IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 8, Issue 9, September 2019

Risk Calculation Methodology to Detect Malicious Applications

Lovi Dhamija

Teaching Assistant, Department of Electrical Engineering and IT, Punjab Agricultural University, Ludhiana, India

Abstract: Currently, in the smart-phone market, third-party stores are contributing a major share in developing Android apps. This paper represents a technique designed to calculate risk-factor for applications installed on smart-phones via third-party stores. The system represents an alternative to the current detection techniques by alerting users with risk alarm signals about malicious behaviour of application for the safety of mobile devices. The implementation methodology initially collects mobile applications from third-party stores and extracts feature set from many parameters of running mobile applications in an emulated environment. After that, for each feature-set, algorithms are proposed to calculate the risk factor for all the parameters which reveals the risk-level and its malicious impacts on mobile phones and leakage of privacy. Finally, the system is tested on four learning algorithms with WEKA API to find the best classifier for classifications of risk-levels. Among them, Logistic Regression shows 96% accuracy, RBF(Radial Basis Function) show 99% accuracy with some false positives, SMO (Sequential Minimal Optimization) shows 96.6% accuracy and Naïve Baye's produces 99.8% accurate results with very low false positives. Therefore, it is concluded that Naïve Baye's classifier can be integrated with the devised technique in future to detect risk levels of third-party applications.

Keywords: Smart Devices; android applications; malware; network traffic; permissions; risk-factor; logistic regression; Naïve Baye's; radial basis function

I. INTRODUCTION

With the advancement of services provided by mobile-devices such as computing, gaming and social networking, they become *'Smart Devices'.* However, smart-devices also carry a lot of personal and business information due to which they become a target to malware authors. Malware authors embed their malicious code into legitimate and useful applications distributed, *via*, application market. Moreover, the Android Platform is popular among malware authors to seek personal information and charging cost from users. Therefore, strong defence mechanisms must be introduced which are able to communicate about risks associated with these third-party applications, available to users on less cost. In this paper, we proposed a technique to calculate risk for Android applications by extracting feature-set from multiple sources. The central idea is to use auditing programs to extract permissions from the application manifest file to get to know about resources and API methods used by the application. Moreover, network traffic statistics are also captured at run time such as a number of packets sent, type of application, sending bytes, receiving bytes, connectivity etc. so that system can track normal traffic pattern and detect if the application starts behaving abnormally by sending data in excess even in off state. Different methods were employed to calculate risk values for two parameters. Therefore to assign a final risk value to an application, merging technique used truth table. Risk value classifies application with four values-normal, low, medium and high. Applications assigned high-risk values are considered dangerous as they are having the capability to leak personal data from mobile devices and can charge money from users.

II. LITERATURE REVIEW

This work is the first attempt to assign risk-scores to Android applications. However, it builds on a history of previous work combining static and dynamic analysis to detect malware from Android applications.

(Wu et al. 2012) proposed a static feature-based approach and develop a system named Droid Mat which is able to detect and distinguish android malwares. However, static based features are able to detect only known malwares[1].

(Joshua Abela et al. 2013) designed an automated behavioural analysis system AMDA for the distinction between normal and malicious behaviour. The model provides information about the access and privilege capacity of the application by analysing android application permissions. However, used static method was unable to detect obfuscated mobile malware. Additionally, time taken by the detection method creates a loophole for malware to infect system[2].
(Yajin et al. 2012) Proposed DroidRanger crawled apps from android market apply two-stage detection technique for analysing mobile application by extracting static-based features such as requested permissions and author information[3].

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 8, Issue 9, September 2019

Additionally, to detect repackaged applications from Android Market, DroidMOSS (Zhou et al.2012) implemented an app-similarity measurement. The system adopted a fuzzy hashing technique to detect changes added by in-app advertisements and revenues [4].

Moreover, dynamic analysis extracts feature-sets from mobile applications at run-time. The bouncer is operated by Google market to dynamically analyse mobile applications. However, the open-source nature of Android Platform allows devices to install applications from other sources also. Effective Risk Analysis and Risk Detection for Android Apps has been described by Patil (2016)[5]. Mining Permission Patterns for Contrasting Clean and Malicious Android Applications was also reported by Moonsamy et al. (2014)[6].

TaintDroid (Enck et al. 2010) dynamically analysed applications by tracking flow-sensitive data leaving from devices[7]. Kirin (Ontang et al. 2009) application certification process automatically mitigate malware at installation time. A set of rules are defined to block applications requiring dangerous permissions[8]. (Bose et al. 2008) presented behavioural detection framework. It detects the behaviour of mobile malware by observing the logical order of actions performed by applications over-time[9]. (Xie et al. 2010)a designed probabilistic model to observe unique behaviour of smart-phones through logs of keyboard operations and LCD displays and then correlated with system calls to detect anomalous activities[10].

III. METHODOLOGY

The method proposed for calculation of risk associated with smart-phone applications is divided into three steps which have been described below:-

A. Feature Extraction: The first step of the methodology is to extract features set from mobile applications which show behavioural analysis of applications.

- Permissions specified in manifest file are extracted and analyzed under four risk categories according to malicious impacts if permissions are granted to the application. Permissions required by the running application reveals about the resources used while execution.
- Network traffic statistics are also captured while an application is communicating with the server. Statistics include a number of packets transmitted and received, data bytes sent or received by an application, application state. By capturing network traffic, the normal behavior of applications is captured while the application is communicating.

A. Risk Calculation for extracted features.

Risk reveals the level of malicious impact by considering how much extent the privacy of mobile phones is leaked via third-party applications. Risk is categorized under four risk categories as described as- normal, low risk, medium risk and high risk. However, for different parameters risk values are calculated separately.

1. Risk Calculation for Permissions:

Permissions are required by the application to acquire sensitive resources. After extracting features from Android applications, they are pre-processed on the basis of score calculation with CVSS(Common vulnerability scoring system)(CVSS,2014) Normal Risk is assigned to permissions required for the normal functionality of applications. Low Risk holds the kind of permissions which can cause little damage to the system. Medium risk depicts the moderate level of threat to the system such as changing system settings and writing data to storage. High risk reveals the privacy and financial impact if permissions are granted. With High-Risk permissions, the attacker is able to charge from users and leak sensitive data from mobile devices. Fig. 1.1 depicts the process of calculating risk factor based on permissions i.e. if a particular set of permissions are used by the application, then it lies under what level of risk factor. Here, initially, for each installed application, it counts the number of permissions required by an application for each risk category. After that, weight of each risk is calculated by dividing a number of permissions required for each category with a number of normal permissions required so that ratio becomes equalized. Then, a maximum of four risk ratios computed is assigned as calculated risk to a particular application. This process repeats for all applications installed on mobile devices.

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319-5940
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 8, Issue 9, September 2019

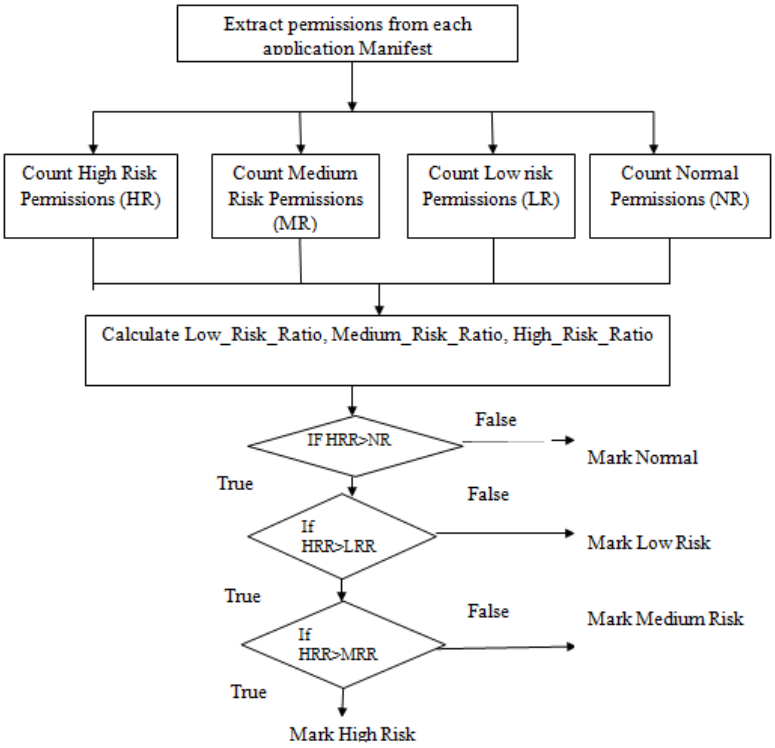

Fig.1. Risk Calculation strategy with manifest Permissions

2. Calculation of Risk Factor for an Application with Captured Network Traffic:

By capturing network traffic statistics such as application id, connectivity, communication interval, data bytes sent or received, it is easily tracked when an application is sending data in an off state. As we know data transferred over the network is in the form of packets. Transferred or received data size for a specific time interval is matched against calculated threshold with naïve variance as shown:-

$$x^2 = \frac{\sum_{i=1}^{N} ai^2 - \left(\sum_{i=1}^{N} ai\right)^2 / N}{N}$$

Here, x represents the threshold value obtained, a_i represent the size of data sent or received and N defines the number of samples which are communicated after some time interval. Fig.1.2 depicts the process of calculating risk-levels with collected network statistics. Initially, the algorithm starts by collecting number of samples of data transferred over network for each communication and for each application. After that, for each communication, the threshold is calculated of transferred packets for each type of application because video streaming applications transfer more data in comparison to web applications. Then for each application, transferred data is measured against threshold if it is more then, tolerance limit of transferred data determines the risk level of application. Tolerance limit can be defined by normalizing the threshold value and transferred data values so that the whole scale of risk calculation becomes equalized.

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319-5940
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 8, Issue 9, September 2019

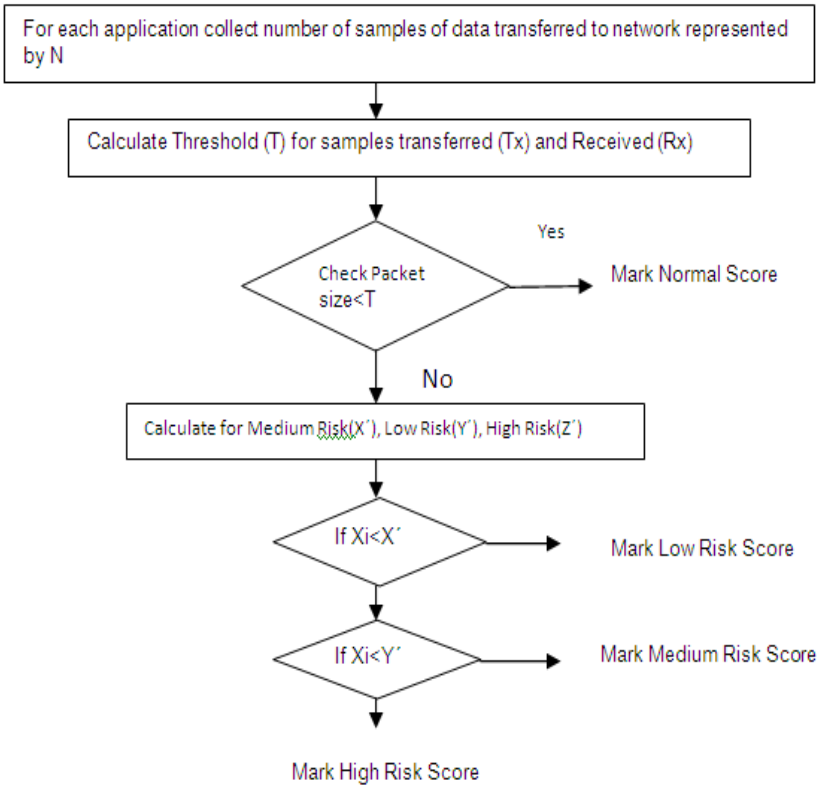

Fig.2. Risk Calculation strategy with Captured Network Traffic Statistics

B. Data Aggregation

This step involves the aggregation of created data set having risk values against applications for each communication interval on network traffic-basis with data-set of calculated risk values on the basis of required permission set by an application. Moreover created data set with two parameters is combined to assign each application a label as-normal, low risk, medium risk and high risk.

IV. Experimental Results and Discussion

The data set created with the proposed method has 59000 instances. Here, data of 59 applications installed on SDK environment are extracted which are communicated for 1000 times. Therefore, 59000 instances are tested on four active learning algorithms to find best classifier using WEKA API. All of the four algorithms are trained with 10 fold cross-validation.

A. Classification Results

1. Logistic Regression:

Our data set contains independent variables. Logistic Regression fits to find a linear relationship between these variables. From the 59000 instances, logistic predicted the class of 56000 instances correctly while 3000 instances are misclassified by the classifier with 94.9% accuracy and 5% error rate.

2. *Radial Basis Function*:

RBF pick a random subset of given data points. It is efficient to predict the data point's class even in cases when data points are not evenly distributed throughout the input space. However, RBF predicted the class of 58000 instances correctly while 1000 are misclassified. RBF shows very accurate results to classify instances belongs to four classes-normal, low-risk medium risk and high risk with 98% accuracy and 2% error rate.

3. Sequential Minimal Optimization (SMO):

SMO is efficient to solve a very large set of QP programming problems by breaking them into a small set of QP problems. Moreover, its computation time is dominated by SVM evaluation and fastest even in case of sparse datasets.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 8, Issue 9, September 2019

It classifies 57000 instances class correctly and 2000 are misclassified.SMO predicted instances with 96.68% accuracy and 3.38% error rate.

4. Naive Baye's:

Naïve Baye's Method is used for training data set as it able to evaluate the approximated algorithm in linear time. It adopts a probabilistic model for classification of data points using Baye's theorem with conditional probability. However, with a small amount of training data, it can efficiently estimate parameters such as means and variance necessary for classification. With given dataset, Naïve baye's predicted very accurate results by classifying 58900 instances correctly while only 100 are misclassified by the algorithm. It shows a 99.83% accuracy and 0.17% error rate.

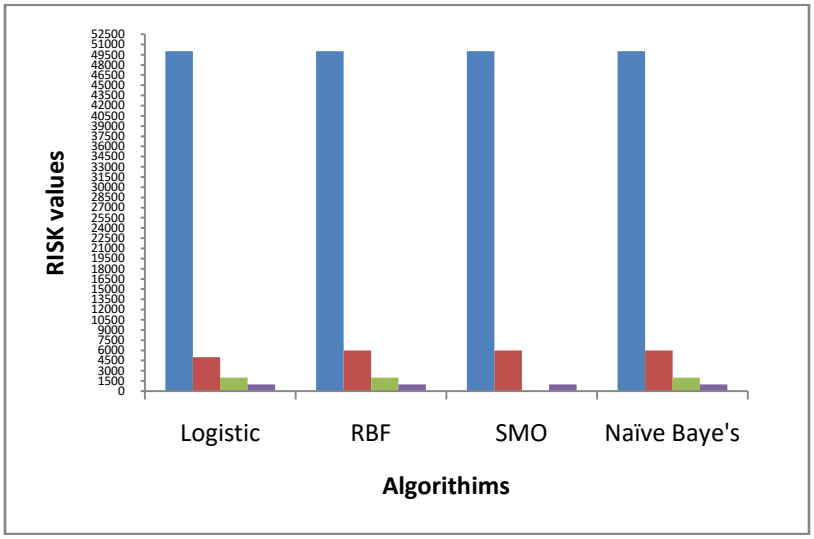

Fig.3. Classification Results of Four Learning Algorithms with given dataset.

B. Performance of Classifiers

Performance of four classifiers is measured in terms of performance metrics as- true positives, false positives, mean absolute error and root mean square error.

1. *True Positives*: True positives represent the proportion of instances that were correctly classified.

2. *Mean Absolute Error:* A quantity used to measure how closely related predicted outcomes class label with the eventual outcomes.

3. *False Positives:* False positives represent the proportion of instances that were misclassified.

4. *Root Mean Square Error:* Root Mean Square error is a frequently used measure of different values predicted by the algorithm and the values actually observed from the environment which is being modeled.

TABLE I COMPARISON OF PERFORMANCE CLASSIFIERS

Algorithm	TP Rate	FP Rate	Mean Absolute Error	Root Mean Square Error	Accuracy
Logistic	0.7085	0.014	0.054	0.0545	96%
RBF	0.995	0	0.0123	0.0554	99%
SMO	0.75	0.095	0.2528	0.3163	96.6%
Naïve Baye's	0.999	0	0.004	0.0145	99.8%

V. CONCLUSION

In this work, a technique is designed to calculate risk-level associated with smart-phone applications available at third-party stores. However, the designed technique is able to extract feature-set from multiple sources by emulating mobile environment. Then, it calculates risk-level for each parameter separately. After aggregation of data collected representing risks associated with each application, learning algorithms are trained to classify risk level of application data. Learning classifiers will classify applications for four classes-normal, low risk, medium risk and high risk. After training and learning data set, work concludes that among four classifiers trained, Naïve Baye's give accurate results with 99% accuracy and very low false positives. Therefore, it can be implemented with a technique to better classify the applications risk-factor. Future research can also be tested on large data-set of applications and analysis can be made to determine the types of mobile malware.

VI. FUTURE SCOPE

The technique can be integrated with efficient classifier for automatic classification. More data set can be used to learn classifiers and to classify number of malicious applications. The study can be done to find more effective feature-sets from benign and malicious applications to get better results. Detection can be performed on more number of samples containing benign and malware applications to effectively train classifiers Analysis can be done after detection to classify the particular types of mobile malware..

REFERENCES

[1]. Wu, D-J, Mao, C-H, Wei, T-E, Lee, H-M & Wu, 'DroidMat: Android Malware Detection through Manifest and API Calls Tracing', *Information Security (Asia JCIS), 2012 Seventh Asia Joint Conference*, IEEE, Tolyo.
[2]. Joshua Abela, K, Kristopher Angeles, D, Raynier, J, Alas, D, Joseph Tolentino, R & Alberto Gomez, M 2013, 'An Automated Malware Detection System for Android using Behavior-based Analysis : AMDA ', International Journal of Cyber-Security and Digital Forensics (IJCSDF), vol 2, no. 2, pp. 1-11.Kim, H, Smith, J & Shin, KG 2008, 'Detecting energy-greedy anomalies and mobile malware variants', MobiSys '08 Proceedings of the 6th international conference on Mobile systems, applications, and services, ACM, New York,NY,USA.
[3]. *Security Symposium (*Yajin, Z, Zhi, W, Wu, Z & Xuxian, 'Hey, You, Get Off of My Market: Detecting Malicious Apps in Official and Alternative Android Markets', Proceedings of the 19th Network and Distributed System NDSS 2012), San Diego, CA.
[4]. Zhou, W, Zhou, Y, Jaing, X & Ning, 'Detecting repackaged smartphone applications in third-party android marketplaces', Proceedings of the second ACM conference on Data and Application Security and Privacy , ACM, New York,NY,USA
[5]. Patil, B. M. (2016). Effective Risk Analysis and Risk Detection for Android Apps. *International Journal of Computer Applications*, *147*(6).
[6]. Moonsamy, V., Rong, J., and Liu, S. Mining Permission Patterns for Contrasting Clean and Malicious Android Applications. Future Generation Computer Systems, 36:122-132, 2014.
[7]. Enck, W, Gilbert, P, Chin, B-G, Cox, LP, McDaniel, P & Sheth, 'TaintDroid: an information-flow tracking system for realtime privacy monitoring on smartphones', OSDI'10 Proceedings of the 9th USENIX conference on Operating systems design and implementation , ACM, Berkley,CA,USA.
[8]. Enck, W, Ongtang, M & Patric, 'On Lightweight Mobile Phone Application Certification', CCS '09 Proceedings of the 16th ACM conference on Computer and communications security, ACM, New York,NY,USA.
[9]. Bose, A, Hu, X, Shin, KG & Park, T 2008, 'Behavioral detection of malware on mobile handsets', MobiSys '08 Proceedings of the 6th international conference on Mobile systems, applications, and services, ACM, New York,NY,USA.
[10]. Xie, L, Zhang, X, Jean-Pierre, S & Zhu, 'pBMDS: a behavior-based malware detection system for cellphone devices', *WiSec '10 Proceedings of the third ACM conference on Wireless network security* , ACM, New York,NY,USA.
[11]. Batyuk, L, Herpich, M, Camtepe, SA, Raddatz, K, Schmidt, A-D & Albayrak, 'Using static analysis for automatic assessment and mitigation of unwanted and malicious activities within Androidapplications', Malicious and Unwanted Software (MALWARE), 2011 6th International Conference, IEEE, Fazardo.
[12]. Blasing, T, Batyuk, L & Schmidt, 'An Android Application Sandbox System for Suspicious Software Detection', Malicious and Unwanted Software (MALWARE), , A-D 2010 5th International Conference, IEEE, Nancy, Lorraine.
[13]. Burguera, I, Zurutuza, U & Simin, ' Crowdroid: behavior-based malware detection system for Android', SPSM 2011 Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices, ACM, New York,NY,USA.
[14]. Chin, E, Felt, AP & Greenwood, 'Analyzing inter-application communication in Android', MobiSys '2011 Proceedings of the 9th international conference on Mobile systems, applications, and services, ACM, New York,NY,USA.
[15]. Enck, W, Octeau, D, McDaniel, P & Chaudhuri, 'A Study Of Android Application Security', SEC'11 Proceedings of the 20th USENIX conference on Security, USENIX Association Berkeley , CA, USA.
[16]. La Polla, M, Martinelli, F & Sgandurra, D 2013, 'A Survey on Security for Mobile Devices', Communications Surveys & Tutorials, IEEE, vol 15, no. 1, pp. 446 - 471.
[17]. Ongtang, M, Enck, W, McLaughlin, S & McDaniel, P 2009, 'Semantically Rich Application-Centric Security in Android', Computer Security Applications Conference, 2009. ACSAC '09. Annual , IEEE, Honolulu, HI.
[18]. P.F. Chan, PKH, & Yiu, SM 2012, 'DroidChecker: analyzing android applications for capability leak', WISEC:Security and Privacy in Wireless and Mobile Networks, ACM , New York, NY, USA.
[19]. Rastogi, V, Chen, Y & Enck, W 2013, 'AppsPlayground: Automatic Security Analysis of Smartphone Applications', CODASPY '13 Proceedings of the third ACM conference on Data and application security and privacy, ACM, New York,NY,USA.
[20]. Samra, AAA, Yim, K & Ghanem, OA 2013, 'Analysis of Clustering Technique in Android Malware Detection', Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), *2013 Seventh International Conference*, IEEE, Taichung.
[21]. Sanz, B, Santos, I, Laorden, C, Ugarte-Pedrero, X & Bringas, 'On the automatic categorisation of android applications ', Consumer Communications and Networking Conference (CCNC), *2012 IEEE*, IEEE, Las Vegas,Nv.
[22]. Schmidt, A-D, Bye, R, Schmidt, H-G & Clausen, 'Static analysis of executables for collaborative malware detection on android', ICC'09 Proceedings of the 2009 IEEE international conference on Communications, IEEE, USA.

[23]. Spreitzenbarth, M, Freiling, F, Echtler, F & Schreck, 'Mobile-sandbox: having a deeper look into android applications', *SAC '13 Proceedings of the 28th Annual ACM Symposium on Applied Computing*, ACM, New York,NY,USA.

[24]. Wikipedia, Weka (machine learning, [online].Available at http://en.wikipedia.org/wiki/Weka_%28machine_learning%29 [Accessed 31st May 2018

[25]. Wikipedia, Logistic regression, [online] Available at http://en.wikipedia.org/wiki/Logistic_regression [Accessed 20June,2018]

[26]. Wikipedia, Sequential minimal optimization [online], Available at http://en.wikipedia.org/wiki/Sequential_minimal_optimization [Accessed 13June,2018]

[27]. Yerima, S, Sezer, S & McWilliam, 'Analysis of Bayesian Classification Based Approaches for Android Malware Detection', *IET Information Security*.

[28]. G. Dini, F. Martinelli, A. Saracino, and D. Sgandurra.,2012. 'Madam: a multi-level anomaly detector for android malware.2012' In Proceedings of the 6th international conference on Mathematical Methods, Models and Architectures for Computer Network Security: computer network security, MMM-ACNS'12, pages 240–253

[29]. H.Wang,Z. Liu, J.Liang, N.V.Rodriguez ,Y.Guo 2018,' Beyond Google Play: A Large-Scale Comparative Study of Chinese Android App Markets', IMC '18, October 31-November 2, 2018, Boston, MA, USA