An Android malware detection system based on machine learning

Cite as: AIP Conference Proceedings **1864**, 020136 (2017); https://doi.org/10.1063/1.4992953
Published Online: 03 August 2017

Long Wen, and Haiyang Yu

An Android Malware Detection System Based on Machine Learning

Long Wen [a] and Haiyang Yu [b]

Faculty of Information Technology, Beijing University of Technology, Beijing 100049, China.

[a] Corresponding author: wenwenwr888@163.com
[b] billyukiwi@163.com

Abstract. The Android smartphone, with its open source character and excellent performance, has attracted many users. However, the convenience of the Android platform also has motivated the development of malware. The traditional method which detects the malware based on the signature is unable to detect unknown applications. The article proposes a machine learning-based lightweight system that is capable of identifying malware on Android devices. In this system we extract features based on the static analysis and the dynamitic analysis, then a new feature selection approach based on principle component analysis (PCA) and relief are presented in the article to decrease the dimensions of the features. After that, a model will be constructed with support vector machine (SVM) for classification. Experimental results show that our system provides an effective method in Android malware detection.

Key words: Static Analysis; Dynamitic Analysis; Relief; PCA; Feature Selection; Support Vector Machine.

INTRODUCTION

Recent years, with the rapid development of smartphones, Android is becoming more and more popular. According to the IDC report [1] shown in the 2016 year, Android market share will reach 85.3% until the end of 2016. Android has been more and more indispensable with its open source character and advantages of free in our daily life. However, the number of malicious software is also growing rapidly. Therefore, how to detect the Android malware with the high accurate rate is a hot issue.

Traditional detection approach based on signature is widely used both in Android devices and PC platform by extracting the signature from APK and comparing with the malicious signature in the virus database, however, this approach is limited to detect unknown malwares which are not existed in the virus database. In order to address this question, the previous researchers find that there are two techniques for the unknown Android malware analysis: static analysis and dynamic analysis [2]. Static analysis, occurs before the Android application is installed, is a technique based on checking the contents of the APK [3] by means of reverse engineering. Different from the static analysis, dynamic analysis monitors the running state of the Android application in the virtual environment. With the development of machine learning technology, machine learning approaches are also mentioned to classify obtained observations as either benign or malicious by collecting different signal features and events from the applications and the system, however the raw features may exist irrelevant or redundant features that may lead to a wrong result, so feature selection process is vital. Feature extraction and feature selection are the crucial steps which determine the accuracy rate of the classifier. After that it will be easier to achieve the detection result with the classifier.

Motivated by above observations, we propose a detection system for Android platform based on the machine learning classifier named SVM. Our system pays attention to the step of feature extraction and feature selection, we combine static analysis and dynamic analysis to obtain the Android application's features, and then a new feature selection algorithm named PCA-RELIEF is proposed to find the most discriminating feature subset. Experimental results show that our method can accurately identify Android malware.

Green Energy and Sustainable Development I
AIP Conf. Proc. 1864, 020136-1–020136-7; doi: 10.1063/1.4992953
Published by AIP Publishing. 978-0-7354-1542-3/$30.00

The rest of the paper is organized as follows. Section 2 present a survey of related work in Android malware detection. In Section 3, we present our detection system in details, including the process of feature extraction and the new feature selection approach. The experiment and evaluation are presented in Section 4. Finally, we conclude the task in Section 5.

RELATED WORK

As is shown in the introduction module, there are two mainly detection approaches in Android detection, which can be described as static analysis and dynamic analysis.

Static analysis, decompiles APK before application is installed, can provide a fast and safe result for Android detection. Researchers usually choose the data flow tracking or the relevant attribute information from the APK to distinguish malware. For instance, Qin [4] analyzes the dangerous permission and stores them in the database, then they extract the permissions from the unknown Android application, after comparing with the dangerous permission, they could get a conclusion. Felt [5] proposes a tool named Stowaway to detect the over-privileged applications by API calls. Yang [6] detects the leakage of sensitive information on Android with static taint analysis. However, the static analysis has a limit to analyze the obfuscated application [7], so the result may be incorrect if application is encrypted.

Dynamic analysis mainly obtains the features when the Android application is running, by monitor the behaviors or the state of the sensitive data, the malware can be found out. Qiao [8] presents a framework named CBM which extracts the API call sequences by dynamic behavior analysis tool, Tam [9] also develops an automatic dynamic analysis system based on VMI to identify malware according to the dynamic behavior. However dynamic detection methods often cannot sure when and how to trigger all the malicious behaviors [10], what's more, dynamic analysis requires a lot of time to analyze the application which is not suitable for the smartphone.

Qiao [11] proposes a machine learning approach to detect malware by mining the patterns of permission and API function calls acquired and used by Android apps, they establish the contact between the permission and API, however, the malicious samples they collected are not enough. Munoz [12] obtains the meta-data from Google Play, then they regarded the meta-data as the features to apply a machine learning approach to identify the highest predictive features and detection malware. Similarly, Peiravian [13] also proposes a feature-based learning framework which focused on the requested permissions and API calls behaviors and applies the SVM, Decision Tree and Bagging algorithms. But they only extract permission and API as the feature which lead to a low accuracy, they extract few types of features.

Feature selection algorithms, a method to dimension reduction by finding the best minimum subset of the raw features are as vital as feature extraction in the machine learning. Chi-Square, relief and information gain are the most popular filter approach [14], however we propose a new feature selection method named PCA_RELIEF based on PCA and relief which performs a high accuracy rate in our experiment.

Different from the existing malware detection methods, our system extracts enough features from the malicious and benign Android samples, and we obtain more refined features than the existing detection approaches by using PCA-RELIEF algorithm.

SYSTEM OVERVIEW

Our system can distinguish the malware directly on the Android smartphone. Considering the limitation of resources in smartphones, client–server model is proposed in our system. Fig.1 describes the whole structure of our malware detection system.

System Description

As is shown in Fig.1, our system is divided into two main sections, namely client and server. In the part of client, it mainly provides the UI (user interface) for the users, which will alert the result that our system predicts. Because of the limited resources, we just put a simple check in the client, we collected a number of malware samples' MD5 value, when a new application is installed, our system will extract its MD5 value and compare it with the malicious MD5 in the SQLite, if the new MD5 value exists in the SQLite, then out system will alert a malicious information and remind the user to delete it, however if the new MD5 value does not exist in the SQLite, then this APK will be submitted to the server. In the server, features of the application will be extracted in the feature extraction module by using static analysis and dynamic analysis, we extract permission, intent, uses-feature, application and API as the static features,

and we choose the CPU consumption, the battery consumption, the number of running processes and the number of short message as the dynamic features. The raw features will be sent to the feature selection module to select some key features and reduce the redundant features based on PCA-RELIEF. Finally, we build a classification model by using SVM and evaluate the unknown Android application by classifying it into malware or benign.

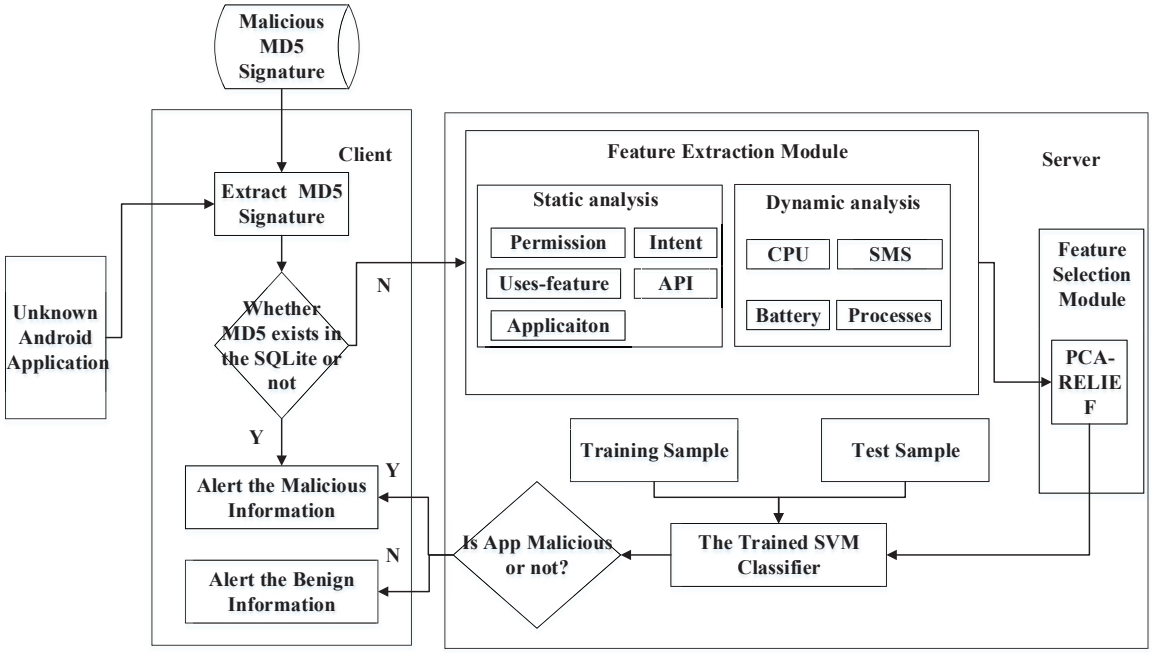

FIGURE 1. The architecture of machine learning-based Android malware detection

Feature Extraction

Static Analysis Module

Static analysis and dynamic analysis are used to extract features. When talking about the static analysis, we implement a decoder based on the Androguard [15] tool which is one of the largest open source projects for Android static analysis. After decompiling, we collect various features from AndroidManifest.xml, classes. dex. Table 1 shows the part of the static features we extract from a malicious sample.

TABLE 1. Features extracted from the APK with the static analysis

Category	Value
Permission	Android. Permission. RECEIVE_SMS
	Android. Permission. SEND_SMS…
API	android/telephony/Telephony Manager; get Device ID
	Android/telephony/Telephony Manager; get Subscribe rid...
Uses-feature	Android. Hard ware. touchscreen
	Android. Hardware. Camera…
Application	Com. air push. Android. Message Receiver
	Com. air push. Android. Delivery Receiver...
Intent	Android .intent. Action. BOOT_COMPLETED android. Intent. Action. PHONE_STATE…

As is shown in Table 1, we choose permission, API, uses-feature, application, intent as the static features. Research finds that permission system in Android is one of the most important security mechanisms, malicious software tends to request sensitive permissions more than benign software, such as *android. Permission. SEND_SMS*, etc. Similarly, uses-feature defines the access to the hardware, as is shown in Table 1, this malicious application applies access to the touchscreen and the camera, requesting access to specific hardware often reflects harmful behavior. Application consists of four different types of components in an application, the names of these components may help identify the famous components of malware. Intent can be used to trigger malicious activities, thus it is also indispensable to collect the intents listed in the manifest. Suspicious API calls give access to sensitive data which can lead to malicious behavior. Therefore, we choose these attributes as the static features.

Dynamic Analysis Module

We implement a dynamic analysis module by using Droid Box [16], dynamic analysis will incur a large number of computational overhead and consume a lot of time, and thus our system will mainly use the static analysis. The dynamic analysis module is motivated when the static analysis failing to decompile the APK because some applications are obfuscated and encrypted. First, the dynamic module will launch the Android virtual machine to load the APK, and then invoke the Droid Box tool to monitor the application behavior and the system state, which is shown in Fig.1 called dynamic analysis module. After that, information collection module will gather the dynamic features generated during the running time.

Feature Selection

Description of Relief and PCA

Relief [17], one of the filter type, is a feature selection algorithm based on the relevance of features and classification which is known to solve the problem of two classifications by giving the weight score. However, relief has a shortcoming for its disability in eliminating redundant features, with the high relevant features the correlations will degrade the result.

PCA is a dimension reduction algorithm which helps transform original features linearly into a low-dimensional subspace to reduce the dimensionality of the raw data set. PCA is usually used to eliminate redundant features. Thus, in order to remedy the defect of relief, a new feature selection PCA-RELIEF is proposed to find the most discriminating Android feature subset.

Implement of PCA-RELIEF

We assume the Android data set as$D$, iterations asn, the nearest neighbor sample number as k, the weight of each feature as W, the target dimension as d, the threshold (the remaining number of features after PCA) as s, $s > d$.

1) Start: the algorithm will perform the PCA dimension reduction processing on the data set D to select s as the principal component, after that the dimension reduction of data sets is noted asD_1.

2) Put the W zero, it means $W_i = 0, \mathrm{i} = 1, 2, \ldots, \mathrm{s}$;

3) Repeat for n times and output the weight of each feature $W_i = 0, \mathrm{i} = 1, 2, \ldots, \mathrm{s}$;

1. Randomly select a sample M from D_1;

2. Find k-nearest neighbor samples $P_j(\mathrm{j} = 1, 2, \ldots, \mathrm{k})$ with the same category as M from D_1, find k-nearest neighbor samples $Q_j(\mathrm{j} = 1, 2, \ldots, \mathrm{k})$ with the opposite category as R;

3. Computing each weight of features with the following formula, the formula $\mathrm{diff}(a, \mathrm{X}, \mathrm{Y})$ expresses the distance between sample X and the feature a as well as the distance between sample Y and the feature a;

$$W_i = W_i - \frac{diff(i, M, P)}{n \times k} + \frac{diff(i, M, Q)}{n \times k} \tag{1}$$

$$diff(i,M,P)=\sum_{j=1}^{k}\frac{|M(i)-P\{j\}(i)|}{\max(i)-\min(i)} \tag{2}$$

$$diff(i,M,Q)=\sum_{j=1}^{k}\frac{|M(i)\{j\}(i)|}{\max(i)-\min(i)} \tag{3}$$

4) if $W_i \geq s$, put the weight into the result set, else remove the feature. Finally the algorithm will output the final W set and order them from high to low;

In general, first, PCA will be used to reduce the dimension of the features as well as eliminate the redundancy of the original features. After that, relief will be used to give the corresponding weights for the features of each sample. Last, d attributes with the highest feature weight are chosen to form the final feature subset.

EXPERIMENT AND DISCUSSION

We gather 2000 Android applications including 1000 benign applications and 1000 malware, the benign samples are collected in the Google Play by the crawler technology, and we gather the malware from the Drebin Project [18] and the Android Malware Genome Project [19]. We use the SVM classifier algorithm to build a classifier, 20% of the samples as the test data set, and 80% of the samples as the trained data set.

Table 2 shows the highest 10 attributes selected by PCA-RELIEF.

TABLE 2. The highest 10 attributes by PCA-RELIEF

Feature Name	Ranked Weight
Android. Permission. Read _Sms	0.915
Android. Intent. Action. Boot_ Completed	0.905
Android. Permission. Send_ Sms	0.873
Android/Telephony/Telephony manager; Getdeviceid	0.840
Android. Permission. Read_ Phone_ State	0.792
Android/Telephony/Telephony manager; Getsubscriberid	0.532
Android. Permission. Call_ Phone	0.521
Android. Permission. System_ Alert_ Window	0.321
Android. Permission. Access_ Wifi_ State	0.242
Android. Net. Wifi. Pick_Wifi_Work	0.232

As is shown in the Table 2, READ_SMS can be defined as the most signal feature in distinguishing the malware and the benign, some original features are removed because of its low rank, such as the INTERNET permission. By using PCA-RELIEF we gathered the highest 10 attributes of each feature for building the classification model.

True Positive Rate (TPR), False Positive Rate (FPR), and Accuracy [20]. TPR is the rate of correctly detection a sample, however FPR is defined as the false detection of benign application as malware. Accuracy shows the precise of the classifier in classifying the samples in the right class. SVM [21], a linear classifier, determines a hyperplane that separates both classes with maximal margin, we consider it for our system.

Table 3 shows the TPR, FPR and Accuracy in different feature selection algorithm.

TABLE 3. The comparison with PCA and Relief in SVM

Feature Selection Algorithm	**True Positive Rate**	**False Positive Rate**	**Accuracy**
PCA	0.873	0.256	0.893
RELIEF	0.918	0.180	0.941
PCA-RELIEF	0.947	0.133	0.952

As is shown in Table 3, the result indicates that the new feature selection algorithm PCA-RELIEF is effective in the Android malware detection with the accuracy of 0.952. Despite the result is similar, the PCA-RELIEF has smaller

FPR and higher TPR than other two algorithms. Hence we conclude that PCA-RELIEF performs well in feature selection based on the SVM classifier.

CONCLUSION

In this paper, we implement an Android malware detection system based on SVM, different from the traditional detection method, it can detect the unknown Android application based on the machine learning. We extract various features with the method of static analysis and dynamic analysis. A new feature selection algorithm PCA-RELIEF is also proposed to dispose the raw features and our experimental result shows that the new method performs better with higher detection rate and lower error detection rate compared with the traditional detection approaches such as the detection method based on signature.

ACKNOWLEDGMENTS

The authors would like to express our thanks to the Drebin Project and the Android Malware Genome Project for sharing the malware samples.

REFERENCES

1. IDC .the Android data statistics in the second quarter of 2016[EB/OL].http://www.baijingapp.com/article/7842
2. X. Li, J. Liu, Y. Huo, R. Zhang and Y. Yao, "An Android malware detection method based on Android Manifest file," *2016 4th International Conference on Cloud Computing and Intelligence Systems (CCIS)*, Beijing, 2016, pp. 239-243.R. T. Wang, “Title of Chapter,” in Classic Physiques, edited by R. B. Hamil (Publisher Name, Publisher City, 1999), pp. 212–213.
3. L. D. Coronado-De-Alba, A. Rodríguez-Mota and P. J. E. Ambrosio, "Feature selection and ensemble of classifiers for Android malware detection," *2016 8th IEEE Latin-American Conference on Communications (LATINCOM)*, Medellin, 2016, pp. 1-6.
4. Qin Z, Xu Y, Liang B, et al. An Android malware static detection method [J]. Journal of Southeast University, 2013, 43(6):1162-1167.
5. Felt A P, Chin E, Hanna S, et al. Android permissions demystified[C]// ACM Conference on Computer and Communications Security. ACM, 2011:627-638.
6. Yang Z, Yang M. LeakMiner: Detect Information Leakage on Android with Static Taint Analysis[C]// Software Engineering. IEEE, 2012:101-104.
7. Wu D J, Mao C H, Lee H M, et al. DroidMat: Android Malware Detection through Manifest and API Calls Tracing[C]// Information Security. 2012:62-69.
8. Qiao Y, Yang Y, He J, et al. CBM: Free, Automatic Malware Analysis Framework Using API Call Sequences [M]// Knowledge Engineering and Management. Springer Berlin Heidelberg, 2014:225-236.
9. Tam K, Khan S J, Fattori A, et al. CopperDroid: Automatic Reconstruction of Android Malware Behaviors[C]// Network and Distributed System Security Symposium. 2015.
10. Zheng C, Zhu S, Dai S, et al. SmartDroid: an automatic system for revealing UI-based trigger conditions in android applications [J]. 2012.
11. Qiao M, Sung A H, Liu Q. Merging Permission and API Features for Android Malware Detection[C]// Iiai International Congress on Advanced Applied Informatics. 2016:566-571.
12. Munoz A, Martin I, Guzman A, et al. Android malware detection from Google Play meta-data: Selection of important features[C]// Communications and Network Security. IEEE, 2015:701-702.
13. Peiravian N, Zhu X. Machine Learning for Android Malware Detection Using Permission and API Calls[C]// IEEE, International Conference on TOOLS with Artificial Intelligence. IEEE, 2013:300-305.
14. Mas'Ud M Z, Sahib S, Abdollah M F, et al. Analysis of Features Selection and Machine Learning Classifier in Android Malware Detection[C]// International Conference on Information Science & Applications. IEEE, 2014:1-5.
15. A. Desnos and G. Gueguen, et al.https://github.com/androguard/androguard, visited August 2015.
16. Droidbox project.https://github.com/pjlantz/droidbox.
17. Kira K, Rendell L A. The feature selection problem: traditional methods and a new algorithm[C]// National Conference on Artificial Intelligence. San Jose, Ca, July. DBLP, 1992:129-134.

18. Arp D, Spreitzenbarth M, Hübner M, et al. DREBIN: Effective and Explainable Detection of Android Malware in Your Pocket[C]// Network and Distributed System Security Symposium. 2014.
19. Zhou Y, Jiang X. Dissecting Android Malware: Characterization and Evolution[C]// Security and Privacy. IEEE, 2012:95-109.
20. Feizollah A, Anuar N B, Salleh R, et al. A Study Of Machine Learning Classifiers For Anomaly-Based Mobile Botnet Detection [J]. Malaysian Journal of Computer Science, 2013, 26(4):251-265.
21. Cristianini N, Shawe-Taylor J. An introduction to support Vector Machines: and other kernel-based learning methods [M]. Printed in the United Kingdom at the University Press, 2000.