Real Face of Android Permission Modal

Ashmeet Kaur, Divya Upadhyay

Department of Computer Science & Engineering, Amity University, Uttar Pradesh, India.

Abstract---Android Operating System works on a permission based model. This permission based model is used to protect user's data and system resources. Every Android app we install request certain permission from the user. Without the user acceptance of the permission, the app cannot be installed in our device. Asking permission from the user is meant to prevent spreading of the malicious apps among the Android devices. But there are some hidden meanings of the permissions which need to be understood before they are granted to the app.

In this paper, we help the Android user to better understand the Android App Permission model and their hidden meaning, so that the user's can protect their devices from unwanted attacks from the cybercriminal activities. It is useful to understand what each Android app permission mean and what all information the app would be using before granting permission to the app's to access your device.

Index Terms---Android permission, Mobile application, Understanding permission, smartphone privacy.

I. Introduction

The growing sales of Android-based Smartphone's has made Android platform as the dominant mobile Operating System. The reason for the increasing popularity of Android operating system is that everyone looks for a smartphone that has got technologically advanced capabilities than just basic communication and messaging features. Everyone's want to have today more than just a basic phone used for messaging and calling, they want to do as many things as possible with their phones.

The most interesting feature offered by the Android OS to the developers is the ability to build their own apps which are rich and innovative. The freedom provided to the user to personalize their smartphone devices according to their needs, is an attractive feature offered by the Android OS. It helps user to personalize their smartphones and make them as unique as possible by downloading a variety of apps available on the Google Play Store.

This consumer-market allows developers to easily create applications and users to easily install them. The apps available on Google Play are both free and paid. Most of the free apps are downloaded from the Google Play. These free apps are the most dangerous thing to be afraid of. As there could be a motive of the developer of posting the app free of cost on the Play Store.

This consumer-market allows developers to easily create applications and users to easily install them. The apps available on Google Play are both free and paid. Most of the free apps are downloaded from the Google Play. These free apps are the most dangerous thing to be afraid of. As there could be a motive of the developer of posting the app free of cost on the Play Store.

Every app available on the Google Play requires some permission before they can be installed on our devices. The Android security system allows the user to see what all permissions are required by the app but doesn't provide choice to the user to select some of the permissions and granting permission to them and rejecting rest of the permissions. The Android security system doesn't provide access to modify the permissions according to the user of the app. The user has no other option but to permit all the permissions of the apps, if he wants that app to be installed to his system. These permissions are certain set of resources which is required by the developer to access user information. This user information resource could be anything like device IMEI number, user's location, user's contact details, user's messages etc. The user of

the app is not even aware of the fact that his personal information is being accessed by someone else [4]. Lack of information about the Android permissions let the user to put their secure information on stake.

An uneducated user is not able to understand the permissions being asked by the developer through the app and therefore just simply grants the permissions tasked by the developer through the app. Sometimes even an educated person could not make out what harm or what all consequences could be by granting permissions to the developers through the app. The consequences and the hidden meaning of the permissions asked by the developers needs to be understand before granting them.

II. Google Policies

Google defines several developer program policies to maintain a positive experience for everyone using Google play [3]. These policies are defined to maintain a strict restriction on to the content displayed by the apps and the information accessed by the applications. The policies defined by the Google are like rules for the developers which they ought to follow in all the circumstances. Following are some of the policies listed by the Google.

A. Deceptive Behaviour

Content, title, icon, description, or screenshots of any product must not contain false or misleading information. The product or the apps available on Google play should not explicit deceptive behaviour and all the content and information it contains should be true.

B. Personal & Confidential Information

Unauthorized publishing or disclosure of private and confidential information of peoples is not permitted. Google doesn't allow the developers to share the private and confidential data of the users with someone else.

C. Dangerous Products

Content that harm, interferes with the operations of, or accesses in an unauthorized manner are not allowed.

Example like

- Transmission of viruses, Trojan horse, worms, defects, malware that may be vulnerable for the security or harm user device, app, or personal data.
- Apps collecting user's information without their knowledge are prohibited.
- Apps with malicious scripts that capture the password entered are prohibited.
- Apps that lead users to download or install apps from developers which are not registered with Google play are not allowed.
- A Google Play app must only use Google Play update mechanism to modify, replace, or update its own APK binary code. Any other method other than this is not allowed.

D. System Interference

- Home screen shortcut, browser bookmarks, or icons of apps and their ads could not be added on the user's device as a service to third parties of for advertising purposes.
- Sending of email, SMS, or other messages on the behalf of the user without providing the user with the ability to confirm the content and intended recipient.

Besides all these and many of the other policies defined by the Google, hardly few of them are followed properly. Google policies are not being followed by the developers accordingly as they are mentioned. Not following the Google policies by the developers is a crime done by them. Abandoning the policies by the developers indicates vulnerability to the user's private data and it could lead in security breaching by the developer of the app. It indicates about the cybercriminal activities and unwanted attacks which could be done to the user's device.

III. How Android Permission Works

Android app permissions are certain set of resources which are required by the developer to access user's information. Only limited system resources could be accessed by the app. If the app wants to access sensitive API's, the apps needs to declare it in the AndroidManifest.XML file [9]. Sensitive API's include location data, camera function, Bluetooth, phone call, SMS/MMS and network communication. Android Manifest file is an XML file which is stored in every app's root directory. All the permissions required for an Android app need to describe in the Manifest file of the app. whenever a user tries to install an app the App Installer of the Android operation system shows all the permissions declared in the manifest file to user to either accept or reject them. This is shown in fig 1.

The permissions accepted by the user are applied to the app as long as it is installed in the device and the system would not show any notification when sensitive API's are being accessed by the app. If an app attempts to access a protected resource which is not mentioned in the manifest file then system will raise a security exception and would terminate the app to avoid the app from accessing the private data.

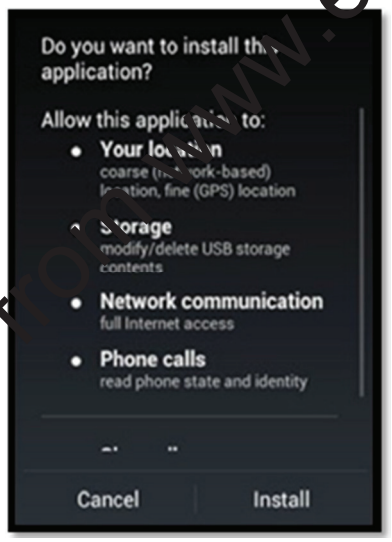

Figure 1: The screenshot of permissions asked by the app

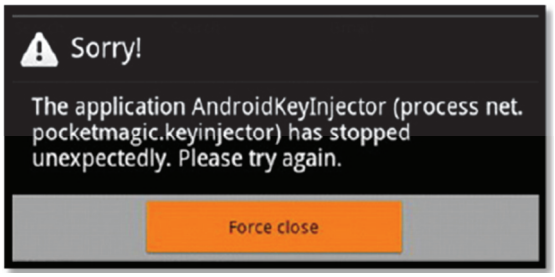

Fig 2: Security exception thrown by the system.

The chances of bypassing the permissions is less but if the permission is defined in the manifest file then the developer can easily befool the system and can access the private data of the user [14]. An example of this could be misusing the browser to upload information. A developer could develop an app to upload stolen data to the desired server by using intent to open a browser. An app can launch any app's content using intent. Figure 3 shows intent having function Intent_ACTION_VIEW that shows that the app wants to view the Google webpage. The Android OS will launch the browser and the malicious app would upload the Device ID to server http://example.com. This example shows how a developer could misuse the permissions mentioned in the manifest file and could do malicious activities without even being noticed by the user or the system.

```
Intent intent = new Intent(Intent.ACTION_VIEW,
Uri.parse("http://example.com/?device_id=xxx");
startActivity(intent);
```

Figure 3: Example of misusing app through intent

IV. Harmfull Permissions

Certain permissions are asked by the developer through app before they are installed in order to work properly in user's device after installation. The apps are dependent upon the permission users grant them, to do what they are supposed to do. Many apps request for network access so that they can download updates. Some apps ask permission to read user's phone state and identity; so calls don't disturb the apps from doing what they are doing [10]. Trojanized apps can alter the permissions to perform malicious actions like record user conversations and send device information like International Mobile Equipment Identity (IMEI) number to a command center.

In 2010, first Android Trojan app was found which sent text messages to certain numbers. Most of the Android apps after that perform one or combination of malicious activities like stealing data i.e. data stealers, allowing remote access i.e. backdoor apps, accessing fraudulent sites i.e. click-fraud enablers, listening to calls and reading personal text messages and contact information i.e. mobile spies, downloading other malwares i.e. downloaders, and gaining root or administrative privileges i.e. rooting enablers [11].

The trojanized apps seek acceptance of the permissions from the user. Once the permissions are granted the Trojan app starts doing attacks. The user not being aware of the outcome; accepts the permissions of the malware app. But there is a lot of difference in what the permissions ask for and what activities do they perform in real [10]. The following list provides information about the permissions asked by the trojanized app. The following list of permissions also tells what all the permissions ought to do and what are they doing in real.

A. Network Communication

Accepting network communication permission means "it allows an application to create network sockets". In real, Network communication is enabled in order to allow apps to access the Internet or Bluetooth-enabled devices. This is the Android permission which is mostly abused because Internet access is required by the malicious app in order to communicate with their command centers or to download updates. Mobile spies and data stealers look for this permission to be granted so that they can steal the information to send it to remote users. Mobile spies send the information about the user to command centers or remote users over internet. Leaving the device Bluetooth discoverable may allow Android malware to infect it like old Symbian OS malware.

B. Storage

The Storage permission 'allows an application to write to the SD card'. Granting an app to modify or delete your Storage Device (SD) card's secure data allows it to read, write and delete anything from the card. In real we are providing authorization to the developers to do anything with our data. The data stealers create a copy of the stolen data from the SD card and send it to the command center. All our private information, photos, audios, videos etc are now accessible to some third person without even our knowledge. What a layman understands by this permission is that the developer would just simply make a folder for his app in the SD card to store data related to his app but there are much more things in real that he do to the card other than this. The malicious app can easily overwrite the existing data on our SD card by this permission.

C. Phone Calls

Phone calls permission 'allows the app to access the phone features of the device. An app with this permission can determine the phone number of this phone, whether a call is active, and the number that call is connected to etc.' Data stealers steal the call log through this app and save them in .TXT file and send it to the command center. The .TXT file is sent over the internet. Phone calls are the favourite target of the data stealers because they provide lots of information. Malicious apps can record your conversation and text messages through this permission. Those users who conduct online banking transactions through their device are at greater risk, as credentials given over device or through messages may land in waiting hands of cybercriminals and then the credentials could be used by the cybercriminals.

D. Hardware Controls

Hardware controls permission 'allows the app to take pictures and videos, record audios, change your audio settings'. Through this permission the developers can take pictures and videos anytime and can send them to the command center or to some third party once your device is connected to the internet. The app could modify the audio setting and could record audio without even user's knowledge. The pictures and audios taken by this app could be used sell to third parties who are ready to buy data related to the user.

E. Messages

The messages permission 'allows reading instant messages, writing instant messages, receiving SMS or MMS, and editing SMS or MMS'. Through this permission the developer could easily sneak into the user device and see all the private and professional messages of the user. Reading someone's messages could give a lot of information about the person. This attack is similar to man in middle attack where the developer can easily read the messages sent and received by the user and can even edit the message. The instant messages sent by the developer without user intervention could cost the user from his balance.

F. System Tools

The System tools 'allows the app change network connectivity, change WLAN state, prevent phone from sleeping, reorder running applications, mount and unmount file Systems for removable storage. It also retrieves information about currently running app'. By abusing this permission, the malware infect the device like, automatically start at boot, Change Wi-Fi state, change network connectivity, and prevent from sleeping, which allows malicious app to run their malicious services. Example a game app doesn't need to start every time our phone boots up so it doesn't need permission to do so. This is a very strong indication that the app silently runs malicious service in the background every time the phone boots up.

G. Personal Information

The Personal Information 'allows the app to read all of the URL's and bookmarks of the browser and can even modify browser history or bookmarks'. Through this permission the cybercriminals can learn about

the browsing pattern of the device user. This mined data could be sold by the data stealers to the cyber criminals and third parties to post ad according and earn profit.

H. Services That Cost Money

Service 'allow app to call phone without user intervention'. The first Android Trojan abused this permission by sending text messages, phone calls and storage permissions. Granting this permission allows sending text messages to premium numbers costing the user a lot of money for the premium services which he don't even used. Through this abused permission, the cybercriminals can pay for certain premium numbers in order to turn profit from every text messages it sends from the infected device.

I. Location

Location 'allows seeing user's current location'. The data stealers sought permission to see where the user is geographically present or located. Information like this can be used to do crimes like stalking. This may be useful when dishing out region-specific spam or malware.

Most of the Android malware sought at least three permissions that are quite unusual for their intended use. These are:

- Full access to internet i.e. Network communication.
- Authority to send text messages i.e. Services that cost money.
- Access to online accounts.

When installing app this is a good indicator of legitimacy. Before giving a green signal to the permissions of the app and granting them, all the users should think carefully.

V. Preventions to be Taken

Android gives us freedom to install any app that we want but the responsibility of keeping the device safe from malware attacks lies in our own hand. Certain measures should be taken to prevent the device from the malicious app [13]. These are:

- All the information of an app should be read before downloading and installing it. The information should be gathered about the creator of the app and reading what other users have to say about this app. Reading the reviews about the help us know the pros and cons of the app. App's store rating should also be checked. Lot of trojanized app lure users to download them but the users should remain alert. Example Fig 4 and Fig 5 show real Angry Bird app and malicious Angry Bird app on Android Market respectively [1].
- Use of mobile security software that protects not only the user's device but also his data should be encouraged. The software's can identify and stop the malwares even before they reach user's device.

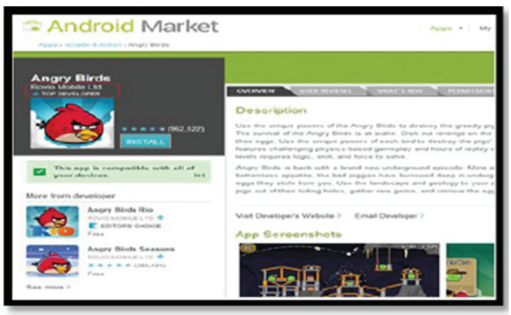

Figure 4: The real Android Market page for Angry Bird app

Figure 5: The malicious Angry Bird app in the Android Market

- The permission an app seeks should be carefully read and understand. Trojanized app happens to seek more permissions than they actually need in order to do work. Asking for too many permission by a legitimate app can put user's device and data in great danger. Example if an app of media player seeks permission to send text messages then it's an indicator that this app might be a malware. The user needs to think before accepting its terms of agreement.

VI. Conclusion

Android market consists of billions of apps. These apps seek certain permissions for the user to grant them before they can be installed on the user's device. This permission granting system in Android is known as Android Permission model. There are various malicious apps present in the Android market who take advantage of the permission modal and the permissions granted by the user. There are certain illegal activities which the malicious apps perform with the help of permission system. The activities which these permissions can lead to are unknown to the common people. The permission can help the trojanized apps to access and steal the private data of the user without the knowledge of the user. The trojanized apps share this users private data with their command centers and other third parties.

This paper deals with Android permission system and the illegal activities performed by these permissions. This paper shows how each and every permission of the Android permission system which seems to be so right, provide a gate to the data stealers to access and steal user's data. Certain preventions must be taken before granting permissions to the malicious apps. An app seeking too many permission than it actually require to do its work can put the user's device in great danger.

References

1. Android App Store. Available at: https://play.google.com/ store/apps.
2. A. R. Beresford, A. Rice, N. Skehin, and R. Sohan. Mockdroid: trading privacy for application functionality on smartphone. In *Proceeding of the 12th Workshop on Mobile Computing Systems and Applications*, HotMobile' 11, pages 49-54, New York, NY, USA, 2011. ACM.
3. Google. Android Reference: Google Play Developer Program policies. Available at: https://play.google.com/ about/developer-contentpolicy.
4. Google. Android Reference: Security. Android Technical Information. Available at: https://source.android.com/tech /security/.
5. Google. Android Reference: Publishing Overview. Available at: https://developer.android.com/guide/ publishing.html.
6. Google. Android Reference: Security and Permissions, 2009. Available at: https://developer.android.com/guide/topics/security/security.html
7. Google. Android basic knowledge. Available at: https://developer.android.com/guide/basics/what-is android.html.

8. Google. Android Reference: Data Storage files. Available at: https://developer.android.com/training/basics/data-storage/files.html.
9. Google. Android Reference: Manifest file. Available at: http://developer.android.com/guide/topics/manifest/manifest-intro.html.
10. M. Nauman, S. Khan, X. Zhang. Apex: extending android permission model and enforcement with user defined runtime constraints. In *Proceeding of the 5th ACM Symposium Information, Computer and Communications Security*, ASIACCS' 10, pages 328-332, New York, NY, USA, 2010. ACM.
11. Trend Micro Newsroom press release 2012. https://www.trendmicro.co.uk/newsroom/pr/the-true-face-of the-android-threat/.
12. Trend Micro Media Security Intelligence. http://www.trendmicro.co.uk/media/misc/when-android-app-wants-more-than-they-need-ebooken.pdf.
13. Trend Micro Media Security Intelligence. http://la.trendmicro/media/misc/secure-your-android-based smartphone-en.pdf.
14. Trend Micro Media Security Intelligence. http://blog.trendmicro.com/trendlabs-security-intelligence/bypassing-android-permissions-whatyou-need-to-know/.
15. W. Enck, P. Gilbert, B. G. Chun, L. P. Cox, J. Jung, P. McDaniel, A. N. Sheth. Taintdroid: an information-flow tracking system for realtime privacy monitoring on smartphone. In *Proceeding of the 9th USENIX conference on Operating systems design and implementation*, OSDI' 10, pages 1-6, Berkeley, CA, USA, 2010. USENIX Association.
16. W. Enck, D. Octeau, P. McDaniel and S. Chaudhuri. A study of android application security. In *Proceeding of the 20th USENIX conference on security*, SEC' 11, pages 21-21, Berkeley, CA, USA, 2011. USENIX.