Defence Science Journal, Vol. 66, No. 6, November 2016, pp. 618-623, DOI : 10.14429/dsj.66.10803

An Android Malware Detection Framework-based on Permissions and Intents

Sushma Verma[*] and S.K. Muttoo[#]

[*]*Scientific Analysis Group, Delhi - 110 054, India*
[#]*University of Delhi, Delhi - 110 054, India*
[*]*E-mail: sushmaverma@sag.drdo.in*

ABSTRACT

With an exponential growth in smartphone applications targeting useful services such as banks, healthcare, m-commerce, security has become a primary concern. The applications downloaded from unofficial sources pose a security threat as they lack mechanisms for validation of the applications. The malware infected applications may lead to several threats such as leaking user's private information, enforcing malicious deductions for sending premium SMS, getting root privilege of the android system and so on. Existing anti-viruses depend on signature databases that need to be updated from time to time and are unable to detect zero-day malware. The Android Operating system allows inter-application communication through the use of component reuse by using intents. Unfortunately, message passing is also an application attack surface. A hybrid method for android malware detection by analysing the permissions and intent-filters of the manifest files of the applications is presented. A malware detection framework is developed based on machine learning algorithms and on the basis of the decision tree obtained from ID3 and J48 classifiers available in WEKA. Both algorithms gave same results with an error percentage of 6 per cent. The system improves detection of zero day malware.

Keywords: Permissions, intents, pruning, clustering, classification

1. INTRODUCTION

Smartphones have become a very soft and vulnerable target for malicious application developers. Malicious software is a common problem for every software platform, and the android platform is no exception. The most serious threats for android users come from applications. However, the markets especially the unofficial markets lack a mechanism to validate whether these applications are malware or not[1,2]. So, malware may leak user's private information without their knowledge or consent. Android relies on granular permissions to avoid granting full privileges to third-party applications. However, the permission system on Android is complex. There are at least 135 permissions. The developers as well as the users are unaware of the actual scope of each permission. Intents in the Android operating system provide a inter-application message passing system which is used to link applications. However, message passing acts as an application attack surface which can be used to modify, steal or sniff the content of messages leading to compromising user privacy. A malicious application can inject malicious messages resulting in user data breaches and violating application security policies. This creates room for exploitations; malicious applications disguise themselves amongst the hundreds of thousands of normal ones.

In the traditional method of malware detection, signature is the only basis. The existing anti-viruses depend on signature database that needs to be updated from time to time and are unable to detect zero day malware[2]

We propose a hybrid detection framework on the basis of Permission and Intents which will be treated as features to detect malware infected applications using classifiers such as ID3 and J48 on the training set in WEKA. The system effectively detects zero day malware.

2. RELATED RESEARCH WORK

The increased usage of smartphones can be combined with tremendous increase in the security breaches due to the exploitation of the increasing number of android application -related vulnerabilities. The continuing exponential growth in the usage of the smart mobile technology has necessitated the research in security solutions for mobile devices. Boksasp and Utnes[3] investigates the used by the application to identify the suspicious applications. The focus of the thesis is not to identify malicious applications but to identify permissions which can indicate malicious behaviour. Derby and Wilson[4] analyses 3rd party apps for possible ways a malicious Intent could trigger privileged calls by the app. Zarni and Zaw[5] develops a framework for classifying Android applications using machine-learning techniques-based on permission features to detect malware or normal applications[5].

A large scale study on application permissions and risk signals[6] analysed the effectiveness of the permission systems

Received : 01 April 2016, Revised : 12 July 2016
Accepted : 08 August 2016, Online published : 31 October 2016

for Android, Chrome and Facebook platforms to investigate privacy risks attempting to trick users into granting unsafe permissions. Droidranger[7] examines the permissions requested by the applications on the Google Play and third party markets by behavioural foot printing. Felt[8], *et al.* reported over-privileged applications are detected-based on permissions in the Android platform. These permissions are useful for the analysis of our malicious data set. A survey of mobile malware in the wild by Pporter[9], *et al.* details their analysis of the incentive behind malicious applications and the effectiveness of measures taken to prevent infections and identifying malicious applications. It explains the motivation behind several types of malicious behaviour, and the measures taken to prevent it.

A methodology for empirical analysis of permission-based security models and its application to android by Barrera[10] illustrates the permissions category wise and discusses ways to improve the android permission model. Analyses of feature-based selection and classification algorithms for permission-based android malware analysis was carried out by Pehlivan[11], *et al*. and Mas' ud[12], *et al*.

3. PROPOSED METHODOLOGY

Our methodology aims at developing a detection system that focuses on feature extraction and selection to measure and characterise the malicious applications on the basis of permissions and intents specified in the manifest file of the application. We have used information gain algorithm for feature selection. The features thus constructed from the selected feature sets are fed to the classifier using WEKA. The permission-based detection is carried out using the algorithm derived from the decision tree obtained from C4.5 (J48) classification and intent-based detection is carried out using the algorithm derived from the decision tree obtained from ID3 classification. According to the hybrid framework proposed as depicted in Fig. 1.

- If a sample application is detected as infected by both permission and intent-based detection then it is classified as Malicious.
- If a sample application is detected as not infected by permission-based detection and is detected as infected by intent-based detection or vice-versa then it is suspicious application.
- If a sample application is detected as not infected by both permission-based and intent-based detection then it is a benign application.

The steps involved in the process are as follows:

Collection of both infected and benign applications from Google Play and other unofficial markets.

- Extraction of features (useful information) from each application which includes the permissions requested, intents etc.
- Organisation of the collected features into structured database
- Classify the malwares using machine learning algorithms.
- Propose a hybrid malware detection framework-based on both permissions and intents.

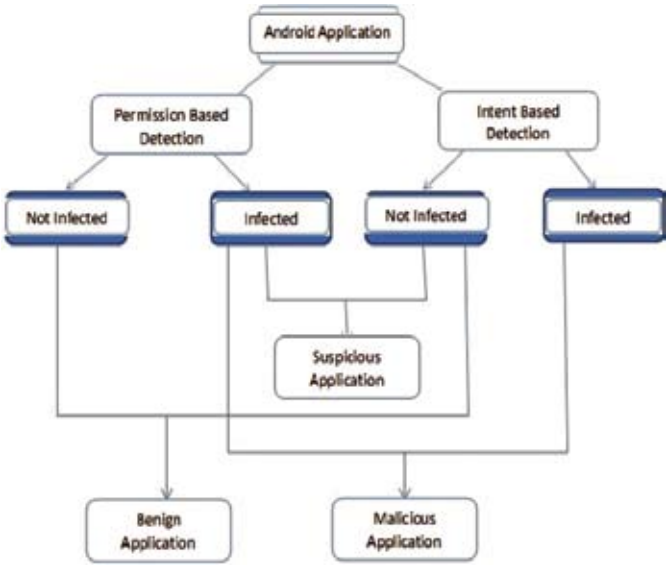


Figure 1. Framework for android malware detection system-based on hybrid features.

3.1 Feature collection

The datasets were generated using both infected and non-infected Android applications. A total of 850 applications were downloaded from the Android official app market – Google Play and 620 malicious apps from contagiomini dump[13]. All downloaded files were scanned using commercial antivirus virustotal[14].

3.2 Feature Extraction

To apply any machine learning classifier, it is important to first collect the relevant features. Dex2jar and JDGUI is used to disassemble apk file. Features are extracted from AndroidManifest.xml. Some of the features that were used for classification are as listed in Table 1.

The AndroidManifest.xml file thus obtained is converted into .arff format to be analysed in WEKA. The permissions requested that are specified in the <uses-permission> and the intents can thus be extracted and used as features for classification.

Two databases are created one containing the permissions requested by all the apps and the other containing the intents of the apps. The permission database include the various attributes including standard permissions, the name of the malware in case of malicious app, infected attribute indicates whether the app is infected or non-infected app. The intent database consists of the flag to indicate the presence or lack of an intent marked as "1" or a "0" in the respective column. Figure 2 shows some of the Android applications permissions used for malware detection.

3.2.1 Permission-based Detection

Some of the malicious permissions are explained as follows:Text messaging permissions, Read Phone State, Install and delete packages, Internet, Change Configuration.

3.2.2 Intent-based Detection

Presence of malware in an application can be detected

Table 1. Features used for classification

	Permissions	Intents	Explanation
Text messaging permission	Send_SMS, Receive_SMS	SMS_received	SEND SMS permission is the most prolific, which makes sense as an attacker leveraging text messages need this permission to send them from the device. The WRITE SMS is required for a premium SMS service to send single-charge messages.
	Read_SMS, Write_SMS	New_outgoing_call/ SMS	
Install and delete packages		Install_application, install_packages	Used by the Pjapps Trojan. The INSTALL PACKAGES permission let application to install backdoors onto infected devices. Can be usedto delete anti-virus application or other applications limiting its capabilities.
		Delete_packages	
	Internet	Connectivity_ changeUser_present	Malware application can send user privacy information to their websites.
Read Phone State	Call_Phone, Read_phone_ state	Phone_state, Modify_Phone_State	It provides any application requesting it with access to a large amount of data, including IMEI, IMSI and number of the device.
	Change_ Configuration	Boot_completed,	It allows to change configuration files of the mobile devices. This permission can be used to block the functionalities and services of the mobile devices.

on the basis of the intents (message passing) used by the application. An Intent is a message that includes a recipient and optionally includes data. Intents can be sent between three of the four components: activities, services, and broadcast receivers. They can be used to start activities; start, stop, and bind services; and broadcast information to broadcast receivers. Confidentiality or integrity can be compromised if a sender does not correctly specify the recipient of the message leading to interception by the attacker. The content of messages can be sniffed, modified, stolen, or replaced, which can compromise user privacy and violate application security policies. Android provides a sophisticated message passing system, in which Intents are used to link applications. It is must to explicitly set the exported attribute in the intent to true or false, else its default value which is true, can invoke any other application on the system[8]. The detection is-based on one type of intent i.e. action type. We analyse the intents along with the permissions for classifying malicious and benign apps.

Some of the common intents used in applications are analysed to understand their functioning[4].

- ACTION_DIAL;_ Perform a call to someone specified by the data. This Intent cannot be used to call emergency numbers or premium numbers by malicious applications.
- ACTION_CALL_BUTTON: This intent is used to involve the dialerby pressing the "call button"
- ACTION_CREATE_SHORTCUTOutput: An Intent representing the shortcut is used to create a SHORTCUT_INTENT
- ACTION_DATE_CHANGED: This intent is used to change the date of the system thereby disabling lot of applications such as browser to seize working.
- ACTION_DELETE: Delete the given data from its container.
- ACTION_INSTALL_PACKAGE: This intent is used tolaunch application installer. It can be used to install malicious applications.
- ACTION_MAIN: Start as a main entry point, does not expect to receive data.
- ACTION_MEDIA_REMOVED: External media has been removed.

The subsequent sections will describe the application of machine learning algorithms to identify malicious and suspicious applications in Android and their results.

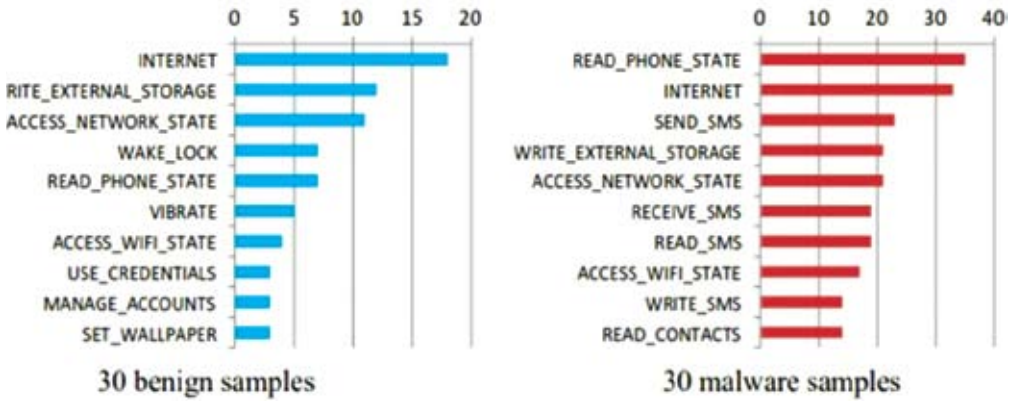


Figure 2. Android applications permissions used for malware detection.

3.3 Feature Selection

All these permissions may not be relevant in detecting the infected app and can result in false positives thereby misleading the learning algorithm and increasing complexity and run-time. These adverse effects are even more crucial when applying. The application of machine learning methods on mobile devices can lead to adverse effects as these are often restricted by processing and storage-capabilities such as battery power. Thus there is a

need for selecting the most relevant features out of the dataset that are sufficient for detecting the infected app.In order to improve the efficiency of classification these redundancy and inconsistency features must be eliminated which is done by Feature Selection. We have used the information gain algorithm for feature selection. This method results in selecting the k best features from the extracted features of android applicationpackage files. It depends on entropy of the attributes and selects the largest value of gain as the best feature.

3.4 Malware Detection using Clustering and Classifiers

The selected features are collected into the signature database and divided into training data and test data and used by standard machine learning techniques to detect the android malware applications. In the first step we have used K-Means clustering to obtain k disjoint clusters on training datasets each cluster depicts a region of similar features instances in terms of Euclidean distances between the instances and their cluster centroids. However, K-Means clustering may lead to anomalies in case of overlapping type of data. So we use decision tree classifiers to classify each cluster. K-means method is cascaded with decision tree learning by using the instances in each K-means cluster. However, K-Means clustering may lead to anomalies in case of overlapping type of data. So we use decision tree classifiers to classify each cluster. K-means method is cascaded with decision tree learning by using the instances in each K-means cluster. We have used K-means clustering as it guarantees at least a local minimum of the criterion function, thereby accelerating the convergence of clusters on large datasets. It is an iterative algorithm which organises the training set in vectors with a dimension equal to the number of features to be evaluated leading to k number of clusters.

K-means consists of two steps:

1. Calculate the centroid for each cluster.
2. Calculate the distance of the training set vectors from each cluster centroid
3. Assign the training set vector to the closest cluster centroid.
4. Move the cluster centroids to the mean of the respective cluster's training sets.

The steps are repeated until the algorithm converges. Convergence is achieved when the second step no longer assigns any vectors to new cluster centroids.

The distance between the n-dimensional vectors (x) and a given cluster centroid is given by:

$$\|x-\mu\|^2$$

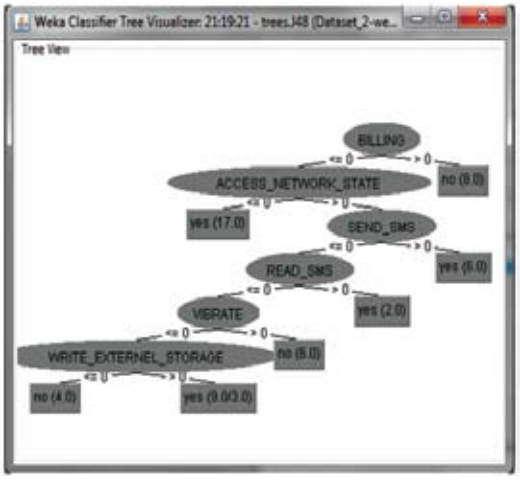
where x is a training example, and μ is the cluster centroid. In the experiments, it was seen that in case of seed ie. Centroid μ (equal to 12), the error percentage was 15.3846 per cent which was the least error percentage obtained.

3.5 Classification

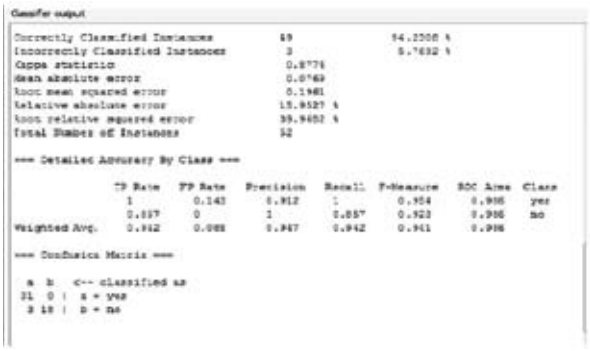
As the training set is usually very large, the branches and layers of generated tree using J48 decision tree algorithm are more, so it is necessary to prune the decision tree. We have also used ID3 and C4.5 classification algorithms to obtain a relationship between the attributes that would make it possible to detect whether an app is infected or not. The experiments show that out of 620 instances of infected applications, 605 were correctly classified using ID3 and the error percentage was only 6 per cent as can be seen from the Table 2. The table shows that ID3 gives better results for permissions whereas J48 performs better for intents. However any of the two classifiers from ID3 or J48 can be used for detection purposes.The Fig. 3 below shows the resultant tree obtained from the database by applying J48 with pruning through which an app can be detected as being infected or not infected.

4. EXPERIMENTAL RESULTS

We have performed our experiments on an Intel Pentium Core 2 Duo 2.33 GHz processor with 2GB RAM. Benign



(a)



(b)

Figure 3. (a) J48 decision tree and (b) result of J48 classification with pruning using WEKA.

Table 2. Parameters for confusion matrix from experimental results

Machine learning technique	Algorithm	No. of correctly detected apps	No. of incorrectly detected apps	Total no. of apps	Error percentage (per cent)	Accuracy percentage (per cent)
Clustering	k-means	1087	382	1470	26	74
Classification	ID3	1352	108	1470	8	92
Classification	J48	1381	88	1470	6	94

apps were downloaded from the Android official app market – Google Play and the malicious apps from contagiominidump[13]. The collected apps were classified as malicious or benign by using virustotal[14].

Our dataset consists of 620 malicious samples and 850 benign applications. We first extracted features and then built ARFF file from the extracted features used to train the dataset by using K-means clustering algorithm. The database is then divided into two - the training data set and the testing data set.Selection of most relevant features out of the dataset is done by using information gain algorithm. It determines the useful attributes in a given set of training feature vectors for discriminating between the classes. Once the training model has been developed, we used decision tree-based classification algorithms ID3 and C4.5 (or J48) for each cluster to classify the malware applications using machine learning tool kit Weka[15] and then the testing set was used to test the efficiency of the model in detecting infected apps. Both algorithms gave same results with an error percentage of 6 per cent.

The following confusion matrices were created from the results obtained from the classifiers.

True Positive (TP): Number of benign applications correctly identified.

False Positive (FP): Number of malware applications wrongly identified.

True Negative (TN): Number of malware applications correctly identified.

False Negative (FN): Number of benign applications which are wrongly identified.

True Positive Rate (TPR): Percentage of correctly identified benign applications

TPR = (TP / TP+FN)

False Positive Rate (FPR): Percentage of wrongly identified malware applications

FPR= (FP / TN+FP)

Overall Accuracy (ACC): Percentage of correctly identified applications

ACC= (TP+TN / TP+TN+FP+FN)

The performances of classifiers were evaluated using the true positive rate, false positive rate and overall accuracy.

5. CONCLUSION

We have implemented a framework for detecting malicious android applications using permission and intent-based analysis. After extracting several permissions and intents as features from several downloaded, feature set was reduced using information gain algorithm. The database is then divided into two - the training data set and the testing data set. We then used k-means clustering followed by the J48 and ID3 classifier using the training data to correctly detect malware among the given applications. From the experiments conducted and data collected thereof, we conclude that the J48 classifier produces the best results with an accuracy of 94 per cent. Performances were compared in terms of accuracy and error percentage derived from the confusion matrix. The hybrid approach for malware detection of android applications will enhance the security of the mobile device. Future work may include evolving and comparing improved classification algorithms for detection of malicious applications.

REFERENCES

1. Zdnet.http://www.networkworld.com/article/2223327/security/fbi-warns-loozfon--finfisher-mobile-malware-hitting-android-phones.html (Accessed on 07 March 2016)
2. Vaibhav Rastogi, Y.C. & Jiang, Xuxian. Evaluating anti-malware against transformation attacks. NU-EECS-13-01, March 2013.
3. Boksasp, T.U.; E. Android apps and permissions: Security and privacy risks. NTNU - Trondheim, Norwegian University of Science and Technology 2012 (Norway, 2012.). (Master's thesis)
4. Kelly Casteel, K.; Owen Derby, oderby & Dennis Wilson, Dennisw. Exploiting common intent vulnerabilities in android applications. *In* Proceedings of the 18th ACM conference on Computer and communications, 2012.
5. Zarni Aung, W.Z. Permission-based android malware detection. *In* Proceedings of the International Journal of Scientific and Technology Research., 2013, **2**(3).
6. Chia, P.; Y.Y. & Asokan, N. Is this app safe? A large scale study on application permissions and risk signals. *In* Proceedings of the 21st International World Wide Web Conference, 2012.
7. Zhou, Y.; Z.W.; Zhou, W. & Jiang, X. Droidranger: Hey, you, get off of my market: Detecting malicious apps in official and alternative Android markets. *In* Proceedings of the 19th Annual Symposium on Network and Distributed System Security, 2012.
8. A. P. Felt, E.C., S. Hanna, D. Song, & D. Wagner, Stowaway: Android permissions demystied. *In* Proceedings of the 18th ACM Conference on Computer and Communications Security, 2011.
9. Porter A.; Felt, M.F.; Chin, E.; Hanna, S. & Wagner, D. A survey of mobile malware in the wild. *In* Proceedings of the 11th Workshop on Security and Privacy in Smartphones and Mobile Devices, 2011. Chicago, IL, USA, pp. 3-14. doi: 10.1145/2046614.2046618
10. Barrera, D.; Kayacik, H.; van Oorschot, P. & Somayaji, A. A methodology for empirical analysis of permission-based security models and its application to android. *In* Proceedings of the ACM conference on Computer and Communications Security, 2010. Chicago, IL, USA, pp. 73-84. doi: 10.1145/1866307.1866317
11. Pehlivan, U.; Baltaci, Nuray; Acartürk, Cengiz & Baykal, Nazife. The analysis of feature selection methods and classification algorithms in permission-based Android malware detection. IEEE Symposium on Computational Intelligence in Cyber Security (CICS), 2014. doi 10.1109/CICYBS.2014.7013371
12. Mas' ud, M.Z. Analysis of features selection and machine learning classifier in android malware detection. International Conference on Information Science & Applications (ICISA), IEEE, 2014.

13. Contagiominidump.www.contagiominidump.blogspot.com. (Accessed on 05 February 2016)
14. Virustotal.www.virustotal.com. (Accessed on 24 February 2016)
15. WEKA.http://www.cs.waikato.ac.nz/~ml/weka/. (Accessed on 17 March 2016)

ACKNOWLEDGEMENTS

I would like to thank my Director, Ms Anu Khosla, Scientist 'G' for giving me the opportunity to working the area of security and privacy in mobile computing andall those who directly or indirectly provided support in the area of malware analysis. And author is thankful to University of Delhi for providing research grant videRC/2015/9677.

CONTRIBUTORS

Ms Sushma Verma obtained his MSc (Ccomputer Science) from J.K. Institute of Applied Physics and Electronics , Allahabad University in 1993. She joined DRDO, Ministry of Defence in May 1993 and working there as scientist. Her area of specialisation includes mobile security, mobile malware analysis, formal verification and information security.

Dr S.K. Muttoo obtained his MSc (Mathematics), MPhil (Mathematics) and PhD from University of Delhi in 1976, 1978 and 1982, respectively. He also did his MTech (CSDP) from IIT Kharagpur in 1990 and joined Department of Computer Science , University of Delhi as Reader in 1991. His area of specialisation include coding theory, cryptography and information hiding techniques. He is life member of Computer society of India and Cryptographic Society of India.