Privilege Separation and Isolation of Advertisements in Android Applications

Krishna Sowjanya.k[1], Neha Patwari[2]
[1] *Assistant Professor, MVJCE*
[2] *Assistant Professor, MVJCE*

***Abstract*— Advertising is a critical part of Android system. Many applications use one or more advertising services in their mobiles. These apps may have purchasing cost or be free, but ad-supported. Advertising supported applications request multiple privacy sensitive permissions and also can share same privilege of the host application, which cause major threats like data loss and accessing location of the user. To overcome these problems, i.e; for privilege separation AdDroid is introduced a new advertising APIs to enable separation of advertising functionality from host applications. For isolation of advertisement AFrame, a method to isolate malicious code from host applications. These methods allow applications to show advertisements without any access of private information.**

***keywords*—ad-libraries, sdk, buggy applications, api, gray ware applications, zygote process.**

I. INTRODUCTION

Advertising in mobile system is cooperation between mobile advertising networks and application developers. Mobile advertising such as AdMob and Mellinial Media play a key role by allowing developers to generate revenue from advertisements which are integrated through Software Development Kit (SDK) into the applications. When a user installs the application, the installation process shows the user to choose those permissions which is needed for installation. User cannot differentiate the permissions required for applications and advertisement. Also, the advertisement and the application will have the same privilege, as they are running in the same process which cannot be separated by the system. This leads to over-privileged permissions for applications.

The major threats caused by these over-privileged permissions are:

- Advertisements collect private information of user such as call logs, phone number, and location and may use this information for legitimate purposes.
- An application having bugs may violate user privacy called buggy applications.
- Malicious code may effect permissions of host applications.
- Advertising libraries may be affected by threat because of unsafe wireless networks.

To combat these threats, the proposal is AdDroid an extension of android platform that provides special support for advertisements and AFrame is solution that is used to restrict the privilege of malicious code.

In AdDroid, the host application and advertising code run in separate protection domains. Application developers may integrate advertisements into their application by calling the AdDroid advertising application programming interface (API).

In AFrame, an activity is embedded in another activity. An activity is an application's window that interacts with user. According to the user, these two activities look like one. From the system perspective, they actually run in two different processes with different user IDs. We call such a frame AFrame (Activity Frame).

II. PROPOSAL

To address the above threats, privilege separation can be done in two ways:

2.1 *Privilege Separation within a Process:* Here, every process will have separate privilege permissions but will run on same virtual machine.

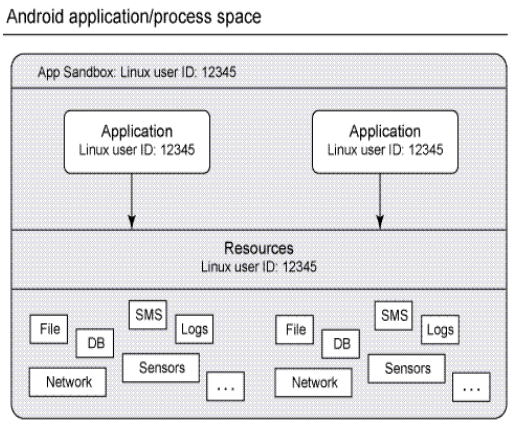

Fig: Privilege seperation within a process.

Both ad-library and applications have their own permissions. Since they run in same virtual machine, ad-libraries can access application permissions and vice versa.

2.2 *Privilege Separation between Processes:* Here, libraries and applications exist as standalone applications. These advertising applications would run in separate processes from the user application.

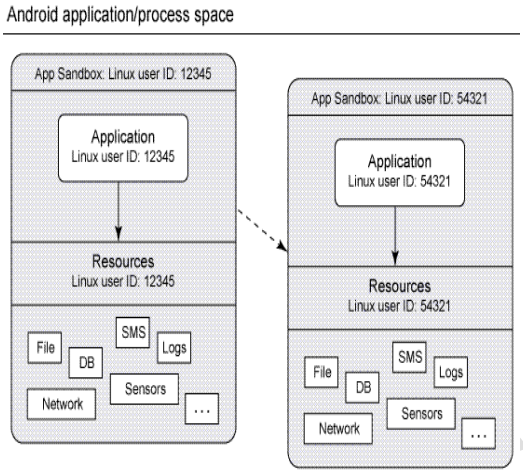

Fig: Two applications with different processes.

Because of this user may uninstall the advertising applications which may cause loss of advertising revenue for developer and companies.

III. IMPLEMENTATION

To achieve the above goals, privilege separation and isolation of process is done in android system as follows:

3.1 PRIVILEGE SEPERATION

AdDroid consists of three parts:

1. A user space library that is part of the Android SDK.
2. A new Android system service.
3. Android permissions.

3.1.1 AdDroid Library API

The AdDroid user space library provides developers a public API, i.e., in writing applications, developers call classes and methods. It supports the insertion of advertisements into applications and communicates data between the application and the AdDroid system. The library includes a new user interface element to display advertisements (an "AdView").

The AdDroid library allows developers to specify which advertising networks they would like to use, and allows use of multiple advertising networks in one application. A separate advertising network can be specified for each AdView, providing flexibility. The AdDroid API is the same for all applications, regardless of which advertising network they use.

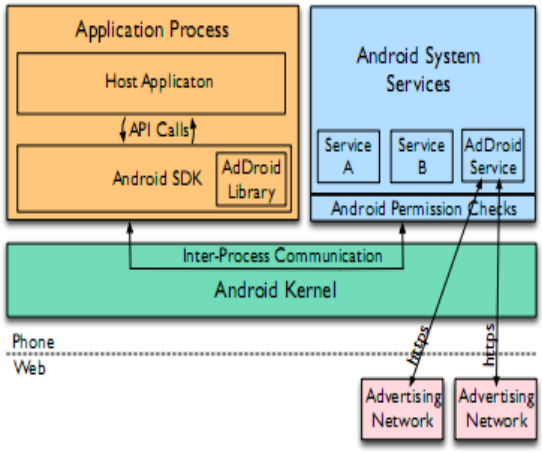

Fig: The AdDroid design

Since the AdDroid library exists in user space, it runs with the host application's permissions. Furthermore, a grayware application could tamper with the user space library. Consequently, the AdDroid library does not perform any privileged operations. Whenever an application requests a new advertisement, the library makes a fetchAd IPC call to the AdDroid system service which in turn performs the necessary privileged operations. Although the AdDroid user space library does not perform any privileged operations, it contains the majority of the advertising functionality.

3.1.2 AdDroid System Service

The AdDroid system service's only job is to receive advertising requests from applications via the AdDroid userspace library and return advertisements. When the AdDroid system ser-vice receives an advertisement request, it establishes a net-work connection to the appropriate advertising network, transmits data to the advertising network, and stores the resulting advertisement. The AdDroid library then makes a follow-up IPC call to the AdDroid system service to retrieve the advertisement.

The data sent to the advertising network during the transaction might include configuration information such as application's customer number, tracking data collected by the application, or advertising context-specific information specified by the application. Some advertising networks might request phone's unique ID (IMEI, MEID OR ESN). But full implementation of AdDroid will supply an alternative ID i.e; ANDROID_ID to advertising networks.

3.1.3 Android Permission Change and code size

AdDroid service verifies its caller's permissions to ensure that ads are fetched only through the AdDroid system service if they have ADVERTISING permissions. This

ADVERTISING permission gives applications to call fetchAds and request advertisements based on data given by application. If it is LOCATION_ADVERTISING, application may request location information to advertisers as well.The below figure shows how the permissions appear to users during installation of AdDroid.

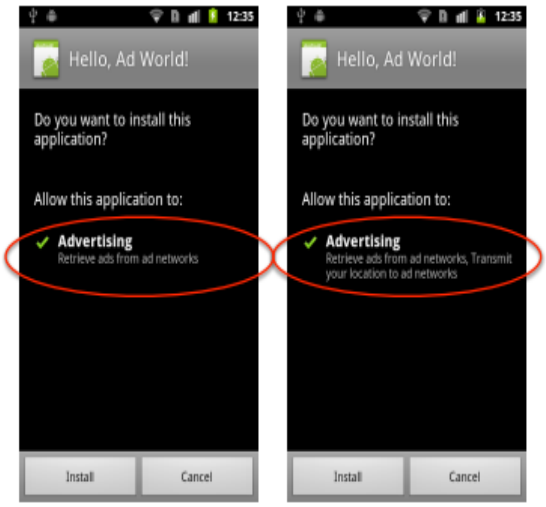

Fig: Installations screens of two applications requesting the new AdDroid permissions.

The implementation of AdDroid needs few modifications to existing Android Open Source Project.

3.2 ISOLATION

Isolation of advertisement from host applications is done with the help of AFrame. AFrame is an activity frame which is embedded in main frame. It is like a view component; it occupies an area in main activity. Inside that a process runs called Aframe Process.

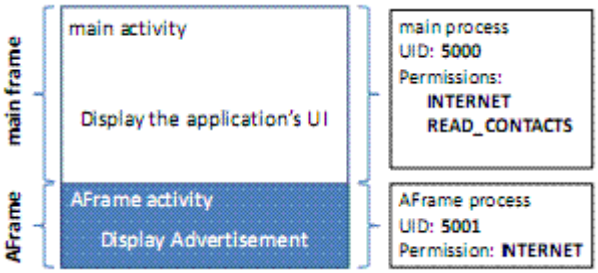

Fig: AFrame

The procedure of isolating advertisement from host applications includes three steps:

1. **Process Isolation**
2. **Permission Isolation**
3. **Display Isolation**

3.2.1 Process Isolation

The goal of process isolation is to separate AFrame activity from main activity by providing different user ID (UID). For this a new process and a new activity for the AFrame region is created.

Package Manager Service (PMS) creates a new user for new application as well as private data folder for its resource usage. When the application is installed, Android checks the manifest file for component information services and content providers. So, a new parsing module in PMS is added in manifest file with tag <aframe> which is used to retrieve AFrame information.

When the application is launched, a process is created by Activity Manager Service (AMS) to run the application. In addition to that AMS also retrieves AFrame information from PMS. It then sends the request to zygote process to create new process to AFrame as well as main frame.

3.2.2 Permission Isolation

At installation time, each application is given a unique user ID (UID) and is associated with its own permissions. At run time, Android uses UID to find out the permissions. Since the UID of main activity and AFrame activity are different, permission isolation happens naturally.

3.2.3 Display Isolation

The AFrame activity and main activity must share the same screen but should be restricted to their own regions.
This can be done in two ways:

1. Soft Isolation
2. Hard Isolation

3.2.3.1Soft Isolation: In this design same buffer memory is mapped to both main process and AFrame process. To restrict their own region standard canvas API is used to draw objects in buffer. This API implements clipping mechanism to make sure that drawing is done only in the region assigned to that process and nothing beyond.

3.2.3.2 Hard Isolation: In this design main process and AFrame process do not share buffer memory. Instead, each process gets a unique buffer and maps that memory to it s own process for drawing. So here memory is totally isolated between the processes.

3.2.3.3 Input Isolation:

Events are generated by user interaction such as clicking, touching and keystrokes. When new activity is started, a request is sent to window manager system service to register a input channel with the system. Window manager forwards request to input manager and establishes input channel with a new activity in z – order. So in AFrame before the events are given to the input channel, UID of the sender process is checked against UID of target process. If this two UIDs are same only then event dispatching is done, if not event will not be dispatched.

IV CONCLUSION

Users of ad-supported applications are vulnerable to grayware, malicious and buggy applications. To overcome privacy and security threats AdDroid and AFrame can be implemented. AdDroid uses privilege separation to isolate privacy sensitive information from applications. Such integration can provide user privacy, security and economic benefits to advertise and developers. With AFrame malicious code can be isolated into a different process with UID.It is also a solution to solve over privileged problem associated with malicious code.

REFERENCES

[1] Mobile Advertising: AdMob http:\\www.admob.com

[2] Theodre book, Adam Pridgen, Dan S Wallach, Rice University. Longitudinal analysis of Android Ad Library Permissions. In arXiv:1303.08572v2 [cs.CR] 18 Apr 2013.

[3] William Enck, Damien Octeau, PatrickMcDaniel, and Swarat chaudhuri. A Study of Android Application Security. In Systems and Internet Infrastructure Security Laboratory, The Pennsylvania Stat University.

[4] Egele, M.Kruegel, C.Kirda, and Vigna. Detecting Privacy Leaks in iOSApplications. in Network and Distributed System Security Symposium 2011.

[5] Michael Grace, Wu Zhou, Xuxian Jiang, Ahmad-Reza Sadeghi: Unsafe Exposure Analysis of Mobile In-App Advertisements. In Center for Advanced Security Research, Technical University Darmstadt, Germany.

[6] Shashi Shekhar, Michael Dietz, Dan S Wallach: AdSplit: Seperating smart phone advertising from applications.

[7] Xiao Zhang, Amit Ahlawat, and Wenliang Du : AFrame: Isolating Advertisements from Mobile Applications in Android. Dept. of Electrical Engineering & Computer Science, Syracuse University, New york, USA.

[8] C.Grier, S.Tang, and S.T.King: Secure web browsing with the OP web browser. In 2008 IEEE symposium on security and privacy,Oakland, May 2008.

[9] Ryan Stevens, Clint Gibler, Jon Crussell, Jeremy Erickson, and Hao Chen: Investigating User Privacy in Android Ad Libraries. University of California, Davis.

[10] Felt, EgelMan, S.Haney, A.Chin and Wagner.D: Android Permissions: User Attention, Comprehension and Behaviour. Tech.Rep. UCB/EECS-2012-26, University of California, Berkely,2012.

[11] Felt, Finifter, M.chin. s.Song and Wagnar: Android Permissions Demystified. In ACM Conference on Computer and Communication Security (CCS), 2011.