Detecting Malicious Mobile Applications in Android OS

Tan, Sun Teck Tan, Choon Rui
School of Computing
National University of Singapore
Singapore
email: dcstanst@nus.edu.sg a0087876@u.nus.edu

Abstract— **The use of smartphones has become increasingly popular over the years due to increasing affordability and access to countless useful applications. The Android OS accounts for the majority of the smartphone market share due to its open source nature. This entices many smartphone manufactures to build Android phones. However, its popularity has also made Android OS an attractive target for cybercriminals who develop malicious applications, thereby putting Android mobile users at risk. One of the greatest challenges in protecting mobile users is detecting malicious application among the numerous applications installed on the smartphone. 2,500 mobile applications have been analysed, with 50 free and 50 paid applications taken from each category in the Google Play Store. We observe a distinct correlation between each application's category and its requested permissions, which mean using the pattern of requested permissions. Therefore, using the pattern of requested permissions to detect malicious applications can be an effective method. A filter list can be constructed by further examination on the pattern of the requested permissions. We developed an Android mobile application which uses these filters to scan all the installed applications to detect the presence malicious applications and to flag them for deletion. Additionally, we developed a gamified to cater to non IT-savvy users to use it in a fun and educational manner.**

Keywords- malicious applications; Android OS; Pattern matching.

I. INTRODUCTION

Society is becoming more and more technologically advanced with every passing year. In 2014, 1 out of 5 people in the world possessed a smartphone [8]. The Android Operating System accounts for 84.7% of the worldwide smartphone market share as of the second quarter of 2014 [9]. The popularity of the Android OS makes it an attractive target for cybercriminals. The impact of one malicious Android application will is far reaching, putting more mobile users at risk compared to other OSes.

There has been a 388% rise in malicious applications for the Android market from 2011 to 2013 [7]. Such a vast increase is due to the fact that the majority of mobile users use Android OS, enticing cybercriminals to it. The main Android application marketplace, Google Play, also doesn't enforce a strict control over submitted applications. Although Android devices only allow the installation of signed applications, this measure can be bypassed by simply using a self-signed certificate [1]. Such lenient policy allows cybercriminals to distribute their malicious applications to the public even more easily.

There are many different types of malicious applications. Malicious applications that masquerade as legitimate applications are one of the more prominent mobile threats in 2014 [6]. Here is a typical scenario in which a malicious masquerading application is created. Firstly, the cybercriminal downloads a legitimate application from the Android market. Secondly, the cybercriminal reverse engineers the legitimate application, adds a malicious payload and requests for more permissions to facilitate the attack. The cybercriminal may also update the version number. Lastly, the cybercriminal will repackage the application and publish it back to the public. When a user installs the repackaged application thinking it is the latest version of the legitimate application, the cybercriminal will be able to carry out malicious attacks on the user using the additional permissions granted. Some common attacks include: stealing confidential data, such as SMSes and contact lists using the "READ_SMS" and "READ_CONTACTS" permissions respectively and stealing money by sending SMS messages or making calls to premium rate numbers using the "SEND_SMS" and "CALL_PHONE" permissions respectively. The impact of such malicious applications is very significant as it is up to the creativity of the cybercriminals to make full use of the list of permissions to facilitate their attacks [14].

Mobile users, especially non-IT savvy users, are falling prey to such malicious applications due to their over reliance on the Android market or the reputation or popularity of the applications. Most tech experts recommend that users only download applications from the official Android Store, Google Play because applications from other unknown sources are dangerous [11]. Although this advice is not wrong, it can mislead users, particularly non-IT savvy ones, into thinking that the applications from the official Android Store will always be safe. There have been cases where malicious applications were successfully published to Android Store and Google Play [13]. Therefore users still have to be alert when downloading applications from the Android Store.

A good example of a reputable and popular application is the game "Flappy Bird". Due to its popularity, there have been many malicious applications masquerading as the game "Flappy Bird". Thus, a popular application that has been played by many users does not necessarily equate to an absolutely safe application because there exist malicious repackaged versions of the original application. In fact, users

 ISBN: 978-1-61208-439-8

should be even more vigilant when downloading popular applications as they tend to attract cybercriminals.

Our objective is to provide a solution to non-IT savvy mobile users from falling prey to malicious mobile applications that have been increasing over the past few years. This objective was not fully met by some other existing methods or techniques proposed by other researchers.

For example, the approach of Cerbo et al. [19] is more specific and narrowed down. They only analyzed SMS-related operations done by the Java APIs. What we want to achieve is to detect every category of malicious activities, not just SMS-related problem, e.g. making phone calls to premium numbers or stealing user's personal information. Their approach is effective in detecting any malicious activities arising from SMS-related operations. However there is no scalability and it is outdated as DVM is used in earlier versions of Android devices. For Android version 5.0, an alternate runtime environment "Android Runtime (ART)" has replaced DVM entirely. This makes their solution obsolete. Our approach allows us to have an independent app that will not be affected by any change in the device hardware/software in this situation.

Lei et al. [20] used a permission-based behavioral foot printing scheme and heuristics-based filtering scheme to detect malicious apps. Their scheme takes into account every app with the permissions that can have possible malicious activity. For example, apps that require "SEND_SMS" permission will be prohibited by their scheme. But this will cause problem with messaging app such as WhatsApp. The question is how to decide whether it is a legitimate app requiring "SEND_SMS" permission. We used app's category to handle this problem. Lei's method is much more time-consuming and resource intensive as they scan the application to find how the app behaves, what APIs the app calls and what function parameters are set by the app and so on. Our discovery of using the app's category as part of the detection criteria means our system is as lightweight as possible without the need to do such intensive scans likes their scheme in [20]

Zhou et al [21] classify the apps into high risk, medium risk and low risk using a set of analysis modules. So they can prioritize to put more effort on evaluating the high risk apps. We also classify the apps into high/medium/low risk so that hopefully the user can understand the level of impact and potential damage the app is capable of causing. However, their detection methodology is different from ours. They analyze the app's code signatures and also reverse engineered DVM bytecode. So once again, their method also becomes obsolete. Time and resource might be of concern as they states that it can process 118,318 total apps in less than 4 days. But will the user still be able to use his/her phone with such program running?

The rest of the paper is organized as follow: In Section II, we describe the current situation where the problems occurred and the need to have an application to help the common users. We present an analysis and propose our methodology of solving the problem in Section III. Section IV describes the implementation and it is followed by a brief conclusion in Section V.

II. The Current Situation

This section describes where the problem occurred and the need to have an application to help the common users.

A. *The Human Factor - User Awareness & Knowledge*

The security of a system is only as strong as its weakest link, and all too often, humans are that weakest link. Even if a perfect solution that detects all malicious applications exists, the end result will still be unacceptable if the user does not correctly utilize the solution to protect oneself. Therefore, the user's point of view must be taken into consideration when designing a solution. Contemporary solutions can be too technical and user-unfriendly for use by non-IT savvy users.

The first approach we considered was to have users scrutinize the list of requested permissions. This requires substantial IT knowledge and awareness for them to be able to decide if there are unnecessary permissions requested. Otherwise users may simply install applications even when presented with a long list of unnecessary permissions. A way we can help users, particularly non-IT savvy ones, is to provide guidance. For instance, we can list permissions commonly requested by legitimate applications. Thus, the user will only need to do a basic comparison with the standard.

The second approach of using an antivirus application will be less technically demanding on the user since it will run and identify any malicious applications masquerading as legitimate ones. Nevertheless, due to Android OS sandboxing feature that limits the capabilities of antivirus applications, the user will still be required to manually remove malicious applications identified by the antivirus program from the device. Thus, this approach still requires a small bit of user awareness and knowledge. However, if rooting of device is required, there will be a huge learning curve for non-IT savvy users. Although it is easy to root devices nowadays with just a few button presses, rooted devices can be attacked in many more different ways [2]. Therefore, rooting of device is recommended only for IT-savvy users.

The third approach of having users restrict permissions given to applications will require about the same level of IT knowledge and awareness as the first approach. The user will need to decide which permissions to restrict because a permission that is legitimate in one application might not be legitimate in another application. The same form of assistance provided for the first approach can be used for this approach to help non-IT savvy users. However, modifying permissions granted to other applications on the device will require root access for devices with Android version 4.4.2 or newer. Once again, rooting of device is not recommended for non-IT savvy users, as rooted devices require greater user knowledge and awareness.

 ISBN: 978-1-61208-439-8

B. Deduction and Assumption

The project is focused on Android users, particularly non-IT savvy users, because they are more at risk to falling prey to malicious applications. An ideal solution is to create a security application that detects malicious applications based on permissions requested by applications being installed on devices. The application should be built with user-friendliness as a priority to help non-IT savvy users. The application should not require a rooted device.

A possible way is to create a blacklist of potentially dangerous permissions, such as the "SEND_SMS" permission, which when granted allows the application to send SMS messages to arbitrary recipients, including premium rate numbers. When an application is detected requesting any of the permissions in the blacklist, it will raise an alert and label the application as dangerous. However, there are situations where the "SEND_SMS" permission is not dangerous. Messaging applications will require the "SEND_SMS" permission in order to function. We need to be able to determine when requested permissions are legitimate and when they are malicious. In the next section, we describe the methodology used to answer this question.

III. ANALYSIS

This section presents an analysis and proposes our methodology of solving the problem.

A. Sampling of mobile applications

In order to create a security application that detects malicious applications based on requested permissions, we conducted an analysis on mobile applications' requested permissions. This allowed us to gain a deeper understanding of which permissions are commonly requested by applications. There are a total of 25 categories of applications in the Android Market, Google Play Store. They are "Books & Reference", "Business", "Comics", "Communication", "Education", "Entertainment", "Finance", "Games", "Health & Fitness", "Libraries & Demo", "Lifestyle", "Media & Video", "Medical", "Music & Audio", "News & Magazines", "Personalization", "Photography", "Productivity", "Shopping", "Social", "Sports", "Tools", "Transportation", "Travel & Local" and "Weather". Each category is split between free and paid applications. Therefore to ensure our analysis covers all cases, the requested permissions of 50 free and 50 paid applications of each category were collected. In summary, the total sample size in our study was 2,500 applications ((50+50)*25). There are 261 different permissions at the time of writing and they are divided among 14 permission groups, "In-app purchases", "Device & app history", "Cellular data settings", "Identity", "Contacts/Calendar", "Location", "SMS", "Phone", "Photo/Media/File", "Camera/Microphone", "Wi-Fi connection", "Bluetooth connection", "Device ID & Call info" and "Other".

The retrieved permissions are consolidated into a table with the respective groupings for each category as shown in Table I. The maximum count is 50 as we considered 50 applications in each category.

As shown in Table I, the common permissions for an application in the "Books & Reference" category are "Read the contents of your USB storage", "Modify or delete the contents of your USB storage" from the "Photo/Media/File" group and "Full network access", "View network connections", "Prevent device from sleeping" the from "Other" group.

The two requested permissions in the "Photo/ Media/ File" group allow the application to read and save data such as books and references to the phone. "Full network access" and "View network connections" permissions allow the application to access the Internet to browse and retrieve books and references. "Prevent device from sleeping" permission prevents the device screen from dimming or turning off due to inactivity on the screen because the user might be reading without touching the screen for some time. Thus, these requested permissions are reasonable and legitimate for a "Books & Reference" application.

An application will be highly suspicious if it requests permissions with zero counts or permissions that do not exist in Table I. Note that the "Other" permission group contains more than a hundred permissions. For brevity, we omitted permissions in this category that were not requested by any app.

After analyzing all 25 tables from their respective categories, we concluded that the most commonly requested permissions across all categories are "Read the contents of your USB storage", "Modify or delete the contents of your USB storage" from "Photo/Media/File" group and "Full network access", "View network connections" from the "Other" group. This is because most applications require Internet access and read/write access to the device storage to save data onto the phone.

We produced a bar graph for the data in each table by plotting the permission count against the different permission groups with bars representing the permissions requested by free and paid applications. The graph gives a visual representation that allows us to observe any difference between free and paid applications of each category as shown in Fig. 1. Once again, the maximum count is 50 for each version.

From Fig. 1, we observe that the pattern of requested permissions for free applications closely resembles the pattern of requested permissions for paid applications.

After analyzing all 25 bar graphs from their respective categories, we conclude that the permissions requested for both free and paid applications from the same category generally have the same pattern. However, we observed some notable variations.

 ISBN: 978-1-61208-439-8

TABLE I. CONSOLIDATED TABLE OF TOP 50 FREE AND PAID "BOOKS & REFERENCE" APPS

Permissions group	Individual permission	Requested count (Free App)	Requested count by groups (Free App)	Requested count (Paid App)	Requested count by groups (Paid App)
In-app purchases	Ask to make purchases	12	12	12	12
Device & app history	Retrieve running apps	6	6	8	8
	Read sensitive log data	4		2	
	Read your web bookmarks and history	0		0	
	Retrieve system internal state	0		0	
Cellular data settings	Change/Intercept network settings and traffic	0	0	0	0
Identity	Find accounts on the device	10	10	8	8
	Add or remove accounts	2		0	
	Read your own contact card	0		0	
	Modify your own contact card	0		0	
Contacts/ Calendar	Read your contacts	2	2	0	0
	Modify your contacts	0		0	
	Read calendar events plus confidential information	0		0	
	Add or modify calendar events and send email to guests without owners' knowledge	0		0	
Location	Approximate location (network-based)	8	8	8	10
	Precise location (GPS and network-based)	6		10	
	Access extra location provider commands	0		0	
SMS	Send SMS messages; this may cost you money	0	0	0	0
	Receive text messages (SMS)	0		0	
	Read your text messages (SMS or MMS)	0		0	
	Receive text messages (MMS, picture or video message)	0		0	
	Edit your text messages (SMS or MMS)	0		0	
	Receive text messages (WAP)	0		0	
Phone	Read call log	2	2	0	2
	Directly call phone numbers; this may cost you money	0		2	
	Reroute outgoing calls	0		0	
	Write call log	0		0	
	Modify phone state	0		0	
	Make calls without your intervention	0		0	
Photo/ Media/File	Read the contents of your USB storage	40	40	44	44
	Modify or delete the contents of your USB storage	40		42	
	Access USB storage filesystem	0		0	
	Format external storage	0		0	
	Mount or unmount external storage	0		0	
Camera/ Microphone	Take pictures and videos	4	4	2	2
	Record audio	0		0	
	Record video	0		0	
Wi-Fi connection	View Wi-Fi connections and names of connected devices	14	14	14	14
Bluetooth connection	Can control Bluetooth on your device, and broadcast to or get information of nearby devices	0	0	0	0
Device ID & Call info	Read phone status and identity	24	24	18	18
Other	Full network access	46	46	48	48
	View network connections	44		46	
	Prevent device from sleeping	26		14	
	Receive data from Internet	12		6	
	Control vibration	10		12	
	Run at startup	8		2	
	Google Play license check	6		20	
	Modify system settings	6		2	
	Install shortcuts	6		2	
	Uninstall shortcuts	4		0	
	Read Google service configuration	4		0	
	Send sticky broadcast	4		0	
	Control system backup and restore	2		2	
	Create accounts and set passwords	2		0	
	Use accounts on the device	2		0	
	Toggle sync on and off	2		0	
	Draw over other apps	0		2	
	Connect and disconnect from Wi-Fi	0		2	
	Bind to an accessibility service	0		2	
	Allow Wi-Fi Multicast reception	0		2	
	Set wallpaper	0		2	

.

 ISBN: 978-1-61208-439-8

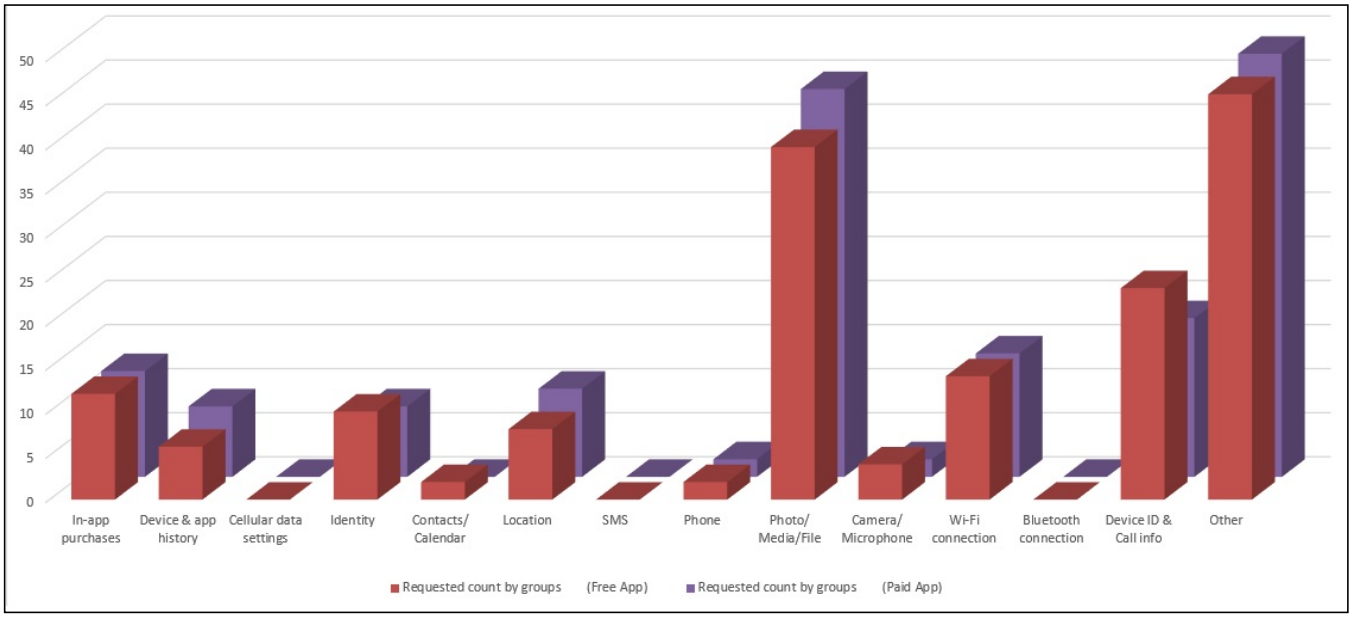

Figure 1. Top 50 free and 50 paid "Books & Reference" Apps

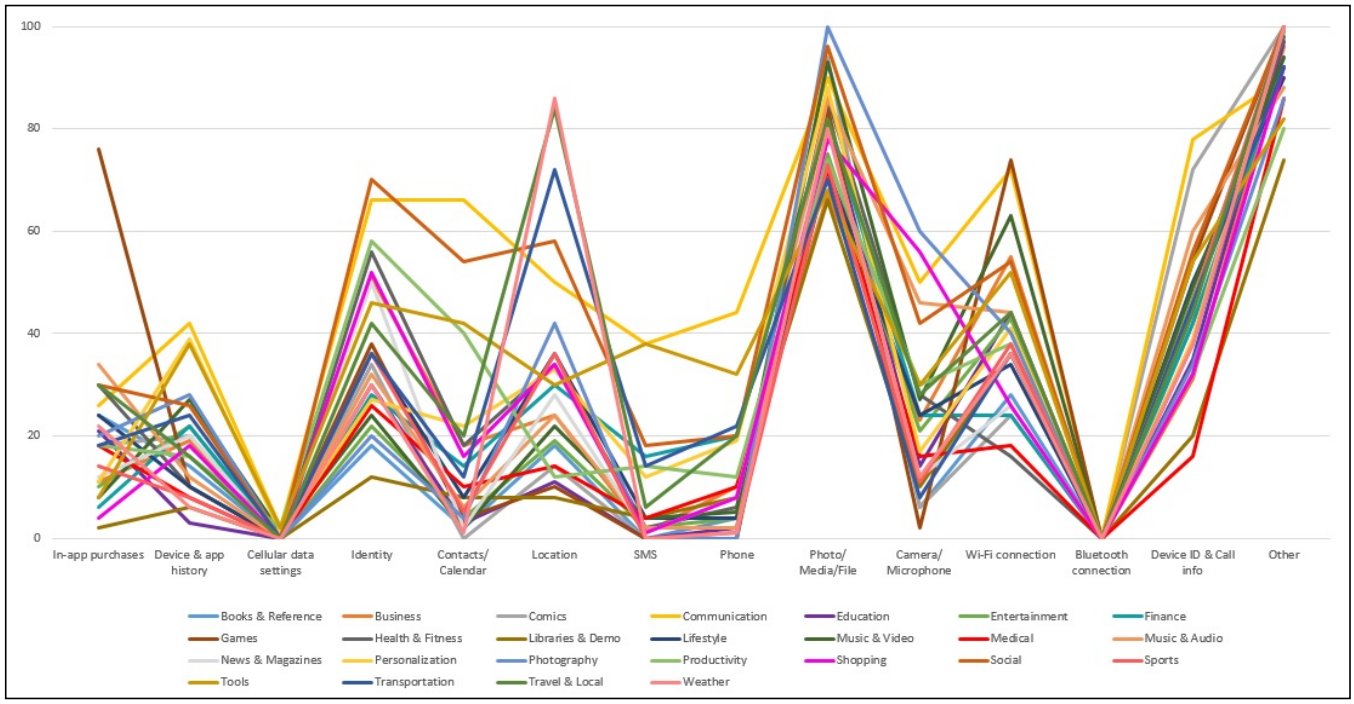

Figure 2. Categories of permissions

The "Google Play license check" permission appeared more in paid applications than in free applications, as paid applications need this permission to check if the user has made any payment. Only a small number of free applications made requests for this permission.

Free applications tend to embed advertisements as a source of income for developers. Thus, additional permissions are required to facilitate the usage of advertisements in the free applications.

Paid applications developed by commercial companies or professionals tend to better understand the concept of permissions and thus request permissions wisely, which lead to fewer requested permissions. A novice developer may request redundant permissions due to uncertainty over the

 ISBN: 978-1-61208-439-8

necessity of various permissions and we observed this in several free applications.

B. *Correlation between each application's category and permissions*

With the required data on the permissions of different categories gathered, we can then attempt to find differences in permissions requested by applications in different categories. As the data collected comprises of both free and paid applications, they will be added and used together in our subsequent analysis. We plot permission counts against permission groups with each line color representing a category of permissions in Fig. 2. The maximum count is 100 because we've summed up counts for both free and paid applications

It is clear that permissions in both the "Photo/Media/File" and "Other" categories are commonly requested for all 25 categories of applications. This result is in line with the conclusion we drew in the previous section, where we observed that the most commonly requested permissions across all categories are "Read the contents of your USB storage", "Modify or delete the contents of your USB storage" from "Photo/Media/File" group and "Full network access", "View network connections" from the "Other" group. It can be observed that there are close to zero permission count for all 25 categories for "Cellular data settings" and "Bluetooth connection" groups, which is due to a change in the permission policy by Android. These permissions have been reassigned - both "BLUETOOTH" and "BLUETOOTH_ADMIN" permissions are now under the "Other" group.

Apart from the points above, each of the 25 categories has a distinct pattern of requested permissions as displayed by each line pattern in the line graph.

Recall that the "Other" group encompasses over a hundred permissions. Thus to further show the different patterns between each of the 25 categories, a deeper analysis is needed. We examine the requested permissions of 25 categories of applications for the "Other" group, and we think this will yield useful results. A line graph is created by plotting permission counts against the different permission groups with each line color representing the permissions from each category of applications as shown in Fig. 3. Once again, the maximum count is 100 due to aggregation over free and paid applications.

All the lines are very high on the left side because the first two permissions are "Full network access" and "View network connections", which are commonly requested across all categories: this result has further reinforced the observation. Different categories have different peaks and patterns in the graph. From both line graphs, we conclude that there is a unique pattern of requested permissions for each of the 25 categories. Amongst permissions, there are no two lines that overlap each other exactly. Thus, each pattern can be used to identify a particular category. Finally, the problem raised previously in this section on how to determine when permission is legitimate or malicious can now be solved**.** Each category has a particular pattern, so the pattern can be used to determine if the permission is malicious or not in that context. Additionally, utilizing these patterns will ensure a better detection rate and also fewer false positives compared to using one general filter, such as the general blacklist method, for every application.

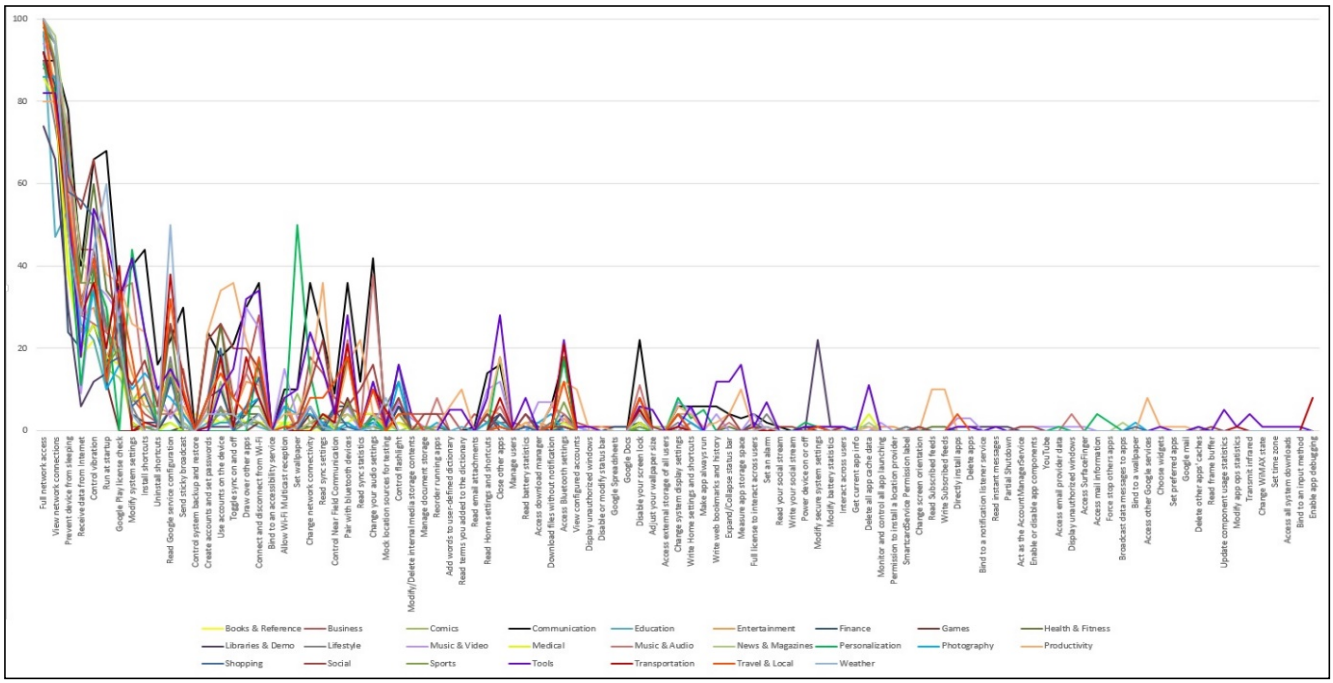

Figure 3. 25 Categories of permissions for "Other" group only

TABLE II. THREAT LEVEL TABLE FOR PERSONALIZATION CATEGORY

Threat Level	Permissions group	Individuals permission	Purpose
Safe	In-app purchases	Ask to make purchases	To allow in-app purchases
	Location	Approximate location (network-based)	To access approximate location derived from network location sources such as cell towers and Wi-Fi
		Precise location (GPS and network-based)	To access precise location from location sources such as GPS, cell towers, and Wi-Fi
	Photo/ Media/File	Test access to protected storage	To read from external storage, such as SD card
		Modify or delete the contents of your USB storage	To write/delete to external storage, such as SD card
	Wi-Fi connection	View Wi-Fi connections and names of connected devices	To check the state of connection before accessing the internet
	Device ID & Call info	Read phone status and identity	To read the phone state by accessing the device identifiers (to know if a call is in progress)
	Other	Full network access	To open network sockets to access the Internet
		View network connections	To access information about networks to check the state of network before connecting to the Internet
		Control vibration	To control the vibrate function of the phone
		Prevent device from sleeping	To keep device and screen active without requiring the user to tap it every minute
		Modify system settings	To read or write the system settings which are common for personalization applications
		Run at startup	To run the application every time upon the phone's startup which is required by some personalized launcher
		Set wallpaper	To personalize the wallpaper
		Install shortcuts	To install shortcuts in homescreen
		Close other apps	To kill the background process of other apps (use to kill apps that cause conflicts when personalizing)
		Connect and disconnect from Wi-Fi	To change Wi-Fi connectivity state
		Access Bluetooth settings	To discover and pair bluetooth devices
		Write Home settings and shortcuts	To write settings and shortcuts in homescreen
		Make app always run	To make the app's activities always active
Mild	Device & app history	Retrieve running apps	To get information about the currently or recently running tasks which will reveal what apps are running on the device
		Read sensitive log data	To read the low-level system log files which include the log files of other applications and might contain sensitive and personal data
		Read your web bookmarks and history	To read the user's browsing history and bookmarks
	Identity	Find accounts on the device	To access the list of accounts in the Accounts Service to choose for use with the app for authentication purposes
		Read your own contact card	To read the user's personal profile data to use as default values or profile picture for some apps
	Contacts/ Calendar	Read your contacts	To read the user's contact lists
		Read calendar events plus confidential information	To read the user's calendar information
	SMS	Read your text messages (SMS or MMS)	To read SMS messages for facilitating the checking of special codes sent by the app to the device
	Phone	Read call log	To read the user's call log, the permission is implicitly granted by "Read your contacts"
	Camera/ Microphone	Take pictures and videos	To access the camera of the device which can be use to take photos to use as wallpaper for personalization
		Record audio	To record audio which can be use in voice search functions provided by some personalized interface
	Other	Change system display settings	To modify the current configuration such as locale
		Receive data from Internet	To accept messages sent by the app's service
		Change network connectivitiy	To change network connectivity state
		Access mail information	To access email information which can be used by personalized notification apps that require email notification
		Change your audio settings	To change the phone audio settings
Danger	Device & app history	Retrieve system internal state	To retrieve the phone internal state dump information from system services. Not common to be requested for "Personalization" apps
	Cellular data settings	Change/Intercept network settings and traffic	To change network settings and to intercept and inspect all network traffic which can potentially monitor, redirect or modify any network packets. Not common to be requested for "Personalization" apps
	Identity	Modify your own contact card	To modify the user's profile. Not common to be requested for "Personalization" apps
		Add or remove accounts	To manage the list of accounts in AccountManager. Not common to be requested for "Personalization" apps
	Contacts/ Calendar	Modify your contacts	To write to user's contact lists but not common to be requested for "Personalization" apps
		Add or modify calendar events and send email to guests without owners' knowledge	To write to user's calendar information but not common to be requested for "Personalization" apps
	Location	Access extra location provider commands	Not common to be requested for "Personalization" apps
	SMS	Edit your text messages (SMS or MMS)	To write SMS messages. Not common to be requested for "Personalization" apps
		Receive text messages (SMS)	To monitor incoming SMS messages which can be recorded or being modified by the app. Not common to be requested for "Personalization" apps
		Receive text messages (MMS, picture or video message)	To monitor incoming MMS messages which can be recorded or being modified by the app. Not common to be requested for "Personalization" apps
		Send SMS messages; this may cost you money	To send an SMS without the user knowing. Not common to be requested for "Personalization" apps
		Receive text messages (WAP)	To monitor incoming WAP push messages which are used by MMS. Not common to be requested for "Personalization" apps
	Phone	Write call log	To modify phone's incoming and outgoing call log which can be used to hide unauthorized calls made. Not common to be requested for "Personalization" apps
		Reroute outgoing calls	To monitor, modify, or drop outgoing calls. Not common to be requested for "Personalization" apps
		Modify phone state	Modify the status of phone functionality which can be used to intercept incoming calls. Not common to be requested for "Personalization" apps
		Directly call phone numbers; this may cost you money	To make calls without user's knowledge or approval. Not common to be requested for "Personalization" apps
	Camera/ Microphone	Record video	To record video. Not common to be requested for "Personalization" apps
	Other	Modify secure system settings	To modify the secure system settings which should only be used by system apps. Not common to be requested for "Personalization" apps
		Access email provider data	To access your email database, including inbox, sent messages, usernames and passwords. Not common to be requested for "Personalization" apps
		Force stop others apps	To force terminate other apps which should only be used by system apps. Can be misused to stop security apps. Not common to be requested for "Personalization" apps
		Download files without notification	To download files without showing any notification. Not common to be requested for "Personalization" apps
		Power device on or off	To power the device on or off. Not common to be requested for "Personalization" apps

If an application requests a permission where its category's line in the figure peaks, this request will be deemed legitimate. If the application requests a permission where the line is lowest in the figure then the application is highly suspicious as this is abnormal behavior.

.

C. *Threat level filter*

Each pattern will be used as the baseline for its respective category and we tailor a threat level filter specifically for that category. There are 3 threat levels for the filter: "Safe", "Mild" and "Danger". A review for each of the permissions found in the respective patterns is performed to further allocate them to the appropriate threat levels.

Permissions in the "Safe" threat level are those that are required to carry out an application's intended core functionalities. For instance, a messaging application will not be flagged as potentially malicious for requesting the "SEND_SMS" permission.

Permissions in the "Mild" threat level are those that may not be necessary for the application's primary functionalities and may raise privacy issues, such as retrieving information about the user and device. However, the potential for malicious activity is still low. An example in this category is an application that retrieves information for an embedded third party advertisement service.

Permissions in the "Danger" threat level are those that can cause some form of damage/loss to the phone or/and the user and are not required for the core functionality of the application. For example, the "CALL_PHONE" permission allows an application to make phone calls without user intervention. An application not in the "Communication" category that requests this permission may be stealing money by calling premium-rate numbers. Permissions that are abnormal, such as those with zero count or those not in Table I above**,** are also in the "Danger" threat level by default as they indicate malicious activity.

The permissions with their respective threat levels are then collated into Table II. As mentioned, any permission not indicated inside Table I is by default in the "Danger" threat level.

When an application is being scanned for malicious intent, the threat level table of the respective application's category is used. The scanner will look up each of the application's requested permissions and check the corresponding mapped threat level in Table I. If there is a deviation from the accepted norm, an alert will be triggered, asking for remedial action, such as the removal of the potentially malicious application.

In the next section, we describe the design and implementation of our proposed security application, DrShield.

IV. DESIGN AND IMPLEMENTATION OF DRSHIELD

This section presents the design and implementation of Dr.Shield.

A. *Application Overview*

The proposed solution is called DrShield. Since Android users are the target audience, the solution is an Android application, which can be installed on the user's phone. The user can then run DrShield, which will scan for malicious applications installed on the phone. For each application classified as potentially malicious, DrShield will highlight abnormal and dangerous permissions requested, along with guidance and recommended remedial action. Upon the user's approval, DrShield will help to delete the detected application from the user's phone

As DrShield is an Android application, it follows the Android structure of bundling Java classes and XML files. The Java classes are used to define the application logic and the XML files are for designing the interface layout. Fig. 4 shows the overall layout of the Java classes created for DrShield. DrShield can be run in one of two modes."Utility Mode" and "Story Mode". The "Utility Mode" offers detection and removal of malicious applications installed on the phone device. "Story Mode" offers the same functionalities of "Utility Mode" but it is repackaged with gaming elements to give users a more fun and educative experience when using the application. Thus the "Utility Mode" is catered towards veteran users who want to get the job done, whereas the "Story Mode" caters to the younger crowd or users who are are less tech-savvy. The "Story Mode" also entices users to know more and raise awareness of the dangers of requested permissions. DrShield will first scan all the installed applications on the phone to detect

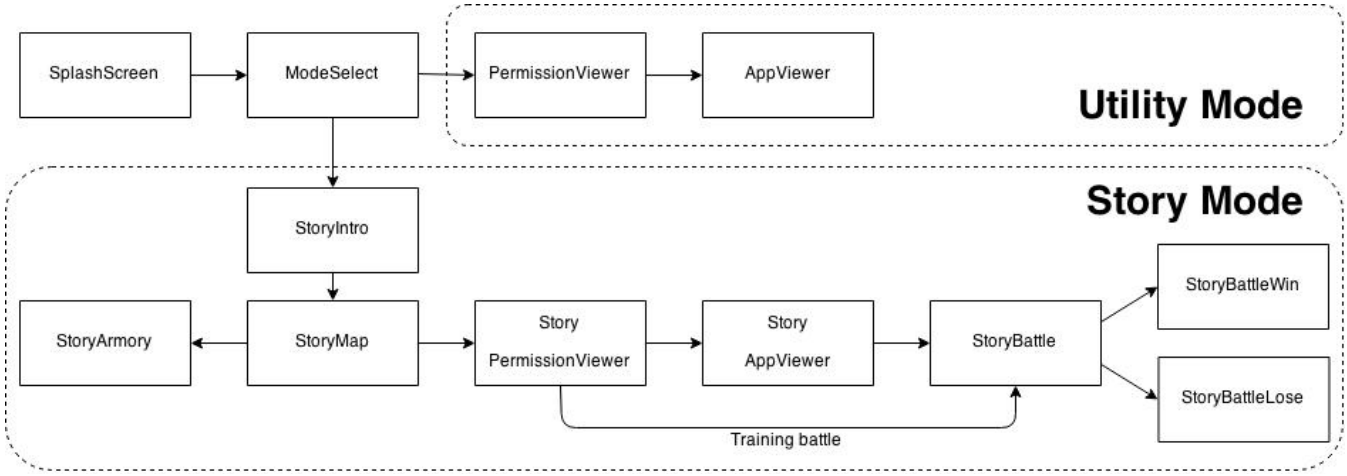

Figure 4. Overall layout of the Java classes created for Dr.Shield

malicious applications. Suspicious applications on the phone will be represented by devils in the game as shown in Fig. 5. The user has to battle and defeat the evil devils to save the world. The game will actually delete each malicious application from the phone when the respective devil is defeated.

When there are no malicious applications detected on the phone, there will be no evil devils in the arena screen to battle. Thus to ensure the continuity of the game, there are also training devils at the bottom of the arena screen. The training devils do not represent actual applications on the phone. There are 3 training devils with different difficulty levels - easy, medium and hard. The difficulty of the battle with the evil devil will depend on the threat level of the corresponding application. Tapping on a devil will move the user to the devil details screen. If the tapped devil is an evil devil application with "High" or "Mild" threat level, the devil details screen will be like Fig. 6. If the tapped devil is a good devil, application with "Low" threat level, the screen will be like Fig. 7.

Figure 5. Devil Arena Screen

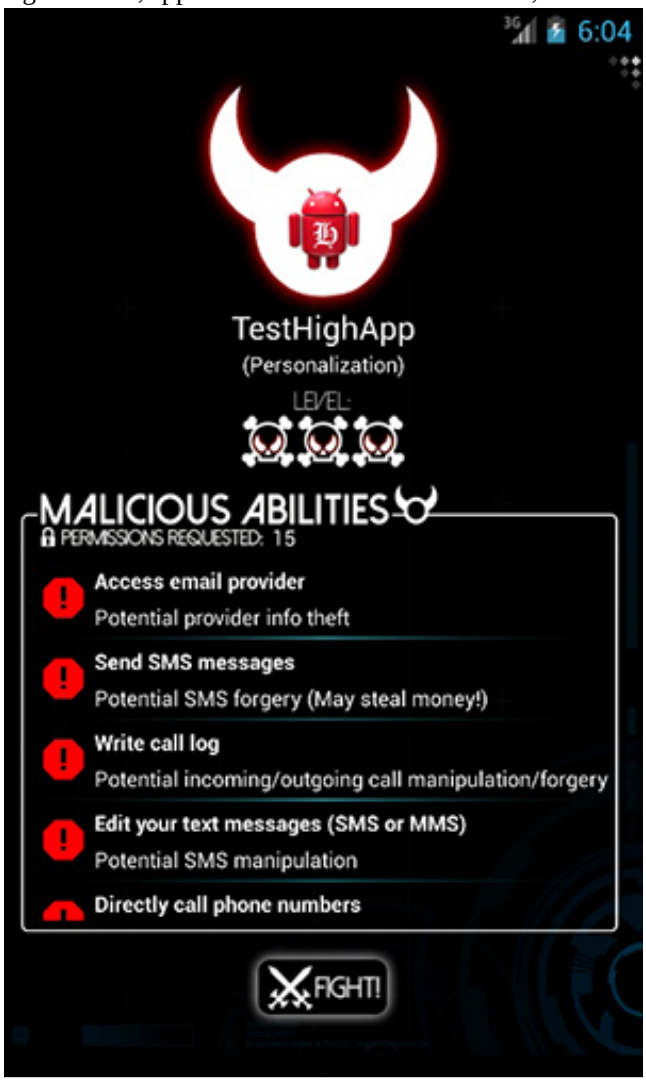

Figure 6. Evil devil detail screen

The devil details screen shows the category of the application and the application's individual requested permissions with its short description of the evaluation. The requested permissions will be mapped as the devil's abilities in the game. When the user taps any of requested permission

Figure 7. Good devil details screen

a pop-up with the actual permission codename and a long description of the evaluation will be displayed.

B. *How the Designed Application is better than Existing Solutions*

The proposed security application, DrShield, special and unique compared to existing solutions on the market.

Firstly, it is simple and specifically designed to detect malicious applications. DrShield only requires one permission, "Full Internet Access", in order to query the online Google Play Store to figure out which category each scanned application belongs to.

Existing security applications provided by commercial companies require many requested permissions. An example is shown in Fig. 8. The solution provided by AVG Mobile requests a total of 51 permissions, including of potentially

Figure 8. Google Play Store displaying the application's permissions

dangerous permissions, such as "send SMS messages" and "directly call phone numbers".

The large number of requested permissions could be due to the application providing extra functionality, such as backup of phone data. These permissions represent a threat vector - a disgruntled employee could sabotage the company's security application to perform unauthorized operations on users' phones, such as collecting confidential data. Since the user agreed to grant these permissions when installing the security application such an attack would be successful.

If the same situation happens to DrShield, the disgruntled employee will not be able to do much damage since the only permission granted to the application is Internet access. The disgruntled employee cannot read your contacts or make calls to premium rate numbers without the "Read your contacts" and "Directly call phone numbers" permissions respectively. By minimizing the number of requested permissions, DrShield keeps such potential risk and damage to a minimum.

Secondly, commercial security applications can be very technical and unfriendly to non IT-savvy users. DrShield provides a gaming aspect, Story Mode, to guide non IT-savvy users to use the application in a fun and educational manner. Over time, users will know more and be more aware about the potential dangers of requested permissions, which may lead them to be more cautious when installing new applications on their phone.

Lastly, a drawback with traditional antivirus solutions is inefficiency. If there a new malicious application is released, traditional antivirus solutions will need to be updated with signatures to detect the new threat. There will be a window of opportunity for malicious applications to wreak havoc before they get detected and removed. However, with DrShield, this will not be the case. Any new variant will still have a category and the appropriate threat filter can be used to scan the application for any potential malicious intent right away

V. CONCLUSION

The use of smartphones has become increasingly popular over the years due to affordability and convenience. Android OS accounts for majority of the smartphone market share due to its open source nature, which entices many smartphone brands to build Android phones. However, this also made Android OS an attractive target for cybercriminals to develop malicious applications, which puts Android mobile users at risk. One of the greatest challenges in protecting users is to detect malicious applications among the numerous applications installed on a phone.

Upon detailed analysis, a distinct correlation between each application's category and its requested permissions was observed. Using the pattern of requested permissions to detect malicious applications can therefore be an effective method. The proposed solution, DrShield, utilizes these patterns to scan all applications installed on a smartphone to detect malicious applications. DrShield also comes in two modes, with the "Utility Mode" catering to veteran users who want to get the job done, whereas the "Story Mode" caters to the younger crowd or less tech-savvy users. The aim of the "Story Mode" is to entice users to play the game and at the end, understand more and be more aware of the potential dangers of requested permissions.

.DrShield has fulfilled all the criteria mentioned in Section II.B, which are creating a security application that detects malicious applications based on the requested permissions, is user-friendly and do not require a rooted device. Additionally, the objective of the project has been met with DrShield. It provides a solution that the user can use to scan and remove malicious applications from a device, protecting the user.

Overall, DrShield demonstrates an effective and unique approach to detecting malicious mobile applications in Android OS compared to traditional anti-virus methods. This approach is new and it has not yet been popularized. DrShield can be used as a stepping stone for future developments in this direction.

REFERENCES

[1] Android, "Signing Your Applications", http://developer.android.com/tools/publishing/app-signing.html [retrieved: Oct, 2015]

[2] R. Broida, "How to easily root an Android device", http://www.cnet.com/how-to/how-to-easily-root-an-android-device [retrieved: Oct, 2015]

[3] Bullguard, "The risks of rooting your Android phone" http://www.bullguard.com/bullguard-security-center/mobile-security/mobile-threats/android-rooting-risks.aspx [retrieved: Oct, 2015]

[4] O. Celestino "Mobile Apps: New Frontier for Cybercrime" http://www.trendmicro.com/vinfo/us/threat-encyclopedia/web-attack/119/mobile-apps-new-frontier-for-cybercrime. [retrieved: Oct, 2015]

[5] A. Decker. "How Mobile Ads Abuse Permissions" 2012 http://blog.trendmicro.com/trendlabs-security-intelligence/how-mobile-ads-abuse-permissions.

[6] F- Secure. "Mobile Threat Report Q1 2014" http://www.fsecure.com/documents/996508/1030743/Mobile_Threat_Report_Q1_2014.pdf

[7] M. Gendron (Ed.). "RiskIQ Reports Malicious Mobile Apps in Google Play Have Spiked Nearly 400 Percent" 2014 http://www.riskiq.com/company/press-releases/riskiq-reports-malicious-mobile-apps-google-play-have-spiked-nearly-400.

[8] J. Heggestuen, (2013). "One In Every 5 People In The World Own A Smartphone, One In Every 17 Own A Tablet " 2013.http://www.businessinsider.com/smartphone-and-tablet-penetration-2013-10.

[9] IDC." IDC: Smartphone OS Market Share" http://www.idc.com/prodserv/smartphone-os-market-share.jsp.

[10] M. Kassner," Some important facts about Android antivirus applications", http://www.techrepublic.com/blog/smartphones/some-important-facts-about-android-antivirus-applications [retrieved: Oct, 2015]

[11] P. Marchant, "Top 10 Android security tips", http://www.computerweekly.com/feature/Top-10-Android-security-tips [retrieved: Oct, 2015]

[12] Nielsen., "Smartphones: So many apps, so much time", http://www.nielsen.com/us/en/insights/news/2014/smartphones-so-many-apps-so-much-time.html [retrieved: Oct, 2015]

[13] P. Paganini, "Phishing goes mobile with cloned banking app into Google Play", http://securityaffairs.co/wordpress/26134/cyber-crime/phishing-goes-mobile-cloned-banking-app-google-play.html. [retrieved: Oct, 2015]

[14] Sophos,"Sophos Security Threat Report 2014" http://www.sophos.com/en-us/medialibrary/PDFs/other/sophos-security-threat-report-2014.pdf, pp. 9

[15] Svetius," How to Root Any Device" http://www.xda-developers.com/root. [retrieved: Oct, 2015]

[16] C. Toombs, "XPrivacy Gives You Massive Control Over What Your Installed Apps Are Allowed To Do", http://www.androidpolice.com/2013/06/23/xprivacy-gives-you-massive-control-over-what-your-installed-apps-are-allowed-to-do [retrieved: Oct, 2015]

[17] L. Tung, "Google removes 'awesome' but unintended privacy controls in Android 4.4.2", http://www.zdnet.com/google-removes-awesome-but-unintended-privacy-controls-in-android-4-4-2-7000024329. [retrieved: Oct, 2015]

[18] M. M. Zaki, S. Shahrin, .A. M. Faizal, S. Rahayu, and Y. Robiah, " Android Malware Detection System Classification". Research Journal of Information Technology, 6: 325-341, http://scialert.net/abstract/?doi=rjit.2014.325.341, pp. 329.

[19] F. D. Cerbo, A. Girardello, F. Michahelles, and S. Voronkova., "Detection of Malicious Applications on Android OS" Computational Forensics, LNCS 6540, 2011, pp 138-149.

[20] L Lei, Y. Wang, J. Jing, Z. Zhang and X. Yu.,"MeadDroid: Detecting Monetary Theft Attacks in Android by DVM Monitoring", Information Security and Cryptology - ICISC 2012, LNCS 7839, 2013, pp 78-91

[21] Y. Zhou, Z. Wang, W. Zhou and X. Jiang.,"Hey, You, Get Off of My Market:Detecting Malicious Apps in Official and Alternative Android Markets" Proceedings of the 19th Network and Distributed System Security Symposium (NDSS 2012), 2012, pp 317-326

 ISBN: 978-1-61208-439-8