Permission-Based Feature Selection for Android Malware Detection and Analysis

Chit La Pyae Myo Hein

Faculty of Computer System and Technologies, University of Computer Studies Taunggyi

chitlapyae86gmail.com

***Abstract* — Malware are spreading around the world and infecting not only for end users but also for large organizations and service providers. There is a real need of dimension reduction approach of malware features for better detection. This system describes for malware detection and characterization framework which is based on Static Approach by only analyzing the Manifest File of android application. This system also describes a Feature Selection Approach which is also based on Manifest File Analysis for the purpose of dimension reducing of malware features. Firstly, a number of Permission-Based Features are extracted by disassembling the Manifest File of Android application. Then, feature dimensions are reduced by proposed Score-based Approach. The results getting from Correlation and Information Gain are used to compare the results of Score-Based Features Selection. According to the experimental results, proposed a light-weight approach can perform as equal as other feature selection methods. After feature selection, manifest file analysis based on malware classification and characterization results are also described in this system. The classification results tested by without reducing features and the results obtained by reduced features are compared to determine which methods or classifiers are the best to detect malware.**

Keywords*— *Android Security, Malware,Smartphone

I. INTRODUCTION

In the past few years, Smartphone users have increased exponentially. The various Smartphone age ranges of products from Nokia, Apple, Google, Blackberry, etc. The operating systems for Smartphone are Symbian, iOS, Android and Blackberry. The Smartphone is viewed as portable PCs as they have all the functionalities of a desktop PC integrated into them. Just as there are hackers/attackers releasing malware for PCs, there are attackers who are now targeting Smartphone. The main reason for this is that mobile security is still in its initial stages and the lack of user awareness regarding how their devices can be undermined if they are not careful enough.

Google is open-source operating systems. Android, are among the most popular Smartphone operating systems. Android is a Linux-based operating system that also includes key applications and middleware. In order to fully benefit from and explore the functionalities of Android, Google allows third party developers to create applications and release it to the Android Market.

Android Market is one application that is mounted on the device that enables a user to browse and download several paid and free applications. It is the same as the AppStore for iPhone. Developers will have to sign their code and test it thoroughly to make sure it is functioning properly without causing any kind of harm to the user and they then release it on the Android Market.It is however possible for attackers to release malware on Android Market. Google is currently making a success at cleaning the market and making it free from malware. However, attackers can create malware or patches for existing applications that once installed make the application behave as a malware or they can simply take an existing application and disassemble it, alter the code to enable it functions abnormally, and repackage the application.

Malware on Android have been huge in number and attackers are constantly discovering newer methods to crack into the devices. The main reason for this is because Smartphone like Android do not just use as a portable telephone these days. Android devices can access the internet, make online bank transmissions,

manage social networks, etc. All these functionalities of a mobile phone seem very tempting for an attacker to obtain information about the user and use it to his/her benefit.

This research purpose to develop malware detection of static based on deriving of permission request using the manifest file of the application. The static approach provides human understandable and explainable terms, which do not prescribe additional post processing. Furthermore, in a court of law a judge and a jury may understand the reasoning behind the extracted terms, which are very important under computational forensics investigation of numerical evidences.

II. LITERATURE REVIEW

Many researchers propose complicated extensions to fortify the Android's security framework. They mainly focus on protecting the user data and mitigating some types of privilege escalation attacks. This section recent some of the most well-known approaches to extract the malware list in android technologies.[11] Alternative research has focused on using machine learning techniques to identify malware. Sanz et al. (2012) applied several types of classifiers to the permissions, ratings, and static strings of 820 applications to see if they could predict application categories. They applied this by using the category scenario as a stand-in for malware detection. [4] Shabtai et al. (2013) similarly built a classifier for Android games and tools, as a proxy for malware detection. [15] Ryo Sato, Daiki Chiba and Shigeki Goto proposes a new method for detecting Android malware by analyzing only manifest files based on malware score. [13] Zhou et al. (2012) found real malware in the wild with DroidRanger, a malware detection system that uses permissions as one input. [14] DroidMat (2013) focuses on using attributes of the manifest to trace API calls requiring permissions. [12] N. Peiravian and X. Zhu (2013) propose a rule-based security mechanism designed to prevent malware at install-time. [1] Explores the use of machine learning algorithms for malware detection using permissions and API calls. The studies in Mila (2009) and android (2015) Android Content License "URL" www.source.android.com /license.html retrieved focus on efficient, scalable, and accurate malware detection in large Android markets.

III. MALWARE DETECTION

A. Proposed Malware Detection FrameWork

The first purpose of malware system is to reduce the detecting and classification for malware by introducing the features selection and extraction step in the process. The second purpose is to classify and characterize the malware by only taking the manifest file analysis in opposition to an existing machine learning approach.

The cost of analysis and risk for detecting malware can decrease by means of static approach rather than a dynamic approach.

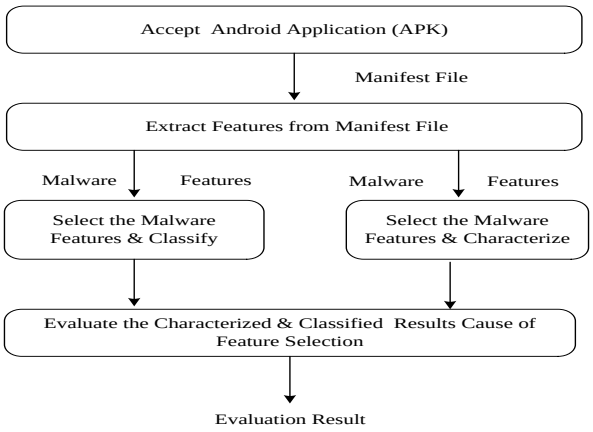

Figure 1. System flow diagram of the proposed system

Therefore, this system is also based on static (code-based) approach. The components of this system are as follows:

(i) Android Application File Accessing Component
(ii) Feature Selection Components
(iii) Malware Detecting Component
(iv) Malware Classification Component
(v) Malware Characterization Components

The system flow of the whole proposed system is illustrated in Figure 1. The feature selection methods follow the *Feature Ranking* approach and, using a specific metric to compute the rank and return a weight average value for each feature individually. By using this attribute selection method, the system can select generic features to merge the relevant and meaningful features for to input the system.

The second part of the system is to produce the feasible set of features. To produce this set of features, merge common features and based features of each detection type. The six parts of the system are a classification.

In this classification step, BayesNet (BN), Naïve Bayes, Multilayer Perceptron (MLP), K-Nearest Neighbor, J48 and Random Forest classifier are utilized to classify the detection types. Finally, the system can prove the performances of the proposed selecting features are higher than unselected features.

1. Data Collection

In this system used different methods for retrieving the application samples from their respective websites, as well as retrieving information from the malicious applications.

2. Benign Dataset

The gathering of the market data set is get from four separate application markets, consisting of multiple features, ranging from developer identity to request permissions. In order to obtain as many application samples as possible, four Android application markets were chosen; Google Play [43] /AppBrain [58], Amazon App Store for Android, F-Droid [62] and SlideMe [78] as shown in Table I.

TABLE I
NUMBERS OF BENIGN FOR EACH CATEGORY

All Categories	Number of Application
Development	35
Phone & SMS	38
Wallpaper	50
Office	51
Science & Education	81
Multimedia	174
System	67
Games	165
Internet	149
Security	25
Reading	51
Navigation	112
Children	2
Total	**1000**

3. Malware Dataset

The experimental study of this work is primarily taken from several sources of malicious data set. Numerous researchers propose complicated extensions to fortify the Android's security framework. The candidates included Symantec's Threat Explorer database, F-Secures Threat Description database [31] and similar sources, in addition to the Contagio Mobile Dump [74]. Databases of Symantec and F-Secure were ultimately decided against because it was impractical to automatically collect information from these databases. Additionally, the technical details were produced by hand by the researchers and as such the information was inconsistent as to whether or not they listed the permissions requested by the malware.

Cantagio collects and presents samples of malicious applications uploaded to the website by the public, and anyone can download these samples from their database as showed in Table II.

TABLE II
NUMBERS OF MALWARE FOR EACH CATEGORY

All Categories	Number of Application
TROJAN	171
ESCOFPRIV	199
PREMSMS	222
INFOSTEAL	189
Root Exploit	200
Total	**981**

B. Proposed Score-Based Feature Selection

This system also proposes a feature selection method based on manifest file analysis approach. Process flow of the propose method is described in Figure 2.

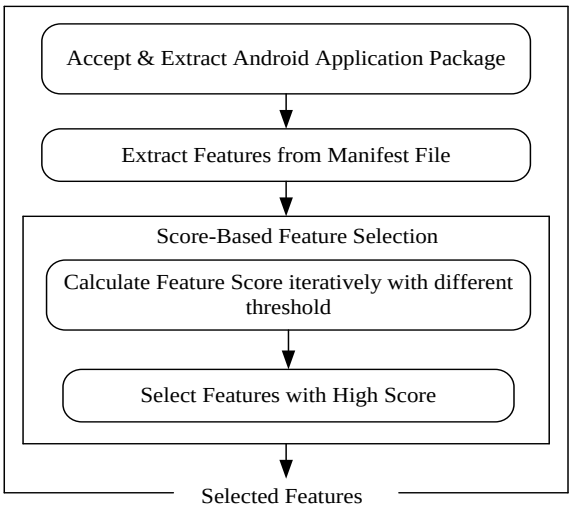

Figure 2. FLOW OF THE PROPOSED SCORE-BASED FEATURE SELECTION

The nature of mobile android application (APK) files, how to extract the features from mobile applications is described in this section. The detail explanation of how to process the proposed score-based feature selection is also referred to in the section below.

Firstly, I extracted the necessary features to analyze from sample applications (benign and malware). Then, I built dataset in (CSV) comma separated values file format from the extracted features. In this system used these two datasets to distinguish malware and benign applications by machine learning approaches. Each record comprises of the summary data of 134 permission features. Experts in malware detection labeled the dataset information as either 'Ben' or 'Mlw.' The labeling process made use of a malware dataset considered 'Mlw' and a benign dataset categorized 'Ben.'

Finally, the ultimate dataset meant the integration of malware with normal datasets. Then, one of the features used in the Weka pre-processor weka.filters. unsupervised. inatance. randomize shuffled the records in the final dataset.

To obtain features set for samples present in data set; the system used a Java based Android package profiling tool for automated reverse engineering of the .apk files named as Apk Analyser. This tool unpacks and decompiles the input .apk files to corresponding .dex and AndroidManifest.xml files. After doing reverse engineering, a set of detectors were applied to the reverse engineered .apk files to detect properties used to build the profile for APK file. The feature vector obtained after property detection contains values for selected features as binary numbers (0 and 1), which is a suite of comma separated values. Let an application characteristic TA obtained from the ApkAnalyser detector is defined by a random variable:

$$TA = \begin{cases} \mathbf{1} \text{ if discovered by the catches sample} \\ \mathbf{0} \text{ Otherwise} \end{cases}$$

To generate the dataset, the first selected the samples. Initially, the system collected 400 samples. Next, the system normalized the values given by different antivirus vendors. The goal of this step was to ascertain their reliability detecting malware in Android. To this end, the system assumed that every sample that was detected as malware by at least one antivirus was, indeed, malware. Then, the system evaluates the request rate of scoring each malware sample with respect to the complete malware dataset.

$$SB_i = \frac{RP_i}{TA}$$

Where RP_i is the number of samples detected by i-th malware and TA is the total number of application samples.

C. Feature Selection of With and Without

Actually, the count 134 android system permissions according to android 4.3 Jelly Bean with API level-18 considering all of android permissions as a feature set will produce an enormous feature vector for each application. So, it is required to reduce the number of the application features, where the high dimension data makes testing and training of general classification methods complicated.

The goal of data reduction is to find a minimum set of features such that the resulting probability distribution of the data classes is as close as possible to the original distribution obtained using all features. Using the reduced set of features has additional benefits. It reduces the number of features appearing in the discovered patterns, helping to make the patterns easier to be understood.

D. Characterization

Categorizing and classifying mobile applications according to their potential for privacy invasion provides detailed information about what is being put at risk by installing and agreeing to various permission and privilege requests by mobile applications. This categorization is based on the permissions requested by an application as detail description in below.

The Cantagio samples in our experiment comprised 12 different families as shown in Table III containing 1000 Android malware samples, but only 250 were used. The machine learning process had three phases: (1) data collection, which captured permission; (2) feature selection and extraction; and (3) the machine learning classifier. For the normal dataset, Application are selected the top 20 free applications from Google Play as express in Figure 3.

TABLE III
CATEGORIES OF PERMISSION CHARACTERIZATION

Number	Description
1	Permissions for *sdcard interaction*
2	Permissions for *things that cost money*
3	Permissions *associated with telephony state*
4	Permissions for *special development tools*
5	Permissions for *accessing accounts*
6	Permissions for *accessing messages*
7	Permissions for *accessing location info*
8	Permissions for *accessing hardware*
9	Permissions for *accessing networks*
10	Permissions for *accessing personal info (contacts and calendar)*
11	Permissions for *low-level system interaction*
12	*Private (signature-only)*

Figure 4 shows the top 20 permissions requested and required by both malicious and benign applications. Compare the results against their statistics, the top three requested permissions are the same. For malicious applications, the top three requested permissions are INTERNET, READ CONTACTS, and ACCESS NETWORK STATE. For benign applications, the top three requested permissions are CALL_PHONE,CHANGE_WIFI_STATE,and READ_PHONE_STATE. Although the number of malicious application, the system evaluated is less than, the ranks of requested permissions are similar.

Figure 4 show the value of malware applications requesting certain number of permission, respectively. It is shown that the best result in Escofpriv of malware application requesting certain number of Risky3 permission. It can be easily seen that lowest level of Escofpriv malware application at Risky1. Compare the categories of Risky2 level as the almost same level of Risky3 malware detection characterization. Trajon, InfoSt, PreSMS and RootEx malware characterization are the lowest value of permission in usage level.

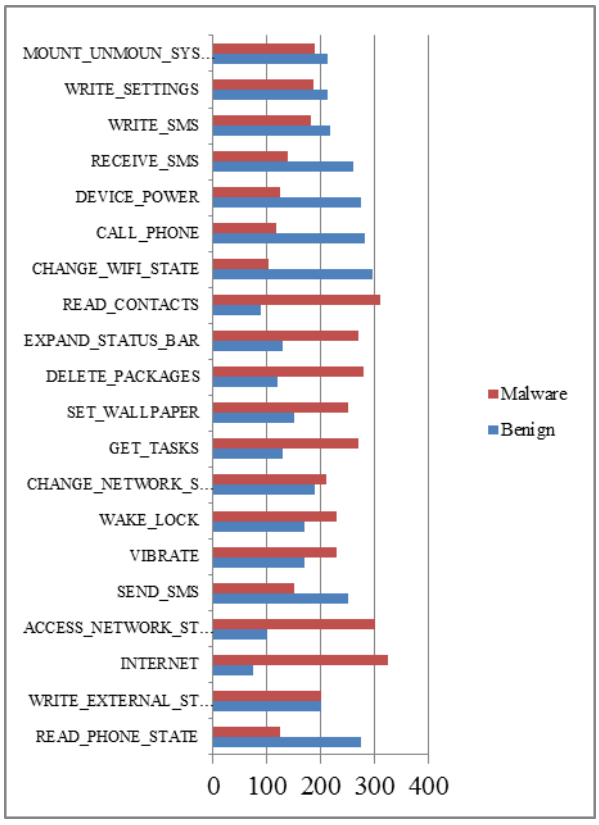

Figure 3. TOP MOST REQUEST PERMISSIONS FROM THE APPLICATIONS

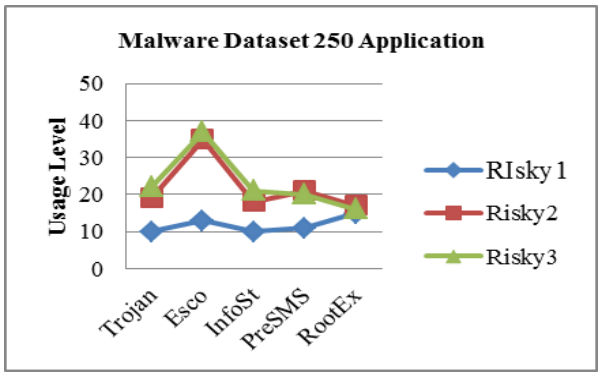

Figure 4. Compare the accuracy result using 250 applications

Catagorization of Risky Permission

Permissions have different danger levels depending on the functions they allow the application to perform and are consequently classified in protection level groups. Likewise, through this attribute, it is possible to determine which applications have access to the permission:

Risk1: They pose a risky1 factor and typically only affect the application's scope. Risk1 permissions are granted by the system automatically without explicit approval of the user.

Risk2: They are risky2 permissions that allow costly access to services. The permissions can be granted by the user during installation. If the permission request is denied, then the application is not installed.

Risk3: They are risky3 permissions are only granted if the requesting application is signed by the same developer that defined the permission. Risk3 permissions are useful for restricting component access to a small set of applications trusted and controlled by the developer.

Identification of Risky Permission

TROJAN pattern: ACCESS COARSE LOCATION, ACCESS FINE LOCATION, CALL PHONE, INTERNET, MOUNT UNMOUNT FILESYSTEMS, READ CONTACTS, READ PHONE STATE, SEND SMS, SET WALLPAPER, WRITE CONTACTS, WRITE EXTERNAL STORAGE.

The Trojan requests a rather distinct set of permissions, many of which are not often requested by legitimate applications. Using this pattern on the data set resulted in no hits from the legitimate markets, only the malicious data set.

ESCOFPRIV pattern: INTERNET, ACCESS NETWORK STATE, READ PHONE STATE, ACCESS WIFI STATE, WRITE EXTERNAL STORAGE, ACCESS COARSE LOCATION, ACCESS FINE LOCATION, RECEIVE SMS, SEND SMS, READ SMS, CALL PHONE, PROCESS OUTGOING CALLS, DELETE PACKAGES, INSTALL PACKAGES, RECEIVE BOOT COMPLETED.

This pattern returns only the Escofpriv samples malware from the malicious data set. Like the Trojan pattern, this pattern can almost uniquely identify Escofpriv infected applications.

PREMSMS pattern: SEND SMS, READ SMS, WRITE SMS, RECEIVE SMS, DEVICE POWER, WRITE APN SETTINGS, ACCESS NETWORK STATE, BROADCAST

PACKAGE REMOVED, ACCESS WIFI STATE, CHANGE WIFI STATE, WAKE LOCK, INTERNET, WRITE EXTERNAL STORAGE, READ PHONE STATE, KILL BACKGROUND PROCESSES.

PremSMS effectively means that the malware can be uniquely identified based only on its permission set, this pattern is effective in determining the presence of this malware.

INFOSTEAL pattern: READ CALENDAR, READ CONTACTS, READ USER DICTIONARY, WRITE CALENDAR, WRITE-CONTACTS, WRITE USER DICTIONARY, SET ALARM, READ HISTORY BOOKMARKS, and WRITE HISTORY BOOKMARKS.

Infosteal pattern READ CONTACT PERMISSION, which means that it is install application: Allows an application to read the user's contacts data.

ROOT EXPLOIT pattern: ACCESS NETWORK STATE, ACCESS WIFI STATE, BLUETOOTH, INTERNET, NFC, USE SIP, ACCOUNT MANAGER.

Using this pattern against the data set resulted in 981 authorize applications identified as Root exploit.

TABLE IV
IDENTIFICATION OF RISKY ASSESSMENT

Identification Grate	Permission request
R1	ACCESS_COARSE_LOCATION,ACCESS_FINE_LOCATION,ACCESS_NETWORK_STATE,ACCESS_WIFI_STATE,BLUETOOTH_ADMIN,GET_TASK,READ_CALENDAR,READ_HISTORY_BOOKMARKS,READ_LOGS, READ_USER_DICTIONARY,RECEIVE_WAP_PUSH,SUBSCRIBED_FEEDS_READ
R2	CAMERA_PROCESS_OUTGOING,CALLS,READ_CALL_LOG,READ_CONTACTS,READ_EXTERNAL_STORAGE,READ_SMS,READ_SOCIAL_STREAM.RECEIVE_MMS,RECEIVE_SMS,RECORD_AUDIO,WRITE_EXTERNAL_STORAGE
R3	AUTHENTICATE_ACCOUNTS,GET_ACCOUNTS,USE_CREDENTIALS

Table IV shows the requested permissions in the benign application and malware datasets. Risk level of identification grate R2: CAMERA is the most frequently used permission by both the benign applications and malware. There are many reasons to request permission for picture access: some of the applications need to log in; some are designed to use internet like browsers and email clients; some need to load advertisement etc. As a result, Camera-related permissions, such as ACCESS NETWORK STATE and ACCESS WIFI STATE, become very popular. Another set of widely used permissions are location related ones such as ACCESS FINE LOCATION and ACCESS COARSE LOCATION for location based services as defined on grate level of R1, etc.

Another observation is that some applications of permissions are requested by such malware applications in the Cantagio dataset. Malware are more favor of changing the settings and use money-related services such as short message service (SMS). Changing settings, especially changing the network settings, generally is the first step before a malware performs any malicious activity. Sometimes malware even try to kill background processes, which could help them avoid being detected by anti-virus applications. Characterization system can see that the usage pattern of SMS related permissions is quite different between the benign applications and the malware applications and many malware applications attempt to request SMS related permissions. SMS is also a risky permission of private threat (Risk3) that is more likely requested by malware applications. Describe the data usage of SMS include the data category of communication as shown in Table V.

TABLE V
IDENTIFICATION OF RISKY ASSESSMENT

Data Category	Data Usage	Permission Request	Private Threat		
			R 1	R 2	R 3
Sensor/Location	Location Audio Video	ACCESS_COARSE_LOCATION	☹		
		ACCESS_FINE_LOCATION	😐		
		BLUETOOTH_ADMIN	😐		
		ACCESS_NETWORK_STATE	😐		
		ACCESS_WIFI_STATE	😐		
		READ_SOCIAL_STREAM	😐		
		RECORD_AUDIO	😐		
		CAMERA	😐		
External Storage		WRITE_EXTENAL_STORAGE		😐	
		READ_EXTERNAL_STORAGE		😐	
Communication	SMS MMS Voice Wap Push	RECEIVE_SMS			😐
		READ_SMS			😐
		RECEIVE_MMS			😐
		PROCESS_OUTGOING_CALLS			😐
		RECEIVE_WAP_PUSH			😐

IV. CONCLUSIONS

Mobile malware performs malicious activities like stealing confidential information, sending messages, SMS, reading contacts and can even harm by exploiting the data. Malware is spreading around the world and infecting not only for end users, but also for large organizations and service providers. Malware classification is a vital component and works together with malware identification to prepare the right and effective malware antidote.

In this research, malware classification and analysis have been used to determine whether a program has malicious intent or not. In this study, collected Android applications have been classified using machine learning approaches whether they are malware or benign. Static approach has been used to classify and detect malware. Several permission features from several manifest files have been extracted. A score-based feature selection approaches, which is only based on manifest file analysis have been proposed and evaluated as a lightweight approach for malware detection. And then, the selected malware was detected using different classifier. According to the experimented consequences, the proposed score-based feature selection has been performed similarly with existing feature selection methods. (Correlation-based and Information Gain). Moreover, by using static-based malware approach, it is more efficient and adaptable because the static approach has the advantages of less cost rather than a dynamic approach.

Therefore, the proposed approach using permissions is effective for malware detection which achieved an average rate of malware detection accuracy. Not only malware classification, but also malware characterization is also important to inform the user and install the malware application, because the user is not aware to install several applications of their device. The Android application requires several permissions to work. An essential step to install an Android application into a mobile device is to allow all permissions requested by the application.

In this work, malware detection system has been provided a systematic study on the exploration of permission-induced risk in Android apps on a large-scale in three levels. First, the research has been focused on ranking all the individual permissions request by using three methods. Second, the research has been identified the subsets of top most permissions with sequential forward selection as well as with score-based. And then, the evaluation of this system has been employed several algorithms, BN, MLP, J48, KNN and RF, to detect malware applications based on the identified subsets of exploit permissions. The design also constructs top most demand sets with feature ranking to detect malware applications with different characteristics. The large official application data set consisting of 1000 benign applications and 981 malware applications, as well as a third-party application set have been used for the evaluation.

The focus of this research is tantamount to design a simple and easy-to-evaluate framework for analyzing mobile privacy. Categorizing and classifying mobile applications according to their potential for privacy invasion provides detailed information about what is placed at risk by installing and agreeing to various permission and privilege requests by mobile applications. This categorization has been based on the permissions requested by an application. Malware detection has been discussed and analyzed in depth the effectiveness as well as the limitations on the detection of malware applications with only permission requests. While the permission requests characterized the behaviors of apps to certain extent and the detection can be effective, only considering the permissions would have difficulties to improve the current detection accuracy, as the permission vectors are very sparse and binary number type. The results obtained from a user training test using five selected classifiers to perform the experiments have been presented in **Appendix A**. The table showed the performance of each classifier in ten (100 and 1000 applications) experiment sets for malware detection. Classifier performance has been needed to be measured with five evaluation metrics, namely TPR, FPR, precision, recall and f-measure.

The results of the average error rates and accuracy for different feature sets with and without category information have been compared. The description has been observed that an increasing accuracy and decreasing error rates when larger numbers of features are utilized to train the classifier. It was also clear that by exploiting the category information; there was a clear improvement in the accuracy and error rates. Also, note that there was almost a difference of 86%, 95% in the performance using with and without feature sets indicating the importance of feature ranking based on the frequency of permission request.

In the experiments, the system has been used all the application data for permission ranking. In the study, Information Gain, Correlation-based as well as score-based methods have been used for ranking the permissions which contained the malware and benign features. The ranking results for the top FSI, FSII and FSIII permissions were presented in Appendix A and 15 permission top most request FSI was the lowest ranked features. It has been observed that Information Gain and Correlation-based produced the same top most requests set, although the ranking order for the first feature permissions was different. The number of intersections of the top request permissions that have been generated by Correlation-based and Information Gain have been nearly consistent with the ranking results. The only one difference permission of score-based in the ranking results was set in boldface.

According to the classification results, the classification of top most requests FSI using 1000 applications has been demonstrated that the classifier bayes network has been produced a higher FPR result with 0.39% compared to the k-nearest neighbor in 0.112%. This has been indicated that the bayes network was less effective than another selected classifier for malware detection.

From the classification of top most requests FSII using 1000 samples in the current result, the J48 classifier has been achieved 87% and RF has been achieved 92% detection rate accuracy. Whereas the highest detection rate attained in this research was 87% with the KNN classifier. Therefore, the observed results indicate the comprehensiveness and efficiency of this study**.**

V. FUTURE WORK

The current work discusses and analyzes in depth the effectiveness as well as the limitations on the detection of malware applications with only permission requests. While the permission requests characterize the behaviors of applications to certain extent and the detection can be effective, only considering the permissions would have difficulties to improve the current detection accuracy, as the permission

vectors are very sparse and binary number. In the future work, there are exploring more relevant features that inherit in application in order to improve the detection accuracy of Android malware application.

ACKNOWLEDGMENT

First of all, I would like to thank Union Minister of Science and Technology for full facilities support during the Ph.D. Course at the University of Computer Studies, Mandalay.

I would like to thank a lot to all my teachers for their mentoring, encouragement, and recommending the research.

REFERENCES

[1] K. Allix, T. F. D. A. Bissyande, J. Klein, and Y. Le Traon, "Machine Learning-Based Malware Detection for Android Applications: History Matters!," 2014.

[2] D. Arp, M. Spreitzenbarth, M. H¨ubner, H. Gascon, K. Rieck, and C. Siemens, "DREBIN: Effective and Explainable Detection of Android Malware in Your Pocket," 2014.

[3] Z. Aung and W. Zaw, "Permission-based android malware detection," International Journal Of Scientific & Technology Research, vol. 2, no. 3, 2013.

[4] G. Canfora, F. Mercaldo, and C. A. Visaggio, "A classifier of malicious android applications," in Availability, Reliability and Security (ARES), 2013 Eighth International Conference on, pp. 607–614, IEEE, 2013.

[5] W. Enck, M. Ongtang, and P. McDaniel, "On lightweight mobile phone application certification," in Proceedings of the 16th ACM conference on Computer and communications security, pp. 235–245, ACM, 2009.

[6] L. Gomez and I. Neamtiu, "A Characterization of Malicious Android Applications," tech. rep., Technical report, University of California, Riverside, 2011.

[7] M. Grace, Y. Zhou, Q. Zhang, S. Zou, and X. Jiang, "Riskranker: scalable and accurate zero-day android malware detection," in Proceedings of the 10th international conference on Mobile systems, applications, and services, pp. 281–294, ACM, 2012

[8] C.-Y. Huang, Y.-T. Tsai, and C.-H. Hsu, "Performance Evaluation on Permission-Based Detection for Android Malware," in Advances in Intelligent Systems and Applications-Springer, 2013.Volume 2, pp. 111–120,

[9] T.M. Mitchell. Machine Learning. McGraw-Hill Series in Computer Science.McGraw-Hill, 1997.

[10] C. Orthacker, P. Teufl, S. Kraxberger, G. Lackner, M. Gissing, A. Marsalek, J. Leibetseder, and O. Prevenhueber, "Android security permissions–can we trust them?," in Security and Privacy in Mobile Information and Communication Systems, pp. 40–51, Springer, 2012.

[11] B. Sanz, I. Santos, C. Laorden, X. Ugarte-Pedrero, and P. G. Bringas, "On the automatic categorization of android applications," in Consumer Communications and Networking Conference (CCNC), 2012 IEEE, pp. 149–153, IEEE, 2012.

[12] B. Sanz, I. Santos, C. Laorden, X. Ugarte-Pedrero, J. Nieves, P. G. Bringas, and G. A´ lvarez Maran˜o´n,"MAMA: Manifest Analysis for Malware Detection in Android," Cybernetics and Systems, vol. 44,no. 6-7, pp. 469–488, 2013

[13] B. Sanz, I. Santos, C. Laorden, X. Ugarte-Pedrero, P. G. Bringas, and G. A´ lvarez, "Puma: Permission usage to detect malware in android," in International Joint Conference CISIS12-ICEUTE´ 12-SOCO´12 Special Sessions, pp. 289–298, Springer, 2013.

[14] B.P. Sarma, N. Li, C. Gates, R. Potharaju, C. Nita-Rotaru, and I. Molloy, "Android permissions: a perspective combining risks and benefits," in Proceedings of the 17th ACM symposium on Access Control Models and Technologies, pp. 13–22, ACM, 2012.

[15] R. Sato, D. Chiba, and S. Goto, "Detecting Android Malware by Analyzing Manifest Files," Proceedings of the Asia-Pacific Advanced Network, vol. 36, pp. 23–31, 2013.

[16] Google Inc. Android Developers. Manifest. permission. http://developer.android.com/reference/android/Manifest.permission.html.

[17] Google Inc. Android Developers. Security and Permissions http://developer.android.com/guide/topics/security/security.html.

[18] Permission>Android Developer - API Guides – Android Manifest.http://developer.android.com/guide/topics/manifest/permission-element.html

[19] Cantagio mobile malware-mini-dump http://contagiominidumo.blogspot.com