Abused Android Permissions by Advertising Networks

Murat Oğul, Selçuk Baktır
Computer Engineering Department
Bahçeşehir University, Istanbul, Turkey

Emin İslam Tatlı
Department of Electrical and Electronics Engineering
Istanbul Medipol University, Istanbul, Turkey

***Abstract*— Android is the leading mobile operating system for smart phone and mobile tablet platforms. Since these mobile devices contain personal and sensitive data, security is a big challenge for them. Even though various security features are supported by Android, its permission model is quite problematic from usability and privacy aspects. When users want to install an application, they must grant all requested permissions. Since manually checking dozens of permissions is cumbersome, users ignore it and accept permissions without reading them. In Google Play Store, there exist thousands of applications that request more permissions than they actually need. Applications with unnecessary permissions can misuse their permissions and endanger their users' security and privacy. Especially, advertising network libraries, integrated within applications, request many unnecessary permissions and get unauthorized access to users' personal data. In this paper, we explain the results of our study which analyzes several advertising networks, their permission requests and behavior for accessing critical resources.**

Keywords—Android permissions, mobile security, Android security, advertising networks

I. Motivation

ANDROID is a Linux-based mobile operating system acquired by Google in 2005. Since then, Android has developed very rapidly, increased its market share and today has dominated the smart phone and mobile tablet platforms. The global market share of Android smartphones in 2013 is 78.4% 0. It is followed by Apples's iOS which has only 15.6% global market share.

Since smart phones are used mostly for personal purposes and for enterprise activities, they store sensitive personal and confidential data. Enforcing security and privacy requirements is therefore inevitable on mobile platforms. Even though Android provides several security features to protect its platform and its mobile users, its permission model has critical design problems. One of these problems is that the permissions of apps and in-app advertising network libraries are not separated. This aspect is misused by several over-privileged advertising networks. They request many unnecessary permissions and use them to get unauthorized access to sensitive data like location, contact list, SMS, etc.

In this paper, we analyze 25 different advertising networks that are integrated in many apps within the official Android store. We analyze extensively their requested permissions and behavior for misusing these permissions.

The paper is organized as follows: Section II explains Android's security and permission model. Section III explains the security risks related to the advertising networks. The details of our analysis are given in Section IV. Section V discusses the related work and Section VI concludes the paper.

II. Android Security and Permissions

Android platform supports several key security features such as OS level security through the Linux kernel, mandatory application sandboxing for applications, secure inter-process communication, application signing, user-granted and application-defined permissions, etc. The common target of all these security features is the protection of user data and system resources and isolation of applications [2]. Most of these security features are inherited from the traditional Linux kernel.

In our study, we focus on abused application permissions. Hence, we start by defining Android's permission model. Permissions define whether a process, user or application is granted permission to trigger a critical activity and access a critical resource. Android gives a unique user and group ID to each Android application. Having a different process ID, an application can access only its own data. This is called application sandboxing. Each application runs in its own sandbox. If an application wants to access system resources or another applications' data, it needs to be first granted the related permission. During installation, each application presents which permissions it needs for execution. Users should accept *all* the requested permissions if they want to install and execute this app. It is not possible to grant only some of the requested permissions. Permissions are defined in the AndroidManifest.xml file, as shown in Figure1. Similar to user-granted permissions, there exist application-granted permissions that define permissions to access data within an app sandbox as illustrated in Figure 2.

```
<uses-permission
android:name="android.permission.READ_PHONE_STATE">
</uses-permission>
<uses-permission
android:name="android.permission.ACCESS_WIFI_STATE">
.......
</uses-feature>
</manifest>
```

Figure1. Permissions in AndroidManifest.xml

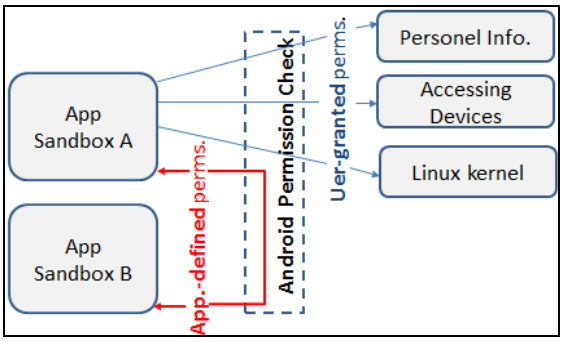

Figure 2. Android's permission control mechanism

Android permissions are categorized into four threat levels as defined below:

- *Normal:* This is the default value. Permissions here are harmless and include low risk to users, system resources and other applications.
- *Dangerous:* This type of permissions can access private user data and take control over devices. For example, RECORD_AUDIO or CAMERA permissions.
- *Signature:* If two applications are signed with the same certificate, they are granted access each other's data.
- *Signature or System:* This is a special purpose category when multiple vendors need to share specific features in the same system.

III. Security Risks of Mobile Advertising Networks

Mobile devices and applications are playing an increasingly more critical role in our lives. People carry them with themselves everywhere they go, store sensitive data on them and also share personal information with others, including exact location information, via apps. This new life style attracts advertising companies who would like to gain financial benefits from mobile users. There exist today several advertising network platforms through which advertisements can be sent to target user groups. As shown in the revenue flow in Figure 3, companies and organizations pay advertising networks for their advertisements. Developers integrate components and libraries of advertising networks in their mobile apps, especially in free apps. Both developers and advertising networks get financial benefit from this life cycle.

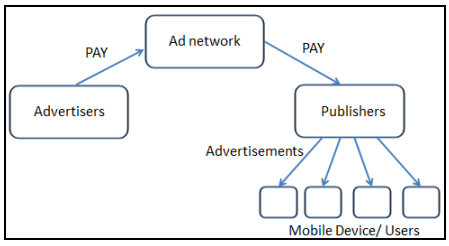

Figure 3. Revenue flow in mobile advertising market

All parties in this revenue and data flow are content except mobile users. Since advertising networks get mostly unauthorized access to mobile users' data, they pose a threat to the security and privacy of mobile users. Advertising network libraries, integrated within mobile apps, request dozens of unnecessary permissions to access critical resources. In general, permission for Internet access is sufficient for advertising network libraries, but they also request permissions to access information such as the location information of the device, SMS messages, contact lists, call details, audio and video functions, external storage data, phone state, etc. They use some of this collected data for customized advertisements. For example, sushi advertisements are sent to mobile users in Japan, whereas pizza advertisements are sent in Italy. It is still unacceptable that GPS coordinates are collected for customization without knowledge of mobile users. Furthermore, access to certain sensitive data (e.g. SMS) for customization is both irrelevant and unacceptable.

IV. The Analysis

All permissions requested by an application are declared in its AndroidManifest.xml file. Permissions required by advertising network libraries are included in the same Manifest file. Users should grant all requested permissions in order to install an application. But they cannot directly distinguish application permissions from permissions of advertising network libraries. Unnecessary permissions are either requested unconsciously by developers or intentionally by advertising network libraries.

In our study we analyzed several advertising networks, their permission request models and how these permissions are used to access critical resources. In order to identify permissions requested by advertising network libraries, we examined several free applications from the official Google Play Store.

We used manual analysis techniques to check permissions within the Manifest files. In the first step, we checked which permissions exist within the AndroidManifest.xml file of a mobile application. For this purpose, the dex2jar utility [11] was used to convert apk packages to java jar files and access Manifest files. Afterwards, we decompiled jar files to access the source codes of the apps. The jar files were decompiled by using the jd-gui tool [13], as shown in Figure 4. Finally, we performed manual code review of advertising network libraries and searched for the relevant method calls which access the existing permissions in the Manifest file.

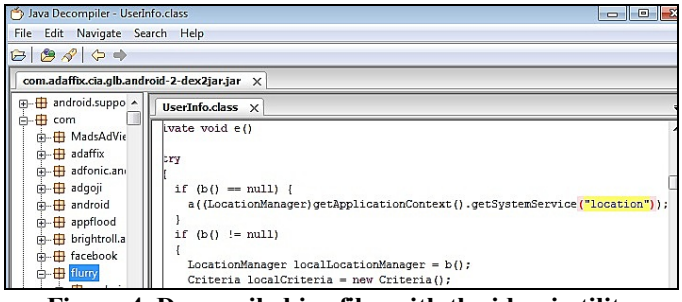

Figure 4. Decompiled jar files with the jd-gui utility

We examined approximately 200 free applications in Google Play Store, and exploited the most relevant apps which

support several different advertising network libraries in our analyses. These apps are Bitstrips Viewer 1.7 [3], Find Viber Friends 0.2 [4], Weather Live 2.4 [5], ensogukespriler 1.2 [6], WetterApp 2.3.3 [7] and CIA 4.0.11 [8]. We installed these apps on Samsung Galaxy S3 (Android 4.3 and API 18) running on a free Android emulator called Genymotion v2.2.2 [9]. We configured the emulator to send the network traffic through the Burp Suit proxy software [10] running on our local machine. By installing Burp's CA certificate to the emulator device, it was possible to inspect both HTTP and encrypted HTTPS traffic generated by either the apps or the advertising network libraries, as shown in Figure 5.

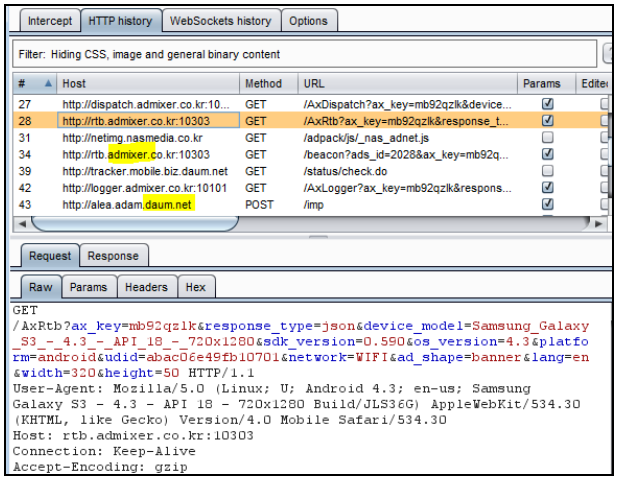

Figure 5. Admixer's HTTP GET request while the Find Viber Friend app is running

While it is running, Find Viber Friend sends critical user location information with exact latitude and longitude values to the Admixer advertising network (see Figure 6). During our traffic analysis, we also realized that some advertising network libraries do not try to establish any HTTP sessions. Either they are not activated or, before execution, they check whether they are installed on a real device or on an emulator.

After converting apk files to jar files, we used the open-source AXMLPrinter2.jar tool [11], which converts Android binary XML files to human-readable XML files, in order to access the Manifest files and defined permissions. We also needed to know which java classes and methods are used to get the Android permissions as defined in the Manifest files.

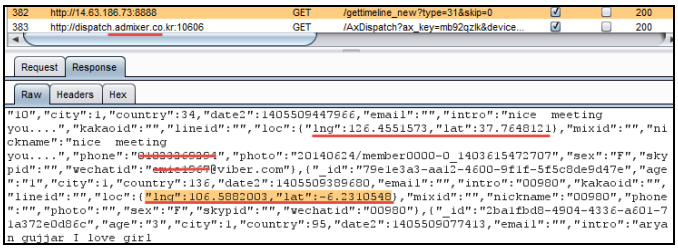

Figure 6. Private information sent via Find Viber Friend

As Vidas et al. [21] explain in detail, the documentation about manifest permissions by Google is still a big challenge. In the official Android documentation, we could not find out about most of the mappings between methods and permissions requested by using these methods. We obtained the majority of the Java methods by searching in Internet and by using the PScout tool [22] which generates API call mappings. Considering the identified permissions in the Manifest files and also the identified mappings between methods and permissions, we checked the source codes of advertising network libraries and searched for the identified methods. For example, if any library includes the *getActiveNetworkInfo* method, we assume that this library uses the ACCESS_NETWORK_STATE permission.

We analyzed 6 different apps and 25 different advertising network libraries that are integrated within these apps. The results of our analysis, listing advertising network libraries and their requested permissions, are shown in Table I. Here, X represents that the related permission is declared in the AndroidmManifest.xml file of the app and Y represents that the relevant permission is requested by the relevant advertising network library.

Our analysis shows several interesting results. Mobile apps mostly contain several advertising network libraries which request different permissions, e.g. the CIA app contains 13 different advertising libraries which request too many unnecessary permissions. They access outgoing calls, access/modify contact lists, read/write to external storage, receive and send SMS, and read phone state, calling function, location, vibration function, read log files, etc. The inMobi advertising network (www.inmobi.com) requests 10 different permissions including recording audio. These are CALL_PHONE, INTERNET, RECORD_AUDIO, ACCESS_WIFI_STATE, ACCESS_FINE_LOCATION, READ_PHONE_STATE, ACCESS_NETWORK_STATE, VIBRATE, READ_LOGS CALL_PHONE and ACCESS_COARSE_LOCATION.

V. Related Work

There are several research studies about Android permission problems and advertising networks. Shekhar et al. [14] point out the unnecessary permissions problem of advertising libraries and propose separation of original applications from their advertising libraries and running the libraries as separate applications. Similar to our analysis, Book et al. [15] analyzed permissions of advertising libraries but they focused on permission changes of the libraries over time. They conclude that more and more permissions are requested by the libraries year by year. Likewise, Stevens et al. [16] investigate 13 popular advertising libraries in terms of Android permissions. But they exploit only documentations and network analysis to identify the requested permissions by the libraries. In our work, we performed additionally manual code review for more accurate results. Grace et al. [17] analyzed ca. 100 advertising libraries by using a self-developed static analysis tool, called AdRisk, which focuses on risks in the context of privacy and

untrusted code downloading and execution. They also show that most existing libraries collect personal private information. Pearce et al. [18] focus on privilege separation and propose a solution, named AdDroid, in which advertising networks do not provide libraries but the Android API is extended to support advertising. Zhang et al. [19] propose a framework called AFrame which isolates untrusted third-party code (e.g. ad libraries) from host applications. Leontiadis et al. [20] conducted an extensive analysis of 250,000 Android apps and identified that the current privacy protection mechanisms of Android are not effective. They propose thus a privacy-aware framework as an alternative advertising service. The framework maintains equilibrium between private information flow and the advertisement revenue.

AD NET LIBRARY	INTERNET	ACCESS_WIFI_STATE	ACCESS_NETWORK_STATE	PROCESS_OUTGOING_CALLS	READ_PHONE_STATE	WRITE_CONTACTS	READ_CONTACTS	CALL_PHONE	READ_EXTERNAL_STORAGE	WRITE_EXTERNAL_STORAGE	ACCESS_COARSE_LOCATION	RECEIVE_BOOT_COMPLETED	GET_ACCOUNTS	VIBRATE	ACCESS_FINE_LOCATION	GET_TASKS	RECORD_AUDIO	RECEIVE_SMS	READ_SMS	SEND_SMS	READ_LOGS	READ_CALL_LOG	WRITE_CALL_LOG	WAKE_LOCK	DISABLE_KEYGUARD	VIBRATE	PROCESS_INCOMING_CALLS	SYSTEM_ALERT_WINDOW	USE_CREDENTIALS	CRYPTO
Bitstrips Viewer 1.7	X	X	X																											
Chartboost	Y	Y	Y																											
StartApp	Y	Y	Y																											
AdMob (Google)	Y		Y																											
com.yumak.ensogukespiriler	X	X	X	X	X		X		X	X		X				X								X	X	X				
AdMob (Google)	Y		Y		Y																									Y
AdSense (Google)	Y		Y		Y																									Y
Appbrain	Y		Y																											
StartApp	Y	Y	Y		Y											Y														
TapContext	Y	Y	Y		Y		Y																							
Find Viber Friend 0.2	X	X	X		X			X	X	X	X		X		X	X	X				X			X		X				
AdMixer	Y		Y		Y					Y																				
Daum	Y	Y	Y		Y			Y		Y	Y				Y		Y				Y									
InMobi	Y	Y	Y		Y			Y			Y			Y	Y		Y				Y									Y
ShallWeAd	Y		Y		Y																									Y
Tad	Y		Y		Y					Y	Y				Y						Y									Y
Weather Live 2.4	X	X	X		X				X	X	X	X			X									X						
Amazon Mobile Ads	Y	Y	Y		Y						Y																			Y
Google Mobile Ads	Y		Y		Y				Y																					Y
Millenial	Y		Y							Y																				
MoPub	Y		Y							Y	Y																			
WetterApp 2.3.3	X		X						X	X	X				X											X				
MOcean	Y								Y	Y	Y				Y															
CIA 4.0.11	X		X	X	X	X	X	X	X	X	X	X	X						X	X	X		X	X			X	X		
AdFalcon	Y		Y	Y	Y			Y			Y																			
AdFonic	Y		Y		Y																									
Appflood	Y		Y	Y	Y				Y	Y	Y																			Y
Brightroll	Y			Y																										
Flurry Ads	Y		Y	Y	Y				Y	Y	Y																			
Google Mobile Ads	Y		Y	Y	Y				Y	Y	Y																			Y
InMobi	Y		Y	Y	Y						Y																			Y
LifeStreet	Y		Y	Y	Y						Y																			
MDotM	Y		Y		Y				Y	Y																				
Millenial	Y		Y	Y	Y				Y	Y																				
MoPub	Y		Y	Y					Y	Y	Y																			
Smaato	Y		Y		Y				Y	Y	Y		Y																	
SmartAdServer	Y		Y	Y						Y																				

Table I. The requested permissions of the analyzed advertising networks

VI. Conclusion

Android is the leading operating system for mobile platforms. Mobile smart phones store and process sensitive personal information including location data. Android platform provides several security features for protecting user privacy, but its permission model has critical design problems. Since permissions of mobile apps and in-app integrated advertising network libraries are not separated by default, this is misused by the advertising network libraries. In this paper, we analyzed 25 different advertising networks, their requested permissions and their behaviors for misusing the requested permissions. Our analysis showed that most of the ad libraries are over-privileged and even access location information and record audio and deliver them to remote servers. It is therefore inevitable to separate privileges of mobile Android apps from advertising networks.

References

[1] Global market share held by smartphone operating systems from 2009 to 2013. Available: http://www.statista.com/statistics/263453/global-market-share-held-by-smartphone-operating-systems/

[2] Android Security Overview, Available: https://source.android.com/devices/tech/security/

[3] Bitstrips Viewer, https://play.google.com/store/apps/details?id=com.bitstrips.viewer

[4] Find Viber Friends, https://play.google.com/store/apps/details?id=com.msgbox.findvibe

[5] Weather Live Free, play.google.com/store/apps/details?id=com.apalon.weatherlive.free

[6] Ensogukespriler, http://www.appbrain.com/app/buz-gi%CC%87bi%CC%87-espiriler/com.yumak.ensogukespiriler

[7] Wetter App, https://play.google.com/store/apps/details?id=de.wetteronline.wetterapp

[8] CIA free caller ID, play.google.com/store/apps/details?id=com.adaffix.cia.glb.android

[9] Genymotion Android Emulator, http://www.genymotion.com/

[10] Burp Suite Proxy, http://portswigger.net/burp/proxy.html

[11] Dex2jar, https://code.google.com/p/dex2jar/

[12] AXMLPrinter2.jar, https://code.google.com/p/android4me/downloads/detail?name=AXMLPrinter2.jar

[13] jd-gui Java Decompiler, http://jd.benow.ca/

[14] Shekhar, S., Dietz, M., and Wallach, D. S. AdSplit: Separating Smartphone Advertising From Applications. In USENIX Security Symposium (SSYM), 2012.

[15] Book, T., Pridgen, A., and Wallach, D. S. Longitudinal Analysis of Android Ad Library Permissions. In IEEE Mobile Security Technologies 2013.

[16] R. Stevens, C. Gibler, J. Crussell, J. Erickson, and H. Chen, "Investigating user privacy in Android ad libraries," in Mobile Security Technologies (MoST) 2012.

[17] M. Grace, W. Zhou, X. Jiang, and A. Sadeghi, "Unsafe exposure analysis of mobile in-app advertisements," in Proceedings of the fifth ACM conference on Security and Privacy in Wireless and Mobile Networks. ACM, 2012, pp. 101–112.

[18] P. Pearce, A. P. Felt, G. Nunez, and D. Wagner. AdDroid: Privilege Separation for Applications and Advertisers in Android. In Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security, 2012.

[19] Xiao Zhang, Amit Ahlawat, Wenliang Du: AFrame: isolating advertisements from mobile applications in Android. ACSAC 2013.

[20] I. Leontiadis, C. Efstratiou, M. Picone, and C. Mascolo, "Don't kill my ads!: Balancing privacy in an ad-supported mobile application market," in Proceedings of the Twelfth Workshop on Mobile Computing Systems & Applications. ACM, 2012, p. 2.

[21] T. Vidas, N. Christin, and L. Cranor. Curbing Android permission creep. In Proceedings of the Web 2.0 Security and Privacy 2011 workshop (W2SP 2011), Oakland, CA, May 2011.

[22] K. Au, Y. Zhou, Z. Huang, and D. Lie, "Pscout: Analyzing the Android permission specification," in Proceedings of the 19th ACM conference on Computer and Communications Security. ACM, 2012, pp. 217–22.