Lie "PScout: Analyzing the Android Permission Specification"-Proceedings of the 2012 ACM conference on Computer and communications security,October 2012.
[3] Bin Liu, Bin Liuy, Hongxia Jin, Ramesh Govindanz "Efficient Privilege De-Escalation for Ad Libraries in Mobile Apps"-Proceedings of the 13th Annual International Conference on Mobile Systems,Applications,and Services,May 2015.
[4] Yajin Zhouy, Kunal Pately, Lei Wuy, Zhi Wangz, Xuxian Jiang"Hybrid User-level Sandboxing of Third-party Android Applications"-Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security,April 2015.
[5] Mengtao Sun, Gang Tan "NativeGuard: Protecting Android Applications from Third-Party Native Libraries"-Proceedings of the 2014 ACM Conference on Security and privacy in wireless & mobile networks, July 2014.
[6] Jaebaek Seo, Daehyeok Kim, "FLEXDROID: Enforcing In-App Privilege Separation in Android Donghyun Cho, Taesoo Kimy, Insik Shin"-Conference on Network and Distributed System Security Symposium, February 2016.
[7]Shashi Shekhar, Mi chae Dietz, Dan S. Walach "AdSplit:Separating Smartphone Advertising from Applications"-Proceedings of the 21st USENIX conference on Security Symposium,August 2012.
[8] Apostolis Zarras, Alexandros Kapravelos, Gianluca Stringhini "The Dark Alleys of Madison Avenue: Understanding Malicious Advertisements"-Proceedings of the 2014 Conference on Internet Measurement,November 2014.
[9] Michael Grace, Wu Zhou, Xuxian Jiang, Ahmad-Reza Sadeghi "Unsafe Exposure Analysis of Mobile In-App

Advertisements"-Proceedings of the 5th ACM Conference on Security and Privacy in Wireless and Mobile Networks,April 2012.

[10] Rubin Xu, Hassen Saıdi, Ross Anderson "Aurasium: Practical Policy Enforcement for Android Applications"-Proceedings of the 21st USENIX Conference on Security Symposium , August 2012.