2013 Seventh International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing

Analysis of Clustering Technique in Android Malware Detection

Aiman A. Abu Samra
Computer Engineering
Islamic University of Gaza
Gaza, Palestine
aasamra@iugaza.edu.ps

Kangbin Yim
Information Security Engineering
Soonchunhyang University
Asan, Korea
yim@sch.ac.kr

Osama A. Ghanem
Information Technology
University Collage of Applied Science
Gaza, Palestine
oghanem@ucas.edu.ps

***Abstract*—Mobile computing is an important field in information technology, because of the wide use of mobile devises and mobile applications. Clustering gives good results with information retrieval (IR); It aims to automatically put similar applications in one cluster. In this paper, we evaluate clustering techniques in Android applications. We explain how we can apply clustering techniques in malware detection of Android applications. We also use machine-learning techniques in auto detection of malware applications in the Android market.**

Our evaluation is given by clustering two categories of Android applications: business, and tools. We have extracted 18,174 Android's application files in our evaluation using clustering. We extract the features of the applications from applications' XML-files which contains permissions requested by applications. The results gives a positive indication of using unsupervised machine learning techniques in malware detection in mobile applications using a combination of the application information and xml AndroidManifest files.

Keywords—Android applications Clustering; Malware; Security; Smartphones.

I. INTRODUCTION

smartphone has rapidly become an extremely prevalent computing platform, with just over 115 million devices sold in the third quarter of 2011, a 15% increase over the 100 million devices sold in the first quarter of 2011, and a 111% increase over the 54 million devices sold in the first quarter of 2010 [1][2]. Android in particular has seen even more impressive growth, with the devices sold in the third quarter of 2011 (60.5 million) almost triple the devices sold in the third quarter of 2010 (20.5 million), and an associated doubling of market share [2]. This popularity has not gone unnoticed by malware authors. Despite the rapid growth of the Android platform, there are already well-documented cases of Android malware, such as DroidDream, which was discovered in over 50 applications on the official Android market in March 2011 [3]. Furthermore, Enck et al. found that Android's built-in security features are largely insufficient, and that even non-malicious programs can (unintentionally) expose confidential information [4]. A study of 204,040 Android applications conducted in 2011 found 211 malicious applications on the official Android market and alternative marketplaces [5][6]. The Android market is the fastest growing mobile application platform. A recent report from Lookout shows that the Android Market is growing at three times the rate of Apple's App store [7]. However, unlike Apple's Apps Store, which may check each available application manually by software security experts, there is lack of complete app inspecting process before the applications being published on the Android market. Google takes passive mechanism that allows anyone to publish applications on the Android market. If users report an application as malicious, it will then be removed. The openness of the Android market attracts both benign and malicious developers. Furthermore, Android allows installing third-party applications that may increase the spread of Android malware. Android security model highly relies on permission-based mechanism [8][9]. There are about 130 permissions that govern access to different resources. Whenever the user install a new app, he would be prompt to approve or reject all permissions requested by the application [10].

Machine learning techniques have been widely applied for classifying applications mainly focused on generic malware detection [11][12][13][14][15]. Besides, several approaches [16][17] have been proposed to try to classify applications specifying the malware class; e.g., Trojan, worms, virus; and, even the malware family [18].

This paper is organized as follows. The next section presents related works; section 3 describes in details our methodology; section 4 presents experiment design and evaluation; section 5 discusses results, finally section 6 concludes.

II. RELATED WORKS

Classification of mobile applications is used more widely than mobile applications clustering. There are many papers that discuss classifications, but we will discuss a clustering technique in Android applications, which is an important technique in machine learning and can give automatic classification of applications. Another approach uses

978-0-7695-4974-3/13 $26.00 © 2013 IEEE
DOI 10.1109/IMIS.2013.111

machine learning techniques in Android applications categorization using couple of two files extracted from applications: java and AndroidManifest files.
In this paper we will discuss clustering technique in Android applications using AndroidManifest file, which will facilitates clustering task and reduce the time.
Sanz et al. propose a method for classifying Android applications using machine-learning techniques [18]. They extract several features from several applications: data from "AndroidManifest.xml", data from Android Market, and the strings contained in applications. They have collected 820 samples of Android applications classified into 7 different categories. They evaluated this approach of automatical categorization of Android applications and have showed that achieves a high performance.

Shabtai et al. propose a new method for categorizing Android applications through machine-learning techniques [19]. To represent each application, the method extracts different feature sets: the frequency of occurrence of the printable strings, the different permissions of the application itself, and the permissions of the application extracted from the Android Market. They evaluated this approach of automatically categorization of Android applications and showed that achieves a high performance.

Sahs and Khan present a machine learning based system for the detection of malware on Android devices [6]. There system extracts a number of features and trains a One- Class Support Vector Machine in an offline (off-device) manner, in order to leverage the higher computing power of a server or cluster of servers.

The authors of [20] were the first to introduce the idea of applying Machine Learning (ML) methods for the detection of different malware based on their respective binary codes. Three different types of features were tested: program header (Portable Executable section), string features (meaningful plain-text strings that are encoded in programs files such as "windows", "kernel", "reloc" etc.) and byte sequence features. The study found that all of the ML methods were more accurate than the signature-based algorithm [19].

III. METHODOLOGY

We apply a clustering technique using K-means algorithm, which is widely used in Machine Learning. We extract many of the android applications in two categories. These applications will be extracted to get features and use it in clustering technique. As shown in Figure 1, there are applications clustering and evaluation system includes following steps:

1. Collect Android applications
2. Extract applications
3. Process Android Manifest files
4. Extract data from Android Manifest files
5. Create ARFF file with extracted features
6. Apply k-means algorithm
7. Evaluation

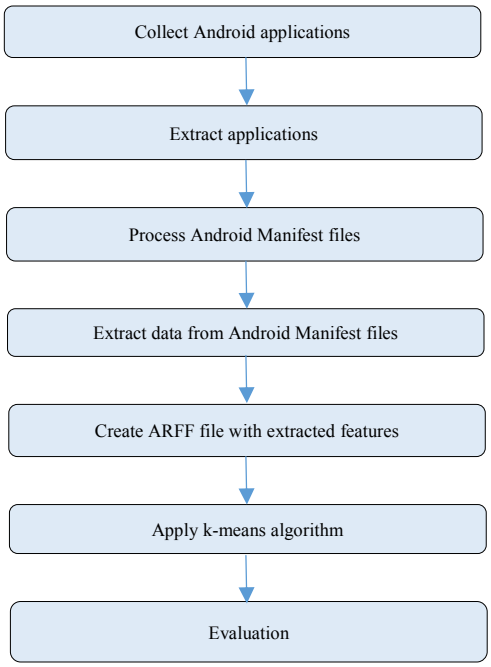

Figure 1. Android applications clustering architecture

A. *Extracted Features*

Android applications are stored in mobile devices and Android market as .apk files, which are as compressed. These files can be extracted to a source files .XML, which give basic features about an application and contains permissions required by these applications. Android Manifest file gives the permissions required by the application.

Every APK must include a manifest file that, among other things, requests permission to access certain restricted elements of the Android operating system. These elements include access to various hardware devices (e.g. GPS, camera), sensitive features of the operating system (e.g. contacts), and access to certain exposed parts of other applications. For example, the permission "android.permission.INTERNET" requests the right to access the Internet, and "android.permission.READ CONTACTS" requests the right to access the users phone contacts database [6][21]. Examples of XML-based features [19] are as the following.

- Count of xml elements, attributes, namespaces, and distinct strings
- Features for each element name (e.g., count of distinct actions, distinct categories)
- Features for each attribute name
- Features for each attribute type (e.g., number, string)
- Used permissions in the Android Manifest

Other features about application are used:

- Application Name
- Application Category
- Application package

- Application description
- Application rating values
- Application rating counts
- Application Price

B. K-means Clustering algorithm:

K-means algorithm is used in our experiments to get the best clustering results. It follows a simple and easy way to classify a given document set through a certain number of clusters (assume k clusters). The main idea is to define k centroids, one for each cluster. The simple K-means algorithm chooses the centroid randomly from the applications set. The next step is to take each application belonging to a given data set and associate it to the nearest centroid. The K-means clustering partitions a data set by minimizing a sum of-squares cost function.

$$J = \sum_{j=1}^{k} \sum_{i=1}^{x} \left\| x_i^{(j)} - c_j \right\|^2 \qquad (1)$$

Where $||x_i^{(j)} - cj||^2$ is a chosen distance measure between an application $x_i^{(j)}$ and the cluster center c_j, is an indicator of the distance of the n applications from their respective cluster centroids [22]. We identify two clusters so the application will be assigned to a cluster that has the smallest distance between this application and any centroid of the two clusters.

C. Clustering Tool (WEKA)

WEKA is a data mining open-source tool in abroad, but it is rarely used at home. WEKA (Waikato Environment for Knowledge Analysis) is a famous with data mining software and is well received in abroad [23]. It supports several methods of mining, pre-processing, clustering, categorization, visualization and feature selection. Data provided for processing must be available on a single file and should have properties and be related in some way (ideally linear dependent). In a WEKA file data is modeled as a number of comma-separated nominal and/or numeric values and may be followed by the proposed class value. The most important drawbacks of WEKA are the inability to perform multi-relational processes and the lack of a merge tool for interconnected tables [24]. WEKA was rarely used in Android applications processing and at home, so we used it in our paper at home and K-means algorithm is used in Android applications clustering because of advantages we mentioned and by adjusting WEKA parameters and prepare ARFF file from extracted features.

IV. EXPERIMENT DESIGN AND EVALUATION

Experiments are applied by using 18,174 Android applications, of which data is available on the Internet[1]. Frank, Dong, Felt and Song collected information about 188,389 Android applications from the official Android Market in November 2011 [25]. This data set encompasses approximately 59% of the Android Market, which contained 319,161 active applications as of October 2011 [26]. To build data set, they crawled and screen-scraped the web version of the Android Market. Each application has its own description page on the Market website, but the Market does not provide an index of all of its applications. To find applications' description pages, they first crawled the lists of "top free" and "top paid" applications. These lists yielded links to 32,106 unique application pages. Next, they fed 1,000 randomly selected dictionary words and all possible two-letter permutations (e.g., "ac") into the Market's search engine. The search result listings provided them with links to an additional 156, 283 unique applications. Once they located applications' description pages, they parsed their HTML to extract applications' names, categories, average rating score, numbers of ratings, numbers of downloads, prices, and permissions. As depicted in Table 1 we used this dataset in our research by selecting two categories of Android applications: Business, and Tools. The first category contains 4,612 samples and the second contains 13,535 samples. We choose two classes of data because we interested in clustering data to two clusters as malicious and non-malicious applications. The evaluation depends on precision, recall and F-measure. Experiment environment as follows: operating system: Windows 7, CPU: Intel Core i7 2670QM 2.2 GHz, Memory:8 GB, WEKA version: 3.6.4.

TABLE I. NUMBER OF APPLICATION SAMPLES IN EACH CATEGORY OF TESTING DATA SET

Application Categories	Number of samples
Business	4,612
Tools	13,535
Total	**18,174**

A. Evaluation

There are many evaluation standards in information retrieval used in clustering such as Entropy, cluster purity, and F-measure, which will be used in this paper.

F-measure: The F-measure is a harmonic combination of the precision and recall values used in information retrieval [27]. Precision shows how many applications are in right cluster with respect to the cluster size. Recall shows how many applications are in the right cluster with respect to total applications. Precision and recall for class i and cluster j are defined as:

$$Recall\ (i,j) = \frac{n_{ij}}{n_j} \qquad (2)$$

$$Precision\ (i,j) = \frac{n_{ij}}{n_i} \qquad (3)$$

where n_{ij} is the number of applications with class label i in cluster j , n_i is the number of applications with class label i, and n_j is the number of applications in cluster j , and n is the total number of applications. The F-measure for class i and cluster j is given as:

$$F(i,j) = \frac{2 * Recall(i,j) * Precision\ (i,j)}{Recall(i,j) + Precision\ (i,j)} \qquad (4)$$

Then total F-measure of clustering process is calculated as:

[1] http://www.mariofrank.net/andrApps/index.html

$$F = \sum {}^{n_i}/_{n * maxF(i,j)} \qquad (5)$$

V. RESULTS AND DISCUSSION

In this section, we will discuss the results, as mentioned above. The data set of 18,174 Android applications consists of 4,612 samples of Business and 13,535 samples of Tools. We compute precision, recall and F-measure for clustering these samples in the mentioned clusters. The results of average precision, recall, and F- measure as shown in Figure 2 are: 0.71, 0.71, and 0.71 respectively. In other hand, false positive and false negative have the same results 0.2897. These results give a good performance of clustering in Android applications and show a positive indication to use this methodology in automatic Android applications categorization by using clustering techniques and can used it for detection malicious applications from big store of applications.

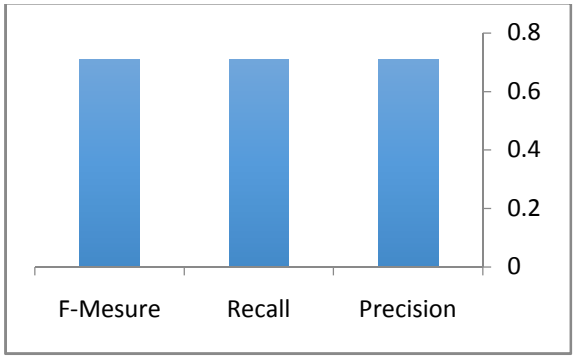

Figure 2. precision, recall and F-measure values for experiments

VI. CONCLUSION

We have applied unsupervised machine learning technique, Clustering, on 18,147 Android applications clustered in two categories by using their features extracted from the application requested permissions and other metrics such as: the price, the number of downloads, the average user rating, and a short prosaic description. We valuated our approach using F-measure, precision and recall that gives good performance results. The results show that using our malware detection technique is sufficient to detect threats without the need to install and running the applications. The technique then clusters the analyzed applications into two categories non-malicious applications, and malicious applications.

VII. REFERENCES

[1] Christy Pettey and Holly Stevens. Gartner says 428 million mobile communication devices sold worldwide in first quarter 2011, a 19 percent increase year-on-year. http://www.gartner.com/it/page.jsp?id=1689814.

[2] Christy Pettey and Holly Stevens. Gartner says sales of mobile devices grew 5.6 percent in third quarter of 2011; smartphone sales increased 42 percent. http://www.gartner.com/it/page.jsp?id=1848514.

[3] Lookout Mobile Security. Security alert: Droiddream malware found in official android market. http://blog.mylookout.com/blog/2011/03/01/security-alert-malware-found-in-official-android-market-droiddream/.

[4]William Enck, Damien Octeau, Patrick McDaniel, and Swarat Chaudhuri. A study of android application security. In Proceedings of the 20th USENIX Security Symposium, 2011.

[5] Yajin Zhou, Zhi Wang, Wu Zhou, and Xuxian Jiang. Hey, you, get off of my market: Detecting malicious apps in official and alternative android markets. In Proceedings of the 19th Network and Distributed System Security Symposium, 2012.

[6] Justin Sahs and Latifur Khan. A Machine Learning Approach to Android Malware Detection. 2012 European Intelligence and Security Informatics Conference.

[7] Lookout app genome report. https://www.mylookout.com/appgenome, 2011.

[8] D. Barrera, H. G. Kayacik, P. C. van Oorschot, and A. Somayaji. A methodology for empirical analysis of permission-based security

models and its application to android. In Proceedings of the 17th ACM conference on Computer and communications security, CCS '10, pages 73–84, New York, NY, USA, 2010. ACM.

[9] A. P. Felt, K. Greenwood, and D. Wagner. The effectiveness of application permissions. In Proceedings of the 2nd USENIX conference on Web application development, WebApps'11, pages 7–7, Berkeley, CA, USA, 2011. USENIX Association.

[10] Dong-Jie Wu, Ching-Hao Mao, Te-En Wei, Hahn-Ming Lee,and Kuo-Ping Wu. DroidMat: Android Malware Detection through Manifest and API Calls Tracing. 2012 Seventh Asia Joint Conference on Information Security.

[11] M. Schultz, E. Eskin, F. Zadok, and S. Stolfo, "Data mining methods for detection of new malicious executables," in Proceedings of the 2001 IEEE Symposium on Security and Privacy, 2001, pp. 38–49.

[12] J. Devesa, I. Santos, X. Cantero, Y. K. Penya, and P. G. Bringas, "Automatic Behaviour-based Analysis and Classification System for Malware Detection," in Proceedings of the 12th International Conference on Enterprise Information Systems (ICEIS), 2010, 395–399.

[13] I. Santos, J. Nieves, and P. G. Bringas, "Semi-supervised learning for unknown malware detection." in Proceedings of the 4th International Symposium on Distributed Computing and Artificial Intelligence (DCAI). 9th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS), 2011, pp. 415–422.

[14] I. Santos, C. Laorden, and P. G. Bringas, "Collective classification for unknown malware detection," in Proceedings of the 6th International Conference on Security and Cryptography (SECRYPT), 2011, pp. 251–256.

[15] I. Santos, F. Brezo, X. Ugarte-Pedrero, and P. G. Bringas, "Opcode sequences as representation of executables for data-mining-based unknown malware detection," Information Sciences, in press.

[16] K. Rieck, T. Holz, C. Willems, P. D¨ussel, and P. Laskov, "Learning and classification of malware behavior," in Proceedings of the 2008 Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA). Springer, 2008, pp. 108–125.

[17] R. Tian, L. Batten, R. Islam, and S. Versteeg, "An automated classification system based on the strings of trojan and virus families," in Malicious and Unwanted Software (MALWARE), 2009 4th International Conference on. IEEE, 2009, pp. 23–30.

[18] Justin Sahs and Latifur Khan . On the Automatic Categorisation of Android Applications. The 9th Annual IEEE Consumer Communications and Networking Conference - Security and Content Protection.

[19] Asaf Shabtai Yuval Fledel Yuval Elovici. Automated Static Code Analysis for Classifying Android Applications Using Machine Learning. 2010 International Conference on Computational Intelligence and Security.

[20]M. Schultz, E. Eskin, E. Zadok, S. Stolfo, "Data mining methods for detection of new malicious executables. Proc. IEEE Symposium on Security and Privacy, 2001.

[21] Google. Manifest.permission — android developers. http://developer.android.com/reference/android/Manifest.permission.html.

[22] M. Shameem, R. Ferdous. " An efficient K-Means Algorithm integrated with Jaccard Distance Measure for Document Clustering", Internet, 2009. AH-ICI 2009. First Asian Himalayas International Conference on.

[23]Hall M, Frank E, Holmes B, "The WEKA data mining software: an update", ACM SIGKDD Explorations Newsletter, Vol 11, No.1, pp. 10-18, 2009.

[24] Ioannis Charalampopoulos, Ioannis Anagnostopoulos. A Comparable Study employing WEKA Clustering/Classification Algorithms for Web Page Classification.

[25] Mario Frank, Ben Dong, Adrienne Porter Felt and Dawn Song. Mining Permission Request Patterns from Android and Facebook Applications.

[26] L. Horn, "Report: Android market reaches 500,000 apps," http://www.pcmag.com/article2/0,2817,2395188,00.asp

[27] C.J. van Rijsbergen, Information Retrieval, 2nd ed., Buttersworth, London, 1979.