Measurement 133 (2019) 396–411

Contents lists available at ScienceDirect

Measurement

journal homepage: www.elsevier.com/locate/measurement

"Less Give More": Evaluate and zoning Android applications

Mohd Faizal Ab Razak [a,b], Nor Badrul Anuar [a,*], Rosli Salleh [a], Ahmad Firdaus [b], Muhammad Faiz [a], Hammoudeh S. Alamri [b]

[a] Department of Computer System and Technology, Faculty of Computer Science and Information Technology, University of Malaya, 50603 Kuala Lumpur, Malaysia
[b] Faculty of Computer Systems & Software Engineering, Universiti Malaysia Pahang, Lebuhraya Tun Razak, 26300 Kuantan, Pahang, Malaysia

ARTICLE INFO

Article history:
Received 3 January 2017
Received in revised form 11 August 2018
Accepted 11 October 2018
Available online 13 October 2018

Keywords:
Risk assessment
Analytical hierarchy process (AHP)
Mobile device
Android
EZADroid

ABSTRACT

The Android security mechanism is the first approach to protect data, system resource as well as reduce the impact of malware. Past malware studies tend to investigate the novel approaches of preventing, detecting and responding to malware threats but little attention has been given to the area of risk assessment. This paper aims to fill that gap by presenting a risk assessment approach that evaluate the risk zone for an application. The permission-based approach is presented for evaluating and zoning the Android applications (EZADroid), based on risk assessment. The EZADroid applies the Analytic Hierarchy Process (AHP) as a decision factor to calculate the risk value. A total of 5000 benign and 5000 malware applications were drawn from the AndroZoo and Drebin datasets for evaluation. Results showed that the EZADroid had achieved 89.82% accuracy rate in classifying the application into a different level of risk zones (i.e. very low, low, medium, and high).

1. Introduction

As technology grows with time, the mobile device landscape continues to evolve. The increasing speed, power, storage space and available application services such as online shopping and games have led mobile users into adopting this technology. As the number of mobile devices increase, malware attacks, especially on Android, also rise consecutively [1]. According to Trend Micro 2016 Security Prediction's report, malware growth is expected to be 20 million by the end of 2016 while Android is labeled as a high-risk mobile application [2]. The trend shows that more sophisticated mobile malware with zero-days attack bypassing a modern security defenses. The attacks are difficult to identify or mitigate since the number of attacks is probably higher and unnoticed. An Android application known as GodLess, for example, has infected 850,000 mobile devices worldwide [3]. Out of the 96% surge in mobile device infections, Android was the hardest hit by 74% cases while the iOS device recorded only four (4) % of infection rate [4]. In 2014, Symantec discovered more than 317 million new malware while PandaLabs was able to neutralize 75 million malware [5]. These figures demonstrate that, on average, nearly one (1) million malware are released every day [6]. This explosive growth in Android malware causes serious infiltrations to the Android system [7]. In addition, the trends that will dominant in 2018 and influence malware attacks such as evasive techniques, malware vaccination, ransomware and Android malware [8,9]. These trends will continue to dominate in 2018 with continually-evolving threats [9]. To combat these problems, security researchers have designed various anti-virus, anti-malware software and risk analysis which used to detect and analyze the risk of malware applications [10]. This is done by analyzing the patterns of the malicious activities which is achieved by looking at their pre-defined signatures or by monitoring the application's behavior.

In particular, the noticeable change of the malware behavior detected through existing approaches such as anti-virus, firewall and the Intrusion Detection Systems (IDSs) [11,12]. However, these approaches are inadequate to fully eliminate the attack since recent malware are capable of evading detection [13,14]. Google Play has introduced a detection approach known as Bouncer [15]; it provides automatic scanning and the removal of potentially malicious applications [13,16]. However, it is inefficient enough to examine applications from third party sources [13] which may lead to privacy violations [17]. Another approach provided by Google Play is a security mechanism which restricts Android applications from accessing private data by using a permission-based system [18]. Nonetheless, the mechanism is also unable to completely protect Android from malware attacks. Expanding on this, the APK Auditor and Stowaway is one mechanism that uses the permission analysis to classify the Android applications as benign or malware

* Corresponding author.
E-mail addresses: faizalrazak@ump.edu.my (M.F.A. Razak), badrul@um.edu.my (N.B. Anuar), rosli_salleh@um.edu.my (R. Salleh), firdausza@ump.edu.my (A. Firdaus), faizzaki@um.edu.my (M. Faiz), ha.amri@gmail.com (H.S. Alamri).

https://doi.org/10.1016/j.measurement.2018.10.034
0263-2241/

[19] while the DroidRisk [20] applies the quantitative risk assessment of both Android permissions and applications to test malware. Nonetheless, both are also inadequate. In addition, without appropriate risk assessment, the users' are insecure and vulnerable to malicious attacks [21] and most of users' ignore such long list of permissions since some of the permission is quite difficult to understand [22]. Consequently, the users may install high-risk applications. These limitations of the Android permission mechanism inspire to evaluate the risk of an Android application based on their critically and providing the user with risk zones. Therefore, it is proposed that a decision-based approach such as the multi-criteria decision making mechanism be used for risk assessment.

Risk assessment is important for guarding users from dangerous application and it is a compelling issue. Although, Android security mechanism such as permission provide protection for user and device, these still show several flaws and threat. Even though, the users have some knowledge in mobile security threat, but the evaluation process for determine the risk is handled by user. This issue happen when user need to install an applications and Android only informs users about which permission need to be granted [19]. Most of users unaware about the permission and some of them just ignore permission because unable to understood [23,19]. Therefore, it is important to identify potential malware in Android applications thus to generate risk zone for warning potential malicious activities.

This study introduces the EZADroid which used to evaluate the Android applications by using minimal features but providing maximum information about them. This paper aims to evaluate the risk of Android applications based on risk zones by using Android permissions. In this paper, the Android applications are zoned into several categories (i.e. very low, low, medium and high) which inform users about the specific risk of the Android applications, based on their criticality. Specifically, this study applies risk assessment and the AHP technique so as to generate results that allow users to identify the risk zones of Android applications, thereby, protecting themselves from being affected by the threat. Both the approaches not only enlighten decision making but also the importance of each level of the findings. The features selection approach was used to select the criteria. The box plot analysis was also used to evaluate the effectiveness of the EZADroid system.

The main contributions of this study are as follows:

i. The evaluation used 5000 samples of malware application from the Drebin [22] dataset and 5000 benign samples from AndroZoo [24], both were considered an extensive amount of data to be collected and analysed for mobile device research works.
ii. The experiment evaluation used the permission based application that was extracted from AndroidManifest.xml.
iii. The EZADroid was able to zone the risks into four (4) categories: very low, low, medium and high.
iv. We applied statistical evaluation to measure the effectiveness of the EZADroid system.

The rest of this paper is organized as follows: Section 2 describes the background of the research. Section 3 describes the related literature. Section 4 presents the research methodology. Section 5 describes the proposed approach. Section 6 explains the experiment and results and Section 7 concludes the paper.

2. Background

This section presents the background of the Android security mechanism, the risk assessment and the malware detection system. The increasing popularity of the Android and its widespread adoption by users for daily personal and professional activities is attracting a significant threat. In order to avoid these threats on Android, there has to be a security mechanism and a risk assessment approach that offer users protection.

2.1. Android security mechanism

The Android security mechanism is important because it is used in a mobile device that contains a lot of sensitive information which used by attackers against the users. The application permission, application signing, and component encapsulation are types of security mechanisms used to avoid threats on the Android [25]. Table 1 illustrates the Android security mechanisms and their descriptions.

2.2. Risk assessment

Risk assessment is a process that identifies the loss, vulnerability, damage and the consequence of the action affecting costs. It is important because it presents the evaluated impact of the attack. It also helps to establish the level of safety or risk of a mobile application. The risk mechanism offered by a mobile device protects the user from unwillingly installing the malware application [26] and it does this by informing the user in the permissive mode and ensuring the right trust. Risk assessment allows the mobile user to lower the potential consequence of a threat to an acceptable level. Refs. [27,28] define risk as a component that is made up of a combination of threats and vulnerabilities. It leaves an impact on an asset when the vulnerabilities are flawed. Some researchers treat risk as a single entity [29]. The risk may result in a number of losses which include cost, performance degradation as well as functionality. Risk assessment identifies the significant measures that used to control the risk; it protects the asset from any of the threats and it cautions users to take reasonable steps to manage the risk. As a system, risk assessment is used to measure the risk so as to generate the risk zones on the Android applications. This study implemented static analysis technique approach to collect Android permission as proposed features for criteria evaluation. The proposed criteria are used for making a judgment in risk assessment.

3. Related works

In mobile devices, a permission is a restriction which limits the application's access to a part of the code such as a restricted API and sensitive information stored in the mobile device. This limita-

Table 1
Android security mechanisms.

Type	Description
Application permission	A special privilege that applications need to grant before installing the applications. The Android permission system is divided into several protection levels: normal, dangerous, and signature of the system. The Android application requests permissions based on the defined permissions in AndroidManifest.xml file. Android's permission is used to notify the user about the risk of installing the application [23]
Application signed	All Android applications must be signed with a certificate before installation. This certificate is important to identify the author of the application; it also contains a public key for added security
Encapsulation	The Android application encapsulates different components which are not able to be accessed from an external entity [25] which requires permission to access certain application components

tion allows users to protect their sensitive data and code which could be compromised if unprotected [16]. Any application to be installed in a mobile device needs to request permission from the user. However, a malware application known as LevelDropper has the capability to auto root itself on mobile devices; it also has the capability to perform the installation of other applications without the need of user's confirmation [30]. This demonstrates that permission-based Android security mechanisms need an enhancement in giving out warnings to users about the malware activities. Other than the Android security mechanism, the malware detection system is also able to detect malware activities. Two categories of malware detection approach are available: anomaly-based and signature based detection [7]. These approaches have worked in the past but it is ineffective in evaluating the application risks. To optimize the malware detection system, the implementation of a risk assessment system is important.

Risk assessment analyzes any potential malicious applications through a statistical evaluation that is derived from chosen features which are the applications' permission. Risk assessment categorized into a few steps encompassing: identifying, assessing, mitigating and monitoring as well as responding to the risk [31]. To do risk assessment, a process of the assessment needs to be carried out in order to identify the probability and impact of the threat, which is calculated from the value of the asset, threat and vulnerability [29,32]. The risk assessment that is developed aims to identify, quantify and prioritize the risk according to the criteria it sets. A traditional risk assessment approach identifies the mobile device as a single entity only [29] and also as an asset for confidential information [32]. Examples of confidential information are calling history, location information, password and others [33]. Similar to [33,32], risk assessment also analyses the asset based on the connectivity, device and data.

Risk assessment is also effective for assessing a specific vulnerability that monitors and responds to the risk. Such a process is normally performed by malware analysts to generate risk notifications which warn against potential malware activities [20]. Nonetheless, without a proper approach that scrutinise the application, it is impossible to obtain an effective risk evaluation and an accurate risk zone. As discussed earlier, the proposed approach evaluates the risk based on the application permission. It is difficult to evaluate the risk when the application does not request access to any dangerous permissions. The Android malware such as DroidKungfuUpdate only uses certain permissions which exclude dangerous permissions [34]. This problem was, however, solved by the mobile malware detection system which uses both the dynamic and the static analysis technique which provided good criteria for risk assessment that evaluates and categorises the application into a risk zone.

Identifying risks on mobile Android applications is an important step in risk assessment. In recent times, risk assessments are being conducted by different domains such as information security [35], health [36] and mobile security [37]. Risk assessment, in combination with the Analytical Hierarchical Process (AHP), is able to improve the accuracy and the effectiveness of evaluating an application's risk. The AHP is a structured technique for the multicriteria decision-making approach that was developed by Saaty [36]. With the ability for multicriteria decision making, AHP has used in many studies. It is broadly applicable because each application is well structured and effective in making a decision. For example, in fields of operation studies such as [38] and [39], they applied AHP to investigate the most applicable mobile service for the consumer. AHP method is used to identify the linkage between perceived performance benefit with and good practice in Small and Medium Enterprise (SME) [40] but [41] applied risk analysis to rating the building based on excellent, good, medium, low and poor as well as to lessen users' safety. Within computer security studies, Refs. [42,43] adopted AHP to evaluate the trustworthiness of Android applications. Moreover, the methodology has proven to be very suitable for decision making and capable producing results that agree with expectations [44]. According to a literature review by [45] shows that AHP most frequently used in multicriteria decision making and has been thoroughly tested by thousands of organization around the world for the last 35 years [46]. The combination of AHP and risk assessment gives an advantage for decision making to assess quantitative of risk. In addition, a survey by [47] shows that AHP is a popular method to assess and manage information security risk. There are relatively few research works that use risk assessment to zone the risk in the study of malware. Based on this lack of research, it is also unlikely that an effective approach for evaluating the application risk is available. This lack has motivated the current study.

The significant process of assessing the risk is to determine the risk level. There are two approaches to do this [48,31]: qualitative [32] and quantitative analysis. The qualitative risk analysis is an approach for ranking the impact and for identifying the risks according to the potentials. Ref. [49] proposed scoring as an approach to improve risk communication for the Android applications by using a probabilistic model. It is one way to prioritise the risks and responses [33,50,51]. Another way is through the quantitative risk assessment approach which usually uses a quantitative scale to calculate cost [32], values of threat, vulnerability, and consequence of risk. Ref. [20] proposed a significant quantitative risk assessment network, the DroidRisk, for evaluating malicious activities on Android permissions. This framework relies on the risk assessment which provides risk signal warnings to the user about the potentially malicious activities. It also comes with a risk mitigation approach that allows for protection and control, blocking any threats before they attack, so as to reduce the impact [52]. Risk mitigation is a process to control, remove and reduce the probability of the risk to an acceptable threshold [48,53]. The andromaly [54] uses threat assessment to mitigate the threat by applying an automatic or manual response before sending the alerts to the user.

The above approaches demonstrate the rationale for applying security mechanisms and risk assessment in evaluating risk on Android applications. The risk assessment evaluation is similar to the malware detection system where performance evaluation such as accuracy, plays a significant role in measuring the efficiency. Various research has been conducted so as to gather the accuracy of risk evaluation which uses permission-based criteria. This approach also helps to locate the risk zone for Android applications. Using permissions to analyse risk that is associated with applications is a well-studied approach, both in industry and academic research. Major antivirus vendors use a similar approach to alert users of potentially risky or privacy-invasive applications. Kirin [55], for example, uses permission-based rules to prevent users from installing an application with risky permission combinations (e.g., Internet and Location permissions together be used to infiltrate location to the network). In contrast, Whyper [37] recommends using application descriptions as well as permissions to assess risk. The risk evaluation performed improves users' awareness to the risk and protects them from the harmful applications. The box plot analysis that is applied in this study shows an acceptable evaluation rate that is able to project a risk zone. The following section describes the methodology applied.

4. Research methodology

This section presents the overall workflow of the experiment. The risk assessment approach was applied to effectively describe and analyze Android applications, in search for the risk zone. The

approach is based on the permission-based system which aims to verify the risk of requesting and granting permissions to Android applications. In this regard, it requires access to the Android's permission system. The permission based system was selected because it is light weight [56] and effective for defining the risk criteria. The outcome of both the risk assessment and the static analysis are combined and then compared with the Android permission group. This will help to define whether the applications are high risk, medium risk, low risk or very low risk. Similar to Refs. [57,58], this study used risk assessment and machine learning to determine the risk.

4.1. Criteria selection

This study used 10,000 Android applications samples as the training set. The total number of benign and malware samples were 5000 each. We manually predefined the samples with their appropriate labels, i.e. benign or malware. However, it is important to note that we validated the labelling process by checking the Android application's status from VirusTotal. In other words, we label a sample as malware after running it through VirusTotal. VirusTotal is an online website that checks for viruses through URL or file upload [59]. It is highly reliable as it inspects the sample and aggregates the result of over 70 antivirus scanners. VirusTotal is also widely used by researchers to provide the ground truth in their works [60,61]. After that, this dataset samples are used for criteria selection.

The criteria selection uses a specific metric which computes and returns a score for each feature individually [54]. The Waikato Environment for Knowledge Analysis (WEKA) approach was implemented for the criteria selection which includes using Information Gain to select the best criteria for use. This paper utilized Weka as the machine learning platform. Weka is a well-established software that has a collection of machine learning algorithms [62]. It is a well-rounded and complete software suite that fits the objective of this research. We implemented Information Gain in Weka directly to our dataset. This approach saves us time from manually coding the algorithm. Besides that, the reliability and accuracy of Weka's algorithms are well recognized [63,64]. The criteria with a high value of Information Gain is selected. In this way, 10, 20 and 30 criteria were selected based on Information Gain values ranging from 0.033 to 1.0. The best criteria helps to improve the performance measure [65]. Table 2 shows a list of the criteria recorded using the Information Gain approach. These were then stored in the database for the risk assessment process.

It appears that phone calls and messages were the top of the group in the permission-based system which represents the type of features noted on the Android developer [66]. It is necessary to assert that this group is significant in the risk assessment approach.

4.2. Risk assessment criteria

Android applications require permission granting from the users in order to invoke the Android API successfully. The declared permission in AndroidManifest.xml file is important and effective for revealing the potential risk and for it to be used as a warning message to notify users. Most of the risky applications require a combination of some permissions in order to launch the attack. Fig. 1 illustrates the percentage of the top ten (10) most requested permission in the benign and malware applications of the sample dataset.

Fig. 1 shows that the INTERNET is the most commonly used permission by both the malware and benign applications, with 94.06% and 80.45% respectively. This is because the INTERNET permission is mandatory for applications to access the Internet, especially for an application update. The permission access to READ_PHONE_STATE is also highly requested by the malware. This figure shows

Table 2
List of criteria.

Information gain value	Criteria	Permission group
0.2776	android.permission.READ_PHONE_STATE	Phone Calls
0.2286	android.permission.SEND_SMS	Messages
0.1908	android.permission.READ_SMS	Messages
0.1739	android.permission.RECEIVE_SMS	Messages
0.1285	android.permission.RECEIVE_BOOT_COMPLETED	Application Information
0.1067	android.permission.WRITE_SMS	Messages
0.1016	com.android.launcher.permission.INSTALL_SHORTCUT	Properties
0.078	android.permission.INSTALL_PACKAGES	Application Information
0.0686	com.android.launcher.permission.UNINSTALL_SHORTCUT	Properties
0.0685	com.android.browser.permission.WRITE_HISTORY_BOOKMARKS	Personal Information
0.067	com.android.browser.permission.READ_HISTORY_BOOKMARKS	Personal Information
0.0631	com.lge.launcher.permission.READ_SETTINGS	Dev_Read_Setting
0.0631	com.motorola.launcher.permission.READ_SETTINGS	Dev_Read_Setting
0.0624	com.motorola.dlauncher.permission.READ_SETTINGS	Dev_Read_Setting
0.06	com.htc.launcher.permission.READ_SETTINGS	Dev_Read_Setting
0.0586	com.fede.launcher.permission.READ_SETTINGS	Dev_Read_Setting
0.0581	com.motorola.launcher.permission.INSTALL_SHORTCUT	Dev_Install
0.0579	com.android.launcher.permission.READ_SETTINGS	Properties
0.0578	com.lge.launcher.permission.INSTALL_SHORTCUT	Dev_Install
0.0575	com.motorola.dlauncher.permission.INSTALL_SHORTCUT	Dev_Install
0.056	org.adw.launcher.permission.READ_SETTINGS	Dev_Read_Setting
0.0526	android.permission.WRITE_APN_SETTINGS	Properties
0.0508	android.permission.RESTART_PACKAGES	Application Information
0.0484	android.permission.CHANGE_WIFI_STATE	Network Communication
0.0434	com.google.android.providers.gsf.permission.READ_GSERVICES	Dev_Service
0.0411	android.permission.ACCESS_NETWORK_STATE	Network Communication
0.0403	com.google.android.c2dm.permission.RECEIVE	Network Communication
0.0401	android.permission.GET_ACCOUNTS	Account
0.0359	com.android.vending.BILLING	Account
0.033	android.permission.READ_CONTACTS	Personal Information

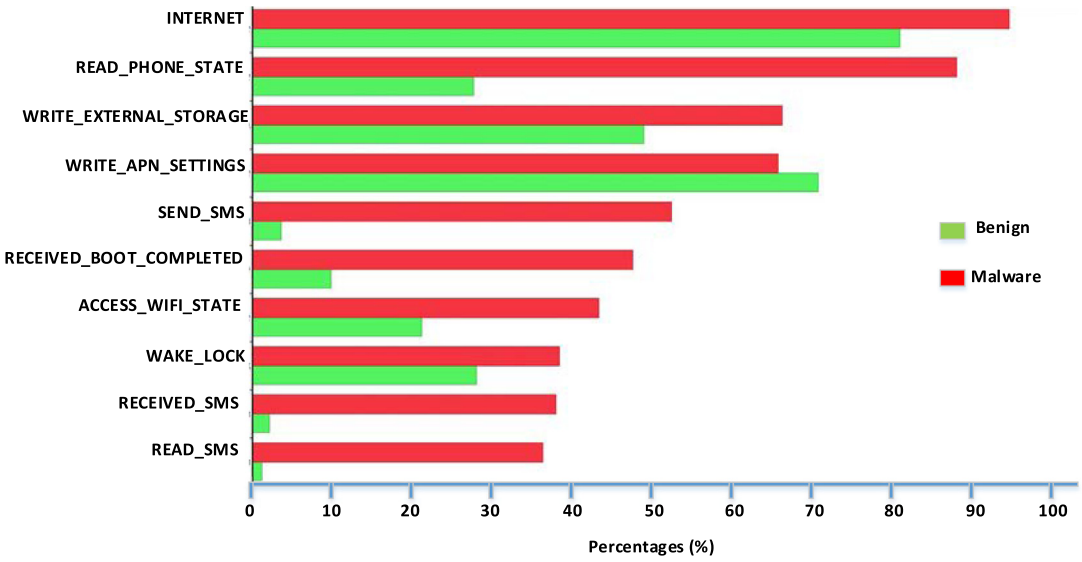

Fig. 1. Percentage of the top 10 most requested permission by malware application.

an outcome that reflects the result of Zhou and Jiang (2012) in detecting malware by permission [34] which was the first research done on the Android malware family. Moreover, the malware also requests more permissions on communication such as SEND_SMS, RECEIVED_SMS, and READ_SMS. It seems clear that the malware application is more interested in dangerous permissions in order to gain access to sensitive information. It appears to be using the Internet as a medium.

Based on the observatory analysis, the related permissions were grouped. The individual permission joins the group through the same attribute of the permission [66]. The grouping process includes reading all the permissions in AndroidManifest.xml and extracting the permission through grouping. This allows the number of comparisons to be reduced. A member of the group is then presented together as a criterion, with each criterion used to calculate the risk of the Android applications.

4.3. General framework

The AHP was used to calculate the risk of the applications. Fig. 2 illustrates the main components of the EZADroid framework which is then categorized into three components: (a) response option, (b) response systems and (c) risk zone. The proposed framework attempts to identify the risk of Android applications whether it malware or benign by assessing the risk zone. It is very important to choose appropriate approach, especially when dealing with the

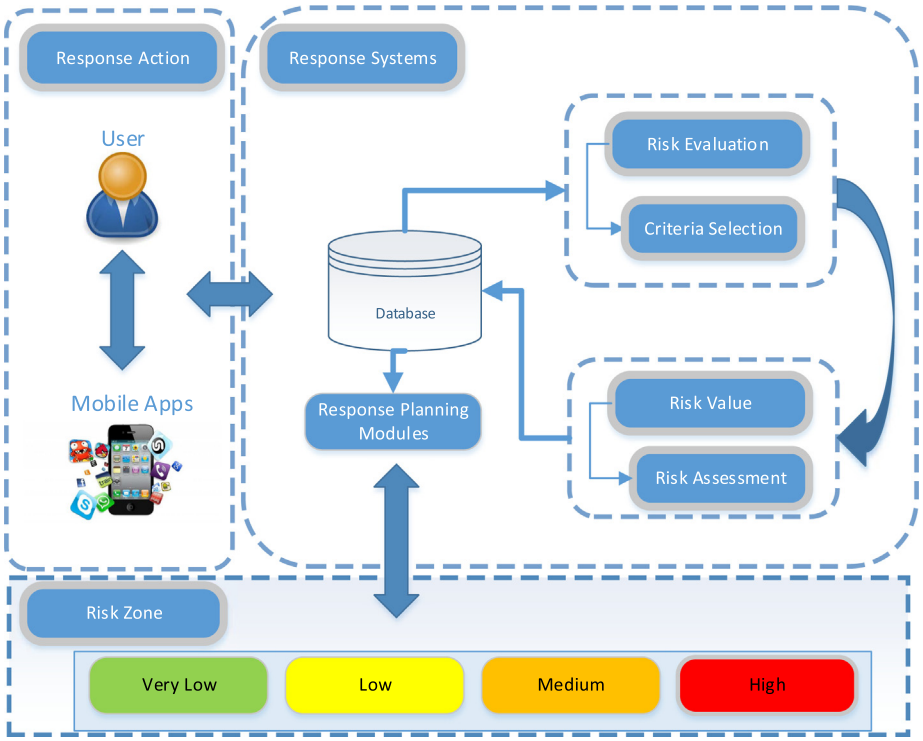

Fig. 2. The EZADroid framework.

technical aspects. With the aid of the intrusion detection system (IDS), risk assessment and machine learning approach, EZADroid supports the procedures in selecting relevant features and response to the user with the risk zone.

The response option category defines the boundaries of the response action where the system notifies the risk zone to users. In this category, the user is able to evaluate the risk on the Android applications and then respond to the risk by either accepting or removing the risk based on the risk zone categories. The response system then conducts an analysis of the Android applications. This analysis consists of risk evaluation, risk value and the response planning module. Here, the first risk evaluation is associated with the selection of the criteria which consists of the grouping permissions. The second is with respect to the risk value. In this regard, the criteria are evaluated using risk assessment. The risk value is stored in the database before it is submitted to the risk zone. This is achieved by using the response planning module. The risk zone categories have four (4) types of indicators: very low, low, medium and high risk. If the risk is greater than a certain threshold, the warning notification appears on the mobile device to notify the risk zone to the user. Each of the risk zones is presented in different colors to indicate the level of risk and to improve user awareness.

5. Proposed approach

The risk assessment approach is used to improve the effectiveness of the risk evaluation by generating a risk zone for the user. It serves as a warning against malicious applications (e.g. very low, low, medium, and high. This study submits Android applications to VirusTotal to validate and ensure the trustworthiness of dataset. In addition, the proposed approach utilized the relevant features as criteria in multicriteria decision making by implementing machine learning approach to significantly increase detection accuracy for risk analysis and malware detection. The proposed approach achieved higher accuracy with 89.82%, The results also validated using statistical analysis. Table 3 presents the description of the proposed approach.

5.1. Risk assessment phase

The AHP approach was implemented to evaluate the risk. The approach uses a pair-wise comparison of criteria to evaluate the weight of the criteria; this is in line with the main objective of the hierarchical structure [67]. The pair-wise comparison was performed through the matrix table which evaluates the consistency of the judgment [67]. The comparison matrix (A) takes the size $n \times n$ where n denotes the number of criteria being compared, which is relative to the specific elements. The elements of the matrix are a_{ij}. Table matrix A demonstrates an evaluation that is similar to [68]:

$$a_{ij} = a_{jk} \times a_{jk}$$

$$a_{ij} = 1/a_{jk}$$

where i, j and k are any elements of the matrix A.

Table 3
Description of proposed.

Propose	Description
AHP approach	One of the Multi-Criteria decision-making approaches for analyzing decisions
Risk assessment phase	An approach that is derived from an evaluation of threat and the impact of risk
Judgment matrix	This technique gives an analysis to construct the matrix consistency

$$\mathrm{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$

where $a_{ij} = 1$ and $i = j$.

In order to evaluate the consistency, the normalization table matrix of A is computed using N matrix

$$N = \begin{pmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{pmatrix}$$

where $w_{ij} = a_{ij}$

$$w_{ij} = \frac{a_{ij}}{\sum_{i=1}^{n} a_{ij}}$$

$\sum_{i=1}^{n} a_{ij} w_i = 1$ is the sum of the columns

The sum of the value of $\mathscr{n}$ is then divided to find the relative weight

The weight of $i = w_i = \frac{\sum_{j=1}^{n} w_{ij}}{n}$

Note that $\sum_{i=1}^{n} w_i = 1$

A is reflected as consistent if $A \times w = n \times w$

The Eigenvalue problem could be solved by using this equation, where the largest Eigenvalue is greater than or is equal to $n(\lambda_{max} \geq n)$. The more the value (λ_{max}) becomes closer to n, the more consistent it becomes. The $(\lambda_{max})$is equivalent to the total of the criteria of the column vector AW.

The consistency ratio (CR) is calculated as CR

$$= \frac{Consistency\ \ Index\ \ (CI)}{Random\ \ Consistency\ \ of\ \ A}$$

The value $CI = \frac{\lambda \max - n}{n - 1}$

The level of consistency is acceptable when the CR is less than or is equal to 0.10. Otherwise, it requires a reexamination of the judgment for the values of a_{ij}. This judgment is necessary in order to keep the consistency.

5.2. Judgment matrix

In the AHP approach, the hierarchy of risk assessment is measured through pairwise comparisons which is important for estimating the relative elements noted in a hierarchy with a preceding level. The pairwise comparison is measured by using the ratio scale which is used to define the intensity of the important judgments. Table 4 shows the fundamental scale of the absolute numbers of the AHP approach [69].

Table 4 illustrates the judgments through the fundamental scale of the AHP. These scales were used to determine the relative importance of each criterion, starting with the scale of 1 until 9. Table 5 illustrates the judgment matrix for the decision factor and indicator used in risk zone threshold.

Table 5 shows a comparison of the same layers with the upper layers so as to generate a judgment matrix. The resulting risk assessment is demonstrated in the experiment and result section.

6. Experiment and result

The assessment consists of two (2) experiments. The first applies the risk assessment approach to evaluate the risk value. The second generates a risk zone. The experiment uses the box plot analysis to show the difference between the malware and the benign applications. The box plot analysis was able to discover

Table 4
Fundamental scale of absolute numbers.

Intensity of importance	Definition	Explanation
1	Equal importance	Two activities contribute equally to the objective
2	Weak	
3	Moderate importance	One activity is slightly being favored over another based on experience and judgment
4	Moderate plus	
5	Strong importance	One activity is strongly being favored over another based on experience and judgment
6	Strong plus	
7	Very strong or demonstrated importance	One activity is very strongly favored over another based its dominance in practice
8	Very strong	
9	Utmost importance	One activity is utmost important favored over another based on the highest possible order of affirmation
Reciprocals of above	If nonzero numbers assigned to activity *I* when compared with activity *j*, then when compared with *I*, *j* has the reciprocal value	A sensible assumption
Rationals	Ratios originate from the scale	If consistency were to be forced by obtaining n numerical values to span the matrix

Table 5
The Judgment matrix criteria.

Permission group	Messages	Personal information	Application information	Properties	Phone calls	Normalized
Messages	1	5	5	5	5	0.54
Personal Information	0.2	1	0.5	0.5	0.5	0.07
Application Information	0.2	2	1	2	2	0.16
Properties	0.2	2	0.5	1	2	0.12
Phone Calls	0.2	2	0.5	0.5	1	0.09

Consistency ratio = 0.043.

the risk potential and to predict the risky malware. It plays an important role in determining how relevant the EZADroid was. Finally, the risk zone was determined after the results were gathered. The following sections describe the experiment in greater details.

6.1. Evaluation procedure

This paper evaluates the risk of 5000 benign and 5000 malware applications extracted from the sample datasets. Each application permission was extracted so as to collect their permissions from AndroidManifest.xml. Each of the permission on the application was stored in the database as a criterion collection. This criterion is important to guarantee the accuracy of the proposed approach. The permission-based approach was used to identify the risk zone. After the EZADroid accepts the application's permission-based behavioral data, it computes the risk value and then determines the risk zone to see whether the application is very low, low, medium or high risk. Table 6 shows the distribution of the sample datasets used by previous works.

As seen in Table 6, previous studies had used sample datasets for testing and training their machine learning algorithms. Besides that, the table also shows that previous studies applied less features to save time and the cost of experiments, thereby, minimising the time for real world implementations. It is crucial to select less features when experimenting to determine the efficiency and effectiveness of research work [71]. The performance of the model largely depends on the quality of the features rather than just the quantity. A large number of features may contain redundant information which leads to lower true positive rates. Besides that, this study used minimal features as an attempt to reduce the dimensionality of the dataset.

Table 6
Distributions of datasets.

Ref.	Total of benign	Total of malware	Total of features
[70]	3500	5000	11
[71]	5551	5551	12
[72]	20	1000	11
[73]	880	880	13
[25]	400	400	13

6.2. Risk assessment evaluation

This section presents the risk assessment evaluation which uses the AHP approach. Table 7 shows the results obtained. It classifies the benign and malware applications accordingly. It appears that all the malware and benign applications had obtained the risk values from the AHP analysis. From the risk values noted, the risk zone was classified accordingly. In order to validate the experiment, VirusTotal [74] was used to strengthen the risk zone evaluation. Table 7 illustrates samples of the evaluation results and the risk zone on applications.

Table 7 is classified into six (6) columns: SHA256, Virus Total, family, risk value, risk zone and class. The experiment in this study used a generated SHA256 hash to provide a unique key for each sample application. The columns named *VirusTotal and class* presents the distinction between the malware and benign applications taken from the sample dataset. The *family* column shows the particular malware family of the sample application that was discovered during the experiment. The *risk value* column shows the weight of the risk that was measured by the risk assessment. In addition, the *risk zone* displays the level of risk generated from the risk value.

In order to estimate the risk value of Android applications, this study calculate the risk value of permission using multicriteria decision making (Table 5). There are group of permission with different of value. Thus, this study able to evaluate the risk zone of Android application. The more permission used in dangerous category by Android application present higher risk value. It is certain

Table 7
Samples evaluation and risk zone on applications.

Sha256	VirusTotal	Family	Risk value	Risk zone	Class
78c1c57100bc14f9689c3f670d48405d9eb7487df1a34a846296f8dd4ab34e33	36	Plankton	0.760934	High	Malware
1ee4f5a8812ba86eda9f12f1e76a1a44e4d318bf4799eb7fd22c3c6a8a0f8ad2	34	Plankton	0.696939	High	Malware
3dc3631adbae697a10edfea65c06ebda751741302b6ba95c4f6c6031db71ce74	41	ExploitLinuxLotoor	0.656039	High	Malware
4419c922b2926246ffb5c4d427920b8785b7853f1f2c400914531ce99ad6164e	38	ExploitLinuxLotoor	0.656039	High	Malware
97d9acf46ba6e3bd759f74d0f2f312165f143a815ff41d26a212f3f99b20b8c6	35	FakeRun	0.559957	High	Malware
9e9fef1079a8d20a3074a1be16be029333d863add15dd0a44d67ab685bee7ea4	34	FakeRun	0.559957	High	Malware
06b53d3ea2aeee828123194b4cea8135f5b868296d8d7ab3cb839e34b2f04d6a	39	Adrd	0.557189	High	Malware
294cfb2bc890b65d7bc9135225369ab9bbd0ca81baa109f829e2c22478b4db2f	37	Adrd	0.557189	High	Malware
08ad6b366abf609018b1866f609d132ecdb66981aae540d3316c7584c816b179	38	BaseBridge	0.423385	High	Malware
09ac19bce6a6c98948ceb7db6398c0cddf2cb9167d547597731bc44411371478	38	BaseBridge	0.423385	High	Malware
00f24c9904ce23bae5a3cc4ca5a1bd13ed811b57ca772032530d415ccda02f04	0	Google Play	0.197446	Medium	Benign
00f28a5c4851f2702fc61753c21867c68916d00536d59b7c4e1d2bbbe8c7ca00	0	Google Play	0.031939	Very Low	Benign
00f2915b170f755efa3409c3ccd12fe5b1edd905592aad75d957295a1a616650	0	Google Play	0.115419	Low	Benign
00f29243375e2151947287b52f81a73a46a9e21b50a82e7cd7e3b8a8d6e6cafc	0	Google Play	0.197446	Medium	Benign
00f29ffd36c87e9138c65e09e5b455b4dbca29cbb5f37fc9bdd01c9ea73fd9a6	0	Google Play	0.031939	Very Low	Benign
00f2ab5f10c9efbfac38cc1db04189e4e762586b65878c0502d56f47ffd77c43	0	Google Play	0.063623	Very Low	Benign
00f2ad31b324ccaa59547f247409be3b16f3ab22fa5f268b5cd7029548936552	0	Google Play	0.179042	Medium	Benign
00f2bc42f0155699fb86c129f6c98eb1d563efe81aeb1d1e82d09f72fcd4be4b	0	Google Play	0.031939	Very Low	Benign
00f2bf903af178f5eb0a4f294b33699129cbf27cf39f6c156fe98d7920628803	0	Google Play	0.031939	Very Low	Benign

that dangerous permission contribute more severity than normal permission. In Section 6.3, we used box plot analysis to determine the risk zone level of Android applications.

The proposed approach provides a solution to give rich insights into the application risk analysis by using a set of criteria that are combined with a multi-criteria decision approach. It automatically performs the permission analysis of applications. The level of risk provides insight information related to the risk of application. It also directly shows the user the risk zone. In fact, it improves the application security and the user awareness. Moreover, the ability to distinct permissions for malware detection provides an additional protection for the user. The results reveal that the proposed approach reduces a significant potential for several malicious applications from accessing mobile devices. It analyzes application permission in order to access the signature of malicious applications. The results of the analyses demonstrate that permissions to request message are requested by a high-risk application as well as the permissions, related to the application information (i.e. RECEIVE_BOOT_COMPLETE, INSTALL_PACKAGEs, and RESTART_-PACKAGES) demonstrated risky permission.

Table 8 also shows that the applications that were infected with malware posed a threat to the mobile device. The high-risk applications came from the Plankton malware family. Once the Plankton malware is installed on mobile devices, it collects device ID and user information before sending it to remote server [75]. The remote server pushes payload dynamic onto users' mobile device to exploit the root. Our approach is effective to detect Plankton malware and identify their risk since it uses permissions such as messages, properties, phone calls and personal information. It has been discovered by [34] and it had been removed from the Android market by Google [76]. Table 8 lists the malware families with their risk values.

The results shown in Table 8 demonstrate the various Android malware families that had been present in the sample datasets, along with their risk values. Four (4) particular major malware families in the malware samples dataset were FakeInstaller, Plankton, DroidKungfu, and Opfake. Each malware family has a different risk value because each used different permissions. Table 8 also shows that the same malware family was unable to show the same risk value. Based on this, it deduced that the applications in the same malware family used different permissions. As a result, they showed different types of risk zones. To further discuss the risk zones, the following section discusses the rating threshold that was proposed in RSM [77]. This is applied to rate the risk zones. The following risk response planning was also implemented as a guide for mobile devices to activate the appropriate response.

6.3. Risk zone threshold

To establish a methodological approach for identifying the risk zone in the Android applications, this section explains the risk value and the zoning process. Four types of risk zones were applied in the experiment: very low, low, medium and high [78]. These risk zones have been used in security for the purpose of evaluating the risk impact. The risk zone has been exemplified in the works undertaken by [32,79]. They used the method for the Android platform and for incident prioritization. The risk zones of the current study are presented in different colors depending on the levels of risk as a means to increase awareness among Android users. This application of colors have also been applied by previous researchers [32,79]. Table 9 illustrates the description of the risk zone.

As the table illustrates, the risk zones determine the levels of severity of application. The different colors can further alert users of the severity of the risks involved. Very low and low-risk zone means that the risk of application is acceptable and safe to use. Fig. 3 illustrates the threshold for the risk zone.

This threshold is not a definitive value. It subjected to other reassessments, thus different scales will produce a different distribution. The selection of this threshold is important to make a suitable and significant decision of the risk zones. The risk zone threshold was adopted from Table 10 which is the distribution analysis taken from the box plot of ten (10) permissions. Furthermore, this threshold indicates the significance of the mapping process between the risk zone and the risk response planning for future works. Table 10 lists the data analysis taken from the box plot which uses ten (10) permissions. The following section describes the results of the experiment.

6.4. Result

In order to avoid any bias, an arbitrary number of criteria selection was used. This study used the criteria selection approach with three different configurations: 10, 20 and 30 criteria to select the highest out of the 378 criteria featured by the feature selection algorithms (e.g. Information Gain). The box plot analysis shown in Fig. 4 is related to the risk value for malware and benign applications. The box plot analysis is also able to identify the criteria more effectively as it differentiates the benign from the malware

Table 8
List of malware family and risk value.

No.	Family	Total	Min value	Max value	No.	Family	Total	min value	max value
1	AccuTrack	9	0.018404	0.018404	79	Loicdos	1	0.063877	0.063877
2	Adrd	78	0.219138	0.557189	80	Loozfon	2	0.139663	0.139663
3	Adsms	3	0.394271	0.412675	81	Luckycat	5	0.101884	0.101884
4	Aks	5	0.08348	0.08348	82	Lypro	1	0.182683	0.182683
5	Ansca	1	0.174222	0.174222	83	Maistealer	1	0.056183	0.056183
6	Antares	2	0.180063	0.192626	84	Mania	6	0.188524	0.401693
7	Anti	2	0.210727	0.210727	85	Maxit	1	0.461164	0.461164
8	Anudow	1	0.341261	0.341261	86	MMarketPay	1	0.380736	0.380736
9	Arspam	1	0.529656	0.529656	87	Mobilespy	12	0.063877	0.400093
10	BaseBridge	310	0.031939	0.5087	88	MobileTx	66	0.142284	0.241487
11	BeanBot	8	0.281533	0.362332	89	Mobinauten	4	0.269323	0.287727
12	Bgserv	1	0.1384	0.1384	90	Mobsquz	1	0.386558	0.386558
13	Biige	6	0.375849	0.375849	91	Moghava	1	0.050342	0.050342
14	Booster	1	0.559957	0.559957	92	MTracker	1	0.324182	0.324182
15	Boxer	8	0.064644	0.115419	93	Nandrobox	13	0.308956	0.308956
16	Cawit	1	0.214622	0.214622	94	Nickspy	11	0.188524	0.404981
17	CellShark	1	0.220463	0.220463	95	NickyRCP	2	0.090742	0.090742
18	CellSpy	2	0.063877	0.063877	96	Nyleaker	18	0.147104	0.436294
19	CeShark	7	0.148124	0.220463	97	Opfake	545	0.080799	0.39652
20	CgFinder	2	0.080799	0.080799	98	PdaSpy	4	0.166528	0.247327
21	Coogos	8	0.237542	0.318341	99	Penetho	18	0.031939	0.275321
22	CopyCat	12	0.250823	0.559957	100	Pirater	1	0.180063	0.180063
23	Cosha	9	0.174222	0.255022	101	Pirates	2	0.420764	0.420764
24	CrWind	2	0.245079	0.245079	102	PJApps	1	0.231172	0.231172
25	Dabom	2	0.145443	0.145443	103	Placms	10	0.35612	0.436919
26	Dogowar	1	0.206928	0.206928	104	Plankton	623	0.133823	0.760934
27	DroidDream	72	0.08348	0.471892	105	Proreso	2	0.163847	0.163847
28	DroidKungfu	647	0	0.401693	106	QPlus	5	0.08348	0.198467
29	DroidRooter	1	0.115419	0.115419	107	Rooter	3	0.133823	0.133823
30	DroidSheep	6	0	0	108	RootSmart	7	0.282761	0.540638
31	Ewalls	1	0.115419	0.115419	109	RuFraud	1	0.182683	0.182683
32	ExploitLinuxLotoor	48	0.031939	0.692865	110	SafeKidZone	1	0.190006	0.190006
33	FaceNiff	5	0.08348	0.115419	111	Saiva	2	0.161598	0.245079
34	FakeDoc	132	0.184165	0.401693	112	Sakezon	8	0.133823	0.190006
35	FakeFlash	3	0.063877	0.063877	113	Sdisp	1	0.080799	0.080799
36	FakeInstaller	864	0.080799	0.277017	114	SeaWeth	5	0.126129	0.35028
37	Fakelogo	2	0.101884	0.170916	115	SendPay	59	0.115419	0.319666
38	FakeNefix	1	0.115419	0.115419	116	SerBG	14	0.214622	0.436919
39	Fakengry	13	0.131969	0.548728	117	SheriDroid	1	0.16428	0.16428
40	FakePlayer	5	0.099203	0.099203	118	SmForw	1	0.121199	0.121199
41	FakeRun	61	0.267627	0.559957	119	SMSBomber	1	0.177382	0.177382
42	FakeTimer	12	0.031685	0.133569	120	Smspacem	1	0.287727	0.287727
43	Fakeview	1	0.279266	0.279266	121	SMSreg	27	0.064644	0.613638
44	FarMap	1	0.308956	0.308956	122	SMSSend	1	0.155819	0.155819
45	Fauxcopy	2	0.050342	0.133823	123	SmsSpy	1	0.123447	0.123447
46	Fidall	2	0.148124	0.148124	124	SmsWatcher	3	0.105044	0.105044
47	FinSpy	3	0.351604	0.351604	125	SMSZombie	2	0.313825	0.313825
48	Fjcon	6	0.162937	0.358741	126	Spitmo	11	0.245079	0.245079
49	Flexispy	2	0.338069	0.369754	127	Spy.GoneSixty	1	0.088888	0.088888
50	FoCobers	15	0.196218	0.268556	128	Spy.ImLog	1	0.115419	0.115419
51	Foncy	1	0.080799	0.080799	129	SpyBubble	3	0.063877	0.362314
52	Gamex	5	0.050342	0.147357	130	SpyHasb	6	0.018404	0.083048
53	Gapev	7	0.133823	0.26209	131	SpyMob	3	0.230405	0.230405
54	Gappusin	58	0.031939	0.237542	132	Spyoo	3	0.271572	0.271572
55	Gasms	1	0.223083	0.223083	133	SpyPhone	5	0	0.031939
56	Geinimi	65	0.281929	0.45347	134	Stealer	14	0.206928	0.25727
57	Generic	1	0	0	135	Stealthcell	5	0.400093	0.400093
58	GGtrack	3	0.204679	0.28696	136	Steek	4	0	0
59	GinMaster	339	0.031939	0.462106	137	SuBatt	1	0.13941	0.13941
60	GlodEagl	1	0.287727	0.287727	138	Tapsnake	1	0.018404	0.018404
61	Glodream	67	0.036808	0.403656	139	Tesbo	2	0.259538	0.259538
62	Gmuse	3	0.133823	0.133823	140	TheftAware	2	0.471874	0.503558
63	Gonca	4	0.088888	0.120827	141	TigerBot	3	0.423385	0.423385
64	GPSpy	3	0.018404	0.018404	142	Trackplus	6	0.133823	0.198467
65	Hamob	28	0.063877	0.149192	143	TrojanSMS.Boxer.AQ	1	0.12388	0.12388
66	Hispo	2	0.297256	0.3215	144	TrojanSMS.Denofow	4	0.031939	0.031939
67	Iconosys	118	0.12388	0.260608	145	TrojanSMS.Hippo	12	0.241134	0.35612
68	Imlog	42	0.031939	0.115419	146	TrojanSMS.Stealer	1	0.25727	0.25727
69	Jifake	25	0.170916	0.292114	147	Typstu	12	0.133569	0.133569
70	JS/Exploit-DynSrc	1	0.109146	0.109146	148	Updtbot	1	0.238866	0.238866
71	JSmsHider	2	0.050342	0.297317	149	UpdtKiller	1	0.540638	0.540638
72	Kidlogger	4	0.30351	0.335449	150	Vdloader	15	0.115419	0.202569
73	Kiser	9	0.45507	0.45507	151	Vidro	5	0.281887	0.281887
74	Kmin	147	0.281533	0.471892	152	Whapsni	1	0.056183	0.056183

Table 8 (*continued*)

No.	Family	Total	Min value	Max value	No.	Family	Total	min value	max value
75	Koomer	2	0.299937	0.299937	153	Xsider	18	0.050342	0.198467
76	Ksapp	6	0.336392	0.60755	154	YcChar	2	0.412675	0.49799
77	Lemon	6	0.196218	0.313472	155	Yzhc	9	0.255022	0.386558
78	LifeMon	3	0.031939	0.260862	156	Zitmo	9	0.331876	0.412675

Table 9
Description of the risk zone.

Color	Risk Zone	Description
Red	High	High to critical risk. This application is able to harm the user
Orange	Medium	Moderate risk. This application is capable to harm the user and should be put under observation
Yellow	Low	Slight risk. This application is safe but could be misused if it has malicious activities or potential threats
Green	Very low	Very low to no risk. This application is safe for use

Fig. 3. The Risk zone threshold.

Table 10
Data analysis for 10 permission.

	Malware	Benign
Min	0	0
1st Quartile	0.2306	0
Median	0.3738	0
Mean	0.3989	0.06447
3rd Quartile	0.5314	0.09468
Maximum	0.9176	0.79265

applications. This difference will suggest that the two populations belong to different distributions. Figs. 4–6 illustrate the malware and benign applications. The trend illustrates that permission-based criteria are significant and relevant for conducting risk assessment.

The distribution shown in Figs. 4–6 demonstrates a different value. This means that there is a distinction between the benign and malware applications. This evidence strengthens the experiment evaluation with a result showing an accuracy of over 80% for the risk zone. It is fascinating to note that the implementation of the EZADroid is able to determine the risk zone based on 10,000 samples.

Fig. 4 shows the distribution of a dataset. The top 50% of the malware application (2500) have high risk. They are represented

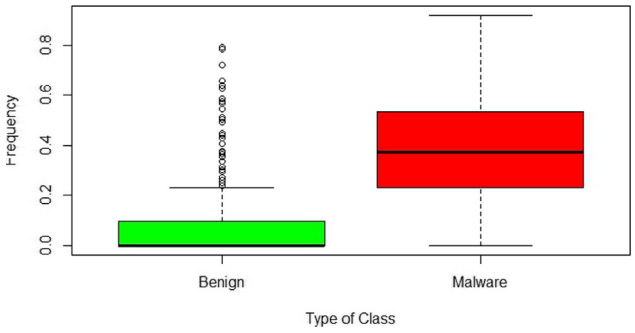

Fig. 4. The boxplot of 10 criteria.

Fig. 5. The boxplot of 20 criteria.

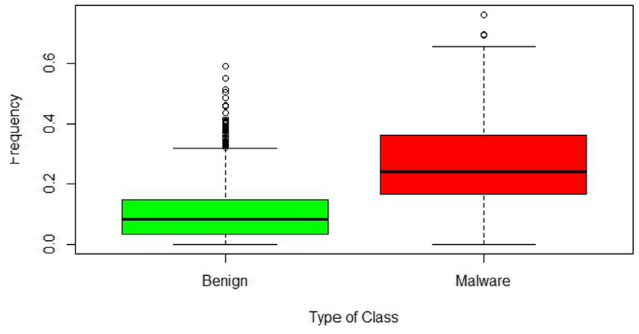

Fig. 6. The boxplot of 30 criteria.

by everything above the medium risk (the black line). The top whisker shows that 25% of the malware application came from 625 applications. The maximum whisker represents the greatest value in the malware application (risk value). Returning to the aims of this study, it is possible to state that an application is malware or benign if it shows a good overview of the data's distribution. Table 11 illustrates the risk evaluation for 10, 20 and 30 criteria.

Table 11
Risk evaluation.

	10		20		30	
	Malware	Benign	Malware	Benign	Malware	Benign
Very low	95	2643	127	2928	127	2373
Low	414	1722	814	1679	697	1530
Medium	869	483	985	300	1800	922
High	3622	152	3074	93	2376	175

Table 11 shows that the most relevant number of criteria is the 10 criteria selection. The evaluation also shows that the 10 criteria evaluation was most effective for classifying the sample dataset into the four types of risk zones. Risk zones are used to evaluate risk level of applications. It has the capability to identify potential malware based on risk level. The risk level in Table 11 explains how the permission affects the risk zone. It shows that the applications whose risk zones raise above a low-risk level set by proposed approach, make them suspicious to be malware. For benign, if any of applications show very low and low risk, are considers as clean applications.

In addition, previous study by [79], they mapped risk value to address the impact of incident in intrusion detection. Based on the risk value, the incidents are ranked from the highest to the lowest value where the highest value contribute to probability of malicious activity. By categorizing malware and benign, the EZADroid had achieved an accuracy rate of 89.82% of the malware and 87.30% for the benign outcomes, particularly when focusing on the 10 criteria selection only. The outcome thus implies that the EZADroid is able to evaluate the risk of the benign and malware applications effectively. Therefore, it improves our study goals to identify the risk of applications and detect malware. The high-risk zone detected from the 10 criteria have the highest proportion of risk with a total of 3622 malwares. This proportion is followed by the 20 and 30 criteria with a total of 3074 and 2376 malware, respectively. Fig. 7 illustrates the risk zone evaluation based on the 10, 20 and 30 criteria.

As can be seen from Fig. 7 which illustrates the risk zone evaluation on risk generated from the malware and benign applications using 10, 20 and 30 criteria, the total frequency of the application is measured by presenting the type of risk zone. In the 10 criteria selection, over 80% of the malware were detected as high risk. Based on this, it concluded that the proposed approach is efficient in evaluating the risk for most of the sample datasets. In order to warrant the significance of the proposed approach, a statistical analysis of the 10 criteria was conducted. The 10 criteria were selected because the results were more reliable, as shown on the box plot analysis. Table 12 illustrates the top applications on Google Play.

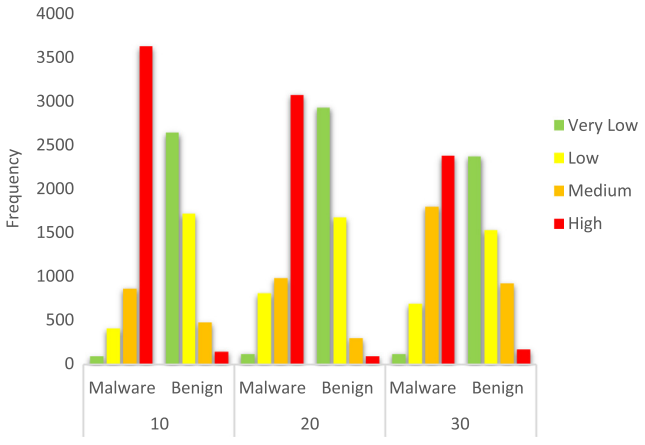

Fig. 7. Risk zone evaluation in 10, 20 and 30 criteria.

Table 12
Top applications on Google Play.

Applications	Categories	Risk zone
Facebook	Social	High
Facebook Messenger	Communication	High
WhatsApp Messenger	Communication	High
WeChat	Communication	High
Instagram	Social	High

Table 12 shows that the top Android applications which belong to the social and communication categories, have high risk because most of these applications requested dangerous permissions such as READ_SMS, RECEIVE_SMS and INSTALL_SHORTCUT. For instance, Facebook applications allow Google to display contents from Facebook mobile applications including public profile information [80]. This information could lead to cyber threat problems.

6.5. Statistical analysis

This section presents a component of the data analytics. In the context of research, statistical analysis scrutinizes data and presents a selection taken from the population. This study applied linear regression which was used to specify the nature of the relation between the malware and benign applications. A total of 10,000 applications taken from the sample dataset were applied. The experiment was able to manage the dependent (risk value) and independent (risk zone) variable score into the same row. Table 13 illustrates the variables used for analysis and the results of the mean and standard deviation.

In order to locate and interpret the relevant regression and correlation coefficients, the experiment needs to consider a variable Entered\Removed, model summary, ANOVA, and coefficient. Table 14 illustrates the independent variables. Tables 15–17 demonstrate the statistics of the data variable score. Table 15 demonstrates the correlation coefficient (r) and the coefficient of the determination (r square). It specifies the strength of the linear trend between the variables. Table 16 indicates the significant value of the independent-variable scores when compared to a predetermined α. Finally, Table 17 illustrates the y-intercept and the slope for the regression equation.

The model summary shows the correlation between the two variables (r): correlation coefficient (r) and the coefficient of determination (r square). The value of R represents the correlation coefficient that indicates the relation strength between the independent variable to the dependent variable.

Table 15 indicates the correlation coefficient (r) with the value of 0.750. It suggests that the number of sample applications has a good linear relationship. Due to this result, the coefficient of the determination (r square) shows a different value of 0.562.

Table 16 lists the ANOVA statistics which indicates whether the regression equation explains the significant portion (sig.) variability of the dependent variable and the independent variable.

Table 13
Description statistics.

	Mean	Std. deviation	N
Risk value	0.40	0.199	5000
Risk zone	3.60	0.720	5000

Table 14
Variables entered/removed.

Model	Variables Entered	Variables removed	efficient
1	Risk zone[a]	.	Enter

[a] Dependent variable: risk value.

Table 16 presents the value of sig. to be 0.0. This indicates that the proposed approach is able to reject a null hypothesis where it demonstrates a significant and fit model.

Table 16 produced a p-value of 0.0% which obviously concludes that the number of malware changes significantly with respect to the number of malware applications. This change is found from the regression equation, $y = -0.349 + (208)x$. Table 17 shows one able dependent-variable score for each independent variable score. The y value is a result from the regression equation which indicates that the pair falls on the regression line while the x value of the risk zone is substituted for the regression equation. This process is able to guess the number of application risk values and the risk zone.

Fig. 8 illustrates the risk zone analysis where the high risk indicates the exponential line. This shows that the proposed approach had managed to evaluate the risk of the samples. The mean risk value for the medium risk zone is 0.2 while the high-risk zone is 0.5.

6.6. Discussion

The EZADroid implemented a permission-based application assessment. It collected 5000 applications from a malware repository, Drebin, which was generally accepted as the Android malware dataset [22]. It composes of different families, Android Malware Genome Project [34] and also detected in the wild by a well-established AntiVirus vendor [81] which is important for detection and training purposes. However, by only detecting the category of the application in two classes (malware or benign), in risk assessment point of view, it is inadequate. Accordingly, it is important to measure and evaluate each application to identify the risk factors and threat that contain the potential to cause malicious to the user. Once the measurements have been made, the subsequent process is to analyze how severe and likely the risk. This is crucial to decide what type of control and elimination that will take place. With this implementation, we able to calculate the risk of the malware in detail. Therefore, this paper proposes EZADroid to measure the risk of the application rather than classify them.

The EZADroid used 5000 benign applications downloaded from AndroZoo [24] which belongs to Google Play. The sample contains more than five millions different Android applications and each of which has been analyses by 10 of different AntiVirus product [82]. These dataset is important for training purpose which contribute to the significant result.

In total, this paper analyzed 10,000 applications by using the risk assessment approach. Both the benign and malware applications were analyzed by VirusTotal [74] to check the validity of the malicious activities [83]. The VirusTotal provides significant results which also include a definition of all kinds of malware through 42 antivirus products (e.g. Symantec and Kaspersky) [19]. The experimental results demonstrate the effectiveness of the EZADroid in differentiating the malware from the benign applications. This is illustrated in Figs. 4–6.

Table 11 illustrates the results of the risk evaluation which evaluated the risk for the different kinds of malware family, as shown in Table 7. Table 11 also shows that the EZADroid is effective in identifying the risk for malware applications. The EZADroid had achieved an accuracy rate of over 89%, thereby raising a challenge of the risk assessment for Android mobile malware. Specifically, the permission based applications could preclude the deployment of the sophisticated risk assessment approach. Interestingly, a higher number of criteria had resulted in a lower performance of the risk assessment.

Using less criteria seems to show promising results, as illustrated in Table 11. The outcome is also supported by [84] where

Table 15
Model summary.

Model	R	R square	Adjusted R square	Std. error of the estimate	Durbin-Watson
1	0.750[a]	0.562	0.562	0.132	0.160

[a] Predictors: (constant), risk zone.

Table 16
ANOVA.

Model		Sum of squares	df	Mean square	F	Sig.
1	Regression	111.820	1	111.820	6421.648	0.000[a]
	Residual	87.030	4998	0.017		
	Total	198.851	4999			

[a] Predictors: (constant), risk zone.

Table 17
Coefficients.

Model		Unstandardized coefficients		Standardized coefficients	t	Sig.	Correlations
		B	Std. Error	Beta			Zero-order
1	(Constant)	−0.349	0.010		−36.680	0.000	
	Risk zone	0.208	0.003	0.750	80.135	0.000	0.750

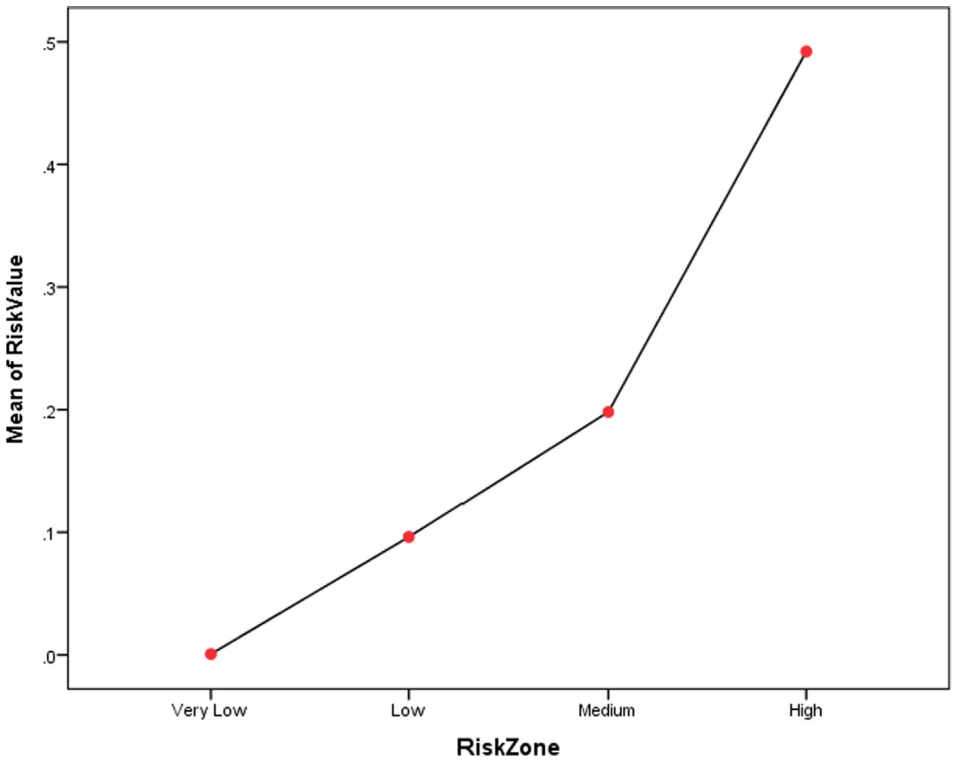

Fig. 8. Risk zone analysis.

it was noted that less criteria resulted in less time; it also reduced the cost of the experiment and it was prominent in showing more accurate results. From another perspective, the presence of the criteria (including benign and malware) provided some unique advantages in generating a risk zone and in preventing the malicious applications from affecting Android mobile users. It thus helped to increase users' awareness. One of the more significant findings that emerged from this experiment is that the number and the type of criteria could have affected the effectiveness of the risk assessment. One possible solution recommended is to adapt a machine learning approach to create a different result in the risk assessment so as to ensure the effectiveness of the risk zone.

To demonstrate the superiority of our work, we compared the results with existing solutions in Table 18. Ref. [58] conducted a study on risk assessment for Android applications that can identify malware applications using Naive Bayes classification. Ref. [58] proposed Android application package (APK) Vulnerability Identification System (AVIS) that classifies an application into malware or benign based on DEX file. The authors used 250 applications samples as the training set where the total number of malware and benign samples were 125 each. Their approach achieved approximately 94% accuracy. As a comparison, although our work recorded a slightly lower accuracy, the size of the dataset used in our study was much higher, i.e. 10,000 samples. As such, the approach proposed in this research managed to lower the risk of overfitting and has better generalization on unseen samples. On the other hand, Ref. [85] proposed XDroid, an Android application and resource risk assessment tool. XDroid implemented dynamic analysis technique to detect malware by observing the applications' behaviors into an observation set. The results showed that XDroid only managed to achieve 82% accuracy with 1000 samples training set. However, XDroid consumed high consumption of computational time to train the model. In addition, AndroDialysis [86] utilized intent and permission as features in detecting malware rather than analyzing risks. Although they marked 91% accuracy, however, they focused on intent and permission only. In contrast, our EZADroid solution implements static analysis techniques which are effective to detect known malware and fast detection. Table 18 lists the comparison with previous study.

Table 18
Table comparison with previous study.

References	Accuracy (%)	Dataset	Method
XDroid [85]	82	1000	Dynamic
AVIS [58]	94	250	Static
AndroDialysis [86]	91	7406	Static
This paper	89.82	10,000	Static

The results of experiments reveal the effectiveness of our approach. Detection rate with 89.82% of accuracy, this shows that our proposed approach performed better. The proposed approach also involves risk analysis on applications as well as responses to users' with risk zone. This raise security concerns from the users. Therefore, the sensitive data of the users are properly protected and response on time. Further that, our approach able to reduce attack from malware application and minimize the risk.

6.7. Future works

This section presents an illustration of future works. As the velocity of the change of malware increase, researchers need a new approach to be adapted into their future works. This include extending the research to refine the risk zone with risk response planning (e.g. due to the noticeable spark of malware) and to increase user awareness. Another potential research area is to address the known challenges in risk evaluation such as identifying the risk zone between the malware and benign applications by minimising the false positive. In this regard, the EZADroid needs to develop an effective risk zone by implementing the machine learning approach to achieve better efficiency and accuracy [87]. Fig. 9 illustrates a diagram of the risk zone for future works.

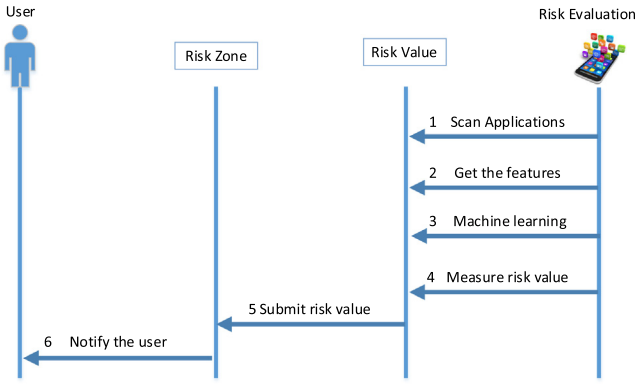

Fig. 9. Diagram of EZADroid.

Here, the user needs to scan an Android application on a mobile device. Firstly, the process of risk evaluation is used to perform the analysis on the application especially on Androidmanifest.xml to obtain application permission. Secondly, the criteria are selected among the best criteria by using the criteria selection approach. In order to achieve high accuracy and low false positive rate, it requires the implementation of a machine learning approach. After that, the process of evaluating the risk value of the criteria is performed. After obtaining the risk value, the application submits the risk value for zoning the applications. Finally, the risk zone sends a notification to alert the user about the application to indicate whether the risk is low, very low, medium or high. Then, the user responds based on the risk zone indication. This clearly demonstrates that an additional approach such as risk response planning and machine learning are needed to ensure the EZADroid's ability to determine the risk of installed applications effectively.

By design, the EZADroid's approach restricts an application from directly accessing sensitive permission. This is achieved by creating a risk zone on each installed application. The intention of this design is to provide a safe installation for all applications. The EZADroid enables users to transparently check their application risk on a mobile device and to respond accordingly based on the risk zone indication.

7. Conclusion

This paper has highlighted the significant findings of risk assessment and the risk zone for Android applications through the EZADroid which implements a permission-based application to determine the risk zone. Based on that knowledge and the effective risk evaluation, it was able to assess the mobile Android application into four (4) types of risk zone (e.g. very low, low, medium and high). The main contributions of this experiment include the following: it evaluated 5000 malware application from the Drebin dataset [22] and 5000 benign applications from the AndroZoo [24] dataset. It engaged the 10, 20 and 30 criteria approach for the risk evaluation and it performed four (4) risk evaluations (e.g. very low, low, medium and high). Based on the results, the risk on Android applications was detected effectively using less features yet yielding high detection rate.

The data were collected from the permission-based applications using static analysis. The collected data were organized in a database. In order to select the effective criteria to increase the effectiveness of the risk evaluation on the EZADroid, this paper applied the criteria selection approach. The combination of the risk assessment approach and the AHP improved the risk evaluation and also determined the risk zone for Android applications. The applications taken from Drebin and AndrZoo were used for fixing the validation and reputation of the EZADroid. It was noted that the EZADroid offered a good risk zone evaluation performance.

The EZADroid performed and achieved a 89.82% accuracy rate in the experimental evaluation of 5000 malware and 5000 benign applications based on the 10 criteria approach. It was also noted that the EZADroid was able to achieve an accuracy rate of over 80% on risk evaluation without using the machine learning classifier. This is the main advantage of the proposed approach. Moreover, the approach was also suitable to be installed on a mobile device as it provides good risk evaluation and increases user awareness about the risk of applications. This was accomplished through the illustration of the risk zone threshold.

The EZADroid approach has a limitation in running the risk evaluation for malware applications that do not use the criteria that were selected in its permission. This is a limitation of all permission-based malware detection mechanisms. Furthermore, the EZADroid was also unable to calculate the risk if the malware application does not have any permission. Nonetheless, this addressed by combining the permission-based applications with other different criteria. Another limitation traced to the general static analysis applied. Here, the static analysis was less efficient in detecting the malware with an obfuscation technique. Considering the weaknesses of this study, an uninstalled or blocked application may be a good protection alternative for mobile devices. Therefore, more investigations and experimentations on Android risk assessments need to be conducted.

Acknowledgement

This work was supported by the Universiti Malaya Research, Malaysia under Grant BKS058-2017 and Universiti Malaysia Pahang, Malaysia the Grant Faculty of Computer Systems and Software Engineering (FSK1000), RDU1803163.

References

[1] Gadget 360, Android Malware Doubled in 2015, Says Trend Micro, New on Gadget 360, 2016. <http://gadgets.ndtv.com/apps/news/android-malware-doubled-in-2015-says-trend-micro-820720> (accessed September 8, 2016).
[2] J. Clay, Continued rise in mobile threats for 2016, Micro Trend, 2015. <http://blog.trendmicro.com/continued-rise-in-mobile-threats-for-2016/> (accessed September 8, 2016).
[3] Fionna Agomuoh, 'Godless' Android malware could infect 90 percent of Google-based smartphones: how to protect your device, IDigitalTimes, 2016. <http://www.idigitaltimes.com/godless-android-malware-could-infect-90-percent-google-based-smartphones-how-protect-542161> (accessed September 8, 2016).
[4] Nokia, Nokia malware report shows surge in mobile device infections in 2016, Nokia News, 2016. <http://company.nokia.com/en/news/press-releases/2016/09/01/nokia-malware-report-shows-surge-in-mobile-device-infections-in-2016> (accessed September 8, 2016).
[5] M. Lopez, PandaLabs, 2015. <http://www.pandasecurity.com/mediacenter/press-releases/pandalabs-neutralized-75-million-new-malware-samples-2014-twice-many-2013/> (accessed November 25, 2015).
[6] Symantec, 2015 Internet Security Threat Report, 2015. <https://www4.symantec.com/mktginfo/whitepaper/ISTR/21347932_GA-internet-security-threat-report-volume-20-2015-social_v2.pdf.
[7] M.F.A. Razak, N.B. Anuar, R. Salleh, A. Firdaus, The rise of "malware": bibliometric analysis of malware study, J. Network Comput. Appl. 75 (2016) 58–76, https://doi.org/10.1016/j.jnca.2016.08.022.
[8] E. Cabrera, The New Mobile Threat Landscape, circa 2017 to 2018, Trend Micro, 2018. <https://blog.trendmicro.com/the-new-mobile-threat-landscape-circa-2017-to-2018/> (accessed July 10, 2018).
[9] M. Dimitrova, Malware trends 2018: how is the threat landscape shaping?, Sensors Techforum, 2018. <https://sensorstechforum.com/malware-trends-2018-threat-landscape/> (accessed July 10, 2018).
[10] A.N. Khan, M. Ali, A. ur, R. Khan, F.G. Khan, I.A. Khan, W. Jadoon, S. Shamshirband, A.T. Chronopoulos, A comparative study and workload distribution model for re-encryption schemes in a mobile cloud computing environment, Int. J. Commun. Syst. 30 (2017) 1–18, https://doi.org/10.1002/dac.3308.
[11] K.S. Adewole, N.B. Anuar, A. Kamsin, A.K. Sangaiah, SMSAD: a framework for spam message and spam account detection, Multimedia Tools Appl. (2017) 1–36, https://doi.org/10.1007/s11042-017-5018-x.

[12] A. Firdaus, N.B. Anuar, M.F.A. Razak, I.A.T. Hashem, S. Bachok, A.K. Sangaiah, Root exploit detection and features optimization: mobile device and blockchain based medical data management, J. Med. Syst. 42 (2018), https://doi.org/10.1007/s10916-018-0966-x.

[13] G. McWilliams, S. Sezer, S.Y. Yerima, Analysis of Bayesian classification-based approaches for Android malware detection, IET Inf. Secur. 8 (2014) 25–36, https://doi.org/10.1049/iet-ifs.2013.0095.

[14] K.S. Adewole, N.B. Anuar, A. Kamsin, K.D. Varathan, S.A. Razak, Malicious accounts: dark of the social networks, J. Network Comput. Appl. 79 (2017) 41–67, https://doi.org/10.1016/j.jnca.2016.11.030.

[15] A. Mylonas, M. Theoharidou, D. Gritzalis, Assessing privacy risks in Android: a user-centric approach, Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 8418 LNCS, 2014, pp. 21–37. http://doi.org/10.1007/978-3-319-07076-6_2.

[16] F. Tchakounte, Permission-based malware detection mechanisms on Android: analysis and perspectives, J. Comput. Sci. Software Appl. 1 (2014) 63–77.

[17] P. Naranjo, Z. Pooranian, S. Shamshirband, J. Abawajy, M. Conti, Fog over virtualized IoT: new opportunity for context-aware networked applications and a case study, Appl. Sci. 7 (2017) 1325, https://doi.org/10.3390/app7121325.

[18] A. Firdaus, N.B. Anuar, Root-exploit malware detection using static analysis and machine learning, in: The 4th International Conference on Computer Science & Computational Mathematics (ICCSCM), 2015, pp. 177–183.

[19] K.A. Talha, D.I. Alper, C. Aydin, A.P.K. Auditor, Permission-based Android malware detection system, Digital Invest. 13 (2015) 1–14, https://doi.org/10.1016/j.diin.2015.01.001.

[20] Y. Wang, J. Zheng, C. Sun, S. Mukkamala, Quantitative security risk assessment of Android permissions and applications, in: Data and Applications Security and Privacy XXVII, 2013, pp. 226–241. http://doi.org/10.1007/978-3-642-39256-6_15.

[21] N. Lo, K. Yeh, C. Fan, Leakage detection and risk assessment on privacy for Android applications: LRPdroid, IEEE Syst. J. 1–9 (2014).

[22] D. Arp, M. Spreitzenbarth, H. Malte, H. Gascon, K. Rieck, Drebin: effective and explainable detection of Android malware in your pocket, in: Symposium on Network and Distributed System Security (NDSS), 2014, pp. 1–15.

[23] A.P. Felt, E. Ha, S. Egelman, A. Haney, E. Chin, D. Wagner, Android permissions: user attention, comprehension, and behavior, Symposium on Usable Privacy and Security (SOUPS) 2012, 2012.

[24] K. Allix, T.F. Bissyandé, J. Klein, Y. Le Traon, AndroZoo: collecting millions of Android apps for the research community, 13th International Workshop on Mining Software Repositories – MSR, vol. 16, 2016, pp. 468–471. http://doi.org/10.1145/2901739.2903508.

[25] L. Gheorghe, B. Marin, G. Gibson, L. Mogosanu, R. Deaconescu, V.-G. Voiculescu, M. Carabas, Smart malware detection on Android, Secur. Commun. Networks 8 (2015) 4254–4272, https://doi.org/10.1002/sec.

[26] S.-H. Seo, A. Gupta, A. Mohamed Sallam, E. Bertino, K. Yim, Detecting mobile malware threats to homeland security through static analysis, J. Network Comput. Appl. 38 (2014) 43–53, https://doi.org/10.1016/j.jnca.2013.05.008.

[27] M.J.M. Chowdhury, R. Matulevičius, G. Sindre, P. Karpati, Aligning mal-activity diagrams and security risk management for security requirements definitions, Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 7195 LNCS, 2012, pp. 132–139. http://doi.org/10.1007/978-3-642-28714-5_11.

[28] N.B. Anuar, S. Furnell, M. Papadaki, N. Clarke, A risk index model for security incident prioritisation, in: Australian Information Security Management Conference, 2011, pp. 24–39.

[29] T. Ledermüller, N.L. Clarke, Risk assessment for mobile devices, in: Trust, Privacy and Security in Digital Business, 2011, pp. 210–221.

[30] V. Critic, Google Play store contains new auto rooting malware that downloads apps without permission, VPN Critic, 2016. <https://vpncritic.com/google-play-store-contains-new-auto-rooting-malware-that-downloads-apps-without-permission/> (accessed September 7, 2016).

[31] A. Shameli-Sendi, M. Cheriet, A. Hamou-Lhadj, Taxonomy of intrusion risk assessment and response system, Comput. Secur. 45 (2014) 1–16, https://doi.org/10.1016/j.cose.2014.04.009.

[32] M. Theoharidou, A. Mylonas, D. Gritzalis, A risk assessment method for smartphones, Information Security and Privacy Research, vol. 376, 2012, pp. 443–456. http://doi.org/10.1007/978-3-642-30436-1_36.

[33] W. Jeon, J. Kim, Y. Lee, D. Won, A practical analysis of smartphone security, in: Human Interface and the Management of Information. Interacting with Information, 2011, pp. 311–320.

[34] Y. Zhou, X. Jiang, Dissecting Android malware: characterization and evolution, in: IEEE Symposium on Security and Privacy, 2012, pp. 95–109. http://doi.org/10.1109/SP.2012.16.

[35] N. Poolsappasit, R. Dewri, I. Ray, Dynamic security risk management using bayesian attack graphs, IEEE Trans. Dependable Secure Comput. 9 (2012) 61–74, https://doi.org/10.1109/TDSC.2011.34.

[36] C.C. Lo, W.J. Chen, A hybrid information security risk assessment procedure considering interdependences between controls, Expert Syst. Appl. 39 (2012) 247–257, https://doi.org/10.1016/j.eswa.2011.07.015.

[37] R. Pandita, X. Xiao, W. Yang, W. Enck, T. Xie, Whyper: towards automating risk assessment of mobile applications, USENIX Security Symposium, 2013.

[38] S. Nikou, J. Mezei, Evaluation of mobile services and substantial adoption factors with Analytic Hierarchy Process (AHP), Telecommun. Policy 37 (2013) 915–929, https://doi.org/10.1016/j.telpol.2012.09.007.

[39] S. Nikou, J. Mezei, H. Bouwman, Analytic Hierarchy Process (AHP) approach for selecting mobile service category: (Consumers' preferences), in: Proceedings - 2011 10th International Conference on Mobile Business, ICMB 2011, 2011, pp. 119–128. http://doi.org/10.1109/ICMB.2011.29.

[40] S. Thanki, K. Govindan, J. Thakkar, An investigation on lean-green implementation practices in Indian SMEs using analytical hierarchy process (AHP) approach, J. Clean. Prod. 135 (2016) 284–298, https://doi.org/10.1016/j.jclepro.2016.06.105.

[41] N. Khalil, S.N. Kamaruzzaman, M.R. Baharum, Ranking the indicators of building performance and the users' risk via Analytical Hierarchy Process (AHP): case of Malaysia, Ecol. Ind. 71 (2016) 567–576, https://doi.org/10.1016/j.ecolind.2016.07.032.

[42] G. Dini, F. Martinelli, I. Matteucci, M. Petrocchi, A. Saracino, D. Sgandurra, Risk analysis of Android applications: a user-centric solution, Fut. Gener. Comput. Syst. 80 (2018) 505–518, https://doi.org/10.1016/j.future.2016.05.035.

[43] G. Dini, F. Martinelli, I. Matteucci, M. Petrocchi, A. Saracino, D. Sgandurra, A multi-criteria-based evaluation of Android applications, in: International Conference on Trusted Systems, 2012, pp. 67–82.

[44] D.L. Olson, The analytic hierarchy process, in: Decision Aids for Selection Problems, 1996, pp. 49–68. http://doi.org/10.3414/ME10-01-0028.

[45] J.C. Cegan, A.M. Filion, J.M. Keisler, I. Linkov, Trends and applications of multi-criteria decision analysis in environmental sciences: literature review, Environ. Syst. Decis. 37 (2017) 123–133, https://doi.org/10.1007/s10669-017-9642-9.

[46] D. Opydo, 6 Reasons to use analytic hierarchy process for collaborative decision making, TransparentChoice, 2013, p. 1. <https://blog.transparentchoice.com/analytic-hierarchy-process/6-reasons-to-use-ahp-for-collaborative-decision-making> (accessed March 5, 2018).

[47] D. Gritzalis, G. Iseppi, A. Mylonas, V. Stavrou, Exiting the risk assessment maze: a meta-survey, ACM Comput. Surv. 51 (2018) 1–30, https://doi.org/10.1145/3145905.

[48] M. Alhomidi, M. Reed, Risk assessment and analysis through population-based attack graph modelling, in: World Congress on Internet Security (WorldCIS-2013), 2013, pp. 19–24. http://doi.org/10.1109/WorldCIS.2013.6751011.

[49] H. Peng, C. Gates, B. Sarma, N. Li, Y. Qi, R. Potharaju, C. Nita-Rotaru, I. Molloy, Using probabilistic generative models for ranking risks of Android apps, in: Proceedings of the 2012 ACM Conference on Computer and Communications Security, 2012, pp. 241–252. http://doi.org/10.1145/2382196.2382224.

[50] C. Orthacker, P. Teufl, S. Kraxberger, G. Lackner, M. Gissing, A. Marsalek, J. Leibetseder, O. Prevenhueber, Android Security Permissions–Can We Trust Them?, Mobisec 2011, 2012, pp. 40–51. <http://www.springerlink.com/index/P162432470QG5671.pdf>.

[51] H. Joh, Y.K. Malaiya, Defining and assessing quantitative security risk measures using vulnerability lifecycle and CVSS metrics, in: The 2011 International Conference on Security and Management (Sam), 2011, pp. 10–16.

[52] D. Fredrik, B.L. Jørgen, V.T. Do, Threat assessment model for mobile malware, in: Information Science and Applications, 2015, pp. 467–474. http://doi.org/10.1007/978-3-662-46578-3.

[53] S. Khan, B. Nazir, I. Ahmed Khan, S. Shamshirband, A.T. Chronopoulos, Load balancing in grid computing: taxonomy, trends and opportunities, J. Network Comput. Appl. 88 (2017) 99–111, https://doi.org/10.1016/j.jnca.2017.02.013.

[54] A. Shabtai, U. Kanonov, Y. Elovici, C. Glezer, Y. Weiss, "Andromaly": a behavioral malware detection framework for Android devices, J. Intell. Inf. Syst. 38 (2012) 161–190, https://doi.org/10.1007/s10844-010-0148-x.

[55] W. Enck, M. Ongtang, P. McDaniel, On lightweight mobile phone application certification, in: Proceedings of the 16th ACM Conference on Computer and Communications Security - CCS '09, 2009, pp. 235–245. http://doi.org/10.1145/1653662.1653691.

[56] S. Lee, J. Lee, H. Lee, Screening smartphone applications using malware family signature, Comput. Secur. (2015) 1–31, https://doi.org/10.1016/j.cose.2015.02.003.

[57] H. Mojaddadi, B. Pradhan, H. Nampak, N. Ahmad, A.H. bin Ghazali, Ensemble machine-learning-based geospatial approach for flood risk assessment using multi-sensor remote-sensing data and GIS, Geomatics Nat. Hazards Risk 8 (2017) 1080–1102, https://doi.org/10.1080/19475705.2017.1294113.

[58] H. Kim, T. Cho, G.-J. Ahn, J. Hyun Yi, Risk assessment of mobile applications based on machine learned malware dataset, Multimedia Tools Appl. (2017) 1–16, https://doi.org/10.1007/s11042-017-4756-0.

[59] B. Quintaro, VirusTotal, Web analysis, 2017. <https://www.virustotal.com/> (accessed April 11, 2017).

[60] H.-D. Huang, C.-S. Lee, M.-H. Wang, H.-Y. Kao, IT2FS-based ontology with soft-computing mechanism for malware behavior analysis, Soft Comput. 18 (2014) 267–284, https://doi.org/10.1007/s00500-013-1056-0.

[61] A. Boukhtouta, D. Mouheb, M. Debbabi, O. Alfandi, F. Iqbal, M. El Barachi, Graph-theoretic characterization of cyber-threat infrastructures, Digital Invest. 14 (2015) S3–S15, https://doi.org/10.1016/j.diin.2015.05.002.

[62] U. of Waikato, Waikato Environment for Knowledge Analysis (Weka), Machine Learning, 2017. <http://www.cs.waikato.ac.nz/ml/weka/> (accessed March 8, 2017).

[63] P. Kaur, M. Singh, G.S. Josan, Classification and prediction based data mining algorithms to predict slow learners in education sector, Proc. Comput. Sci. 57 (2015) 500–508, https://doi.org/10.1016/j.procs.2015.07.372.

[64] K. Deepa, G. Radhamani, P. Vinod, Investigation of feature selection methods for Android malware analysis, Proc. Comput. Sci. 46 (2015) 841–848, https://doi.org/10.1016/j.procs.2015.02.153.

[65] V. Kumar, S. Minz, Feature selection: a literature review, Smart Comput. Rev. 4 (2014) 211–229, https://doi.org/10.6029/smartcr.2014.03.007.
[66] A. Developer, Android Permission, Android Permission, 2016. <https://developer.android.com/guide/topics/security/permissions.html> (accessed September 2, 2016).
[67] Shahid Farid, R. Ahmad, Mujahid Alam, A hierarchical model for e-learning implementation challenges using AHP, Malaysian J. Comput. Sci. 28 (2015) 166–188.
[68] F. Dweiri, S. Kumar, S.A. Khan, V. Jain, S. Ahmed, V. Jain, Designing an integrated AHP based decision support system for supplier selection in automotive industry, Expert Syst. Appl. 62 (2016) 273–283, https://doi.org/10.1016/j.eswa.2016.06.030.
[69] T.L. Saaty, Decision making with the analytic hierarchy process, Int. J. Services Sci. 1 (2008) 83, https://doi.org/10.1504/IJSSCI.2008.017590.
[70] M.F.A. Razak, N.B. Anuar, F. Othman, A. Firdaus, F. Afifi, R. Salleh, Bio-inspired for features optimization and malware detection, Arabian J. Sci. Eng. (2017), https://doi.org/10.1007/s13369-017-2951-y.
[71] A. Firdaus, N.B. Anuar, M.F.A. Razak, A.K. Sangaiah, Bio-inspired computational paradigm for feature investigation and malware detection: interactive analytics, Multimedia Tools Appl. (2017), https://doi.org/10.1007/s11042-017-4586-0.
[72] F.A. Narudin, A. Feizollah, N.B. Anuar, A. Gani, Evaluation of machine learning classifiers for mobile malware detection, Soft Comput. 20 (2016) 343–357, https://doi.org/10.1007/s00500-014-1511-6.
[73] Z. Yuan, Y. Lu, Y. Xue, Droiddetector: Android malware characterization and detection using deep learning, Tsinghua Sci. Technol. 21 (2016) 114–123, https://doi.org/10.1109/TST.2016.7399288.
[74] V. Total, Analyzes suspicious files and URLs, Virus Total, 2016. <https://www.virustotal.com/en/#dlg-join> (accessed September 8, 2016).
[75] F. Idrees, M. Rajarajan, M. Conti, T.M. Chen, Y. Rahulamathavan, PIndroid: a novel Android malware detection system using ensemble learning methods, Comput. Secur. 68 (2017) 36–46, https://doi.org/10.1016/j.cose.2017.03.011.
[76] Vanja Svajcer, Plankton malware drifts into Android Market, SophosLab, 2011. <https://nakedsecurity.sophos.com/2011/06/14/plankton-malware-drifts-into-android-market/> (accessed September 21, 2016).
[77] N.B. Anuar, M. Papadaki, S. Furnell, N. Clarke, A response selection model for intrusion response systems: Response Strategy Model (RSM), Secur. Commun. Networks 7 (2013) 71–81, https://doi.org/10.1002/sec.
[78] National Institute of Standards and Technology Gaithersburg, Guide for conducting risk assessments, NIST Special Publication, 2012, p. 95. http://doi.org/10.6028/NIST.SP.800-30r1.
[79] N.B. Anuar, M. Papadaki, S. Furnell, N. Clarke, Incident prioritisation using analytic hierarchy process (AHP): Risk Index Model (RIM), Secur. Commun. Networks 6 (2013) 1087–1116, https://doi.org/10.1002/sec.673.
[80] J. Westenberg, Facebook now allowing Google to index its mobile app, Android Authority, 2015. <http://www.androidauthority.com/facebook-allowing-google-to-index-its-mobile-app-655958/> (accessed April 4, 2017).
[81] K. Tam, A. Feizollah, N.B. Anuar, R. Salleh, L. Cavallaro, The evolution of Android malware and Android analysis techniques, ACM Comput. Surv. 49 (2017) 1–41, https://doi.org/10.1145/3017427.
[82] Université du Luxembourg, AndroZoo, 2018. <https://androzoo.uni.lu/> (accessed July 3, 2018).
[83] P. Faruki, V. Ganmoor, V. Laxmi, M.S. Gaur, A. Bharmal, AndroSimilar : robust statistical feature signature for Android malware detection, in: Proceedings of the 6th International Conference on Security of Information and Networks, ACM, 2013, pp. 152–159.
[84] A. Firdaus, N.B. Anuar, A. Karim, M.F.A. Razak, Discovering optimal features using static analysis and genetic search based method for Android malware detection, Front. Inf. Technol. Electron. Eng. (2017) 1–27, https://doi.org/10.1631/FITEE.1601491.
[85] B. Rashidi, C. Fung, E. Bertino, Android resource usage risk assessment using hidden Markov model and online learning, Comput. Secur. 65 (2017) 90–107, https://doi.org/10.1016/j.cose.2016.11.006.
[86] A. Feizollah, N.B. Anuar, R. Salleh, G. Suarez-Tangil, S. Furnell, Androdialysis: analysis of Android intent effectiveness in malware detection, Comput. Secur. 65 (2017) 121–134, https://doi.org/10.1016/j.cose.2016.11.007.
[87] S. Learning, F. Orphan, A. Problemsoftware, A. Recovery, M. Bibi, O. Maqbool, J. Kanwal, Supervised learning for orphan adoption problem in software architecture recovery, Malaysian J. Comput. Sci. 29 (2016) 287–313.