Secure Way to Spot Malware in Android Applications using Multi-Classification Technique

Pooja B. Kote
Student
Department of Computer Engineering
Sir Visvesvaraya Institute of Technology, Chincholi, Sinnar

Prof. S. M. Rokade
Head of Department
Department of Computer Engineering
Sir Visvesvaraya Institute of Technology, Chincholi, Sinnar

Abstract

Modern malware uses advanced techniques to hide from static and dynamic analysis tools. To achieve stealthiest when attacking a mobile device, an effective approach is required for the diagnosis of the application. In current approach evaluates the android application for the detection of malware as it performs the analysis part by simple code or the pattern combination. The hacker can override this combination of diagnosis of pattern, as a result which may infect the device with the, malware. This paper introduce approach which is using various techniques like patterns, flow based, behaviour based, state based and do analysis of each individual data by its associated specialized algorithms. The results obtained are fused to get the final results of that application. This paper aims to spot malware using various combinations of algorithms and detect the malware. The algorithms that are going to used are Call Graph Based Classification, NN based Classification, and Naive Byes Based Classification. Experimental results show the feasibility and effectiveness of the proposed approach to detect the malware.

Keywords: malware detection, Naive Byes, Call Graph, android, Neural Network

I. INTRODUCTION

In Today's world, Modern malware uses advanced techniques to hide from static and dynamic analysis tools. To achieve stealthiest when attacking a mobile device, an effective approach is required for the diagnosis of the application. Even if designing a malware is nowadays considered quite common the most advanced programmers try to hide malicious behaviours by using different techniques, such as the repackaging of legitimate applications or the obfuscation/ciphering of code. In current approach evaluates the android application for the detection of malware as it perform the analysis part by simple code or the pattern combinations. The hacker can override this combination of diagnosis of pattern, as a result which may infect the device with the malware.

In this perspective, the system aims to spot malware using various combinations of algorithms and detect the malware. The algorithms that we are going to use are Call Graph Based Classification, NN based Classification, Naive-Byes Based Classification Experimental results show the feasibility and effectiveness of the proposed approach to detect the malware .To detect malware three approaches are used first approach require call graph technique to predict the calling relationship between subroutines. Second method is based NN classification i.e. back propagation algorithm and third method used is Bayesian classification represents a supervised learning method as well as a statistical method for classification.

To summarize, the paper has three main contribution as follows:

1) Provide efficient security as all results are fused together to get final result for application.
2) It aims to spot malware using various combination of algorithms and detect malware.
3) To study various classification techniques.

The remaining part of the paper is organized in different section as follows. The literature survey is given in section II. In section III the proposed technique with system architecture is explained. Conclusion is given in Section V.

II. LITERATURE REVIEW

In this section various malware detection techniques is discussed. It includes Karen A. Garcia post-mortem ID, Wei-Wang permission-induced risk based detection.

A. Analyzing Log Files for Post-mortem Intrusion Detection [1]:

In this paper, a novel approach to host-based, post-mortem intrusion detection. In this post-mortem ID is of type anomaly, and is based on a classification method that combines a hidden Markov model (HMM) and *k*-means which we call KHMM.

To build a model for ordinary behaviour, in the first step, we factor out repetitive behaviour in a collection of ordinary (attack-free) log files. As a result, we obtain, first, a compressed version of all log files, and, second, a relation of the sequences of most frequent occurrence across all those logs, which we call across repetitive sequences. In the second step, we follow a 100-size, 100-step sliding-window approach to analyze every reduced log: starting at the first position of the log, we retrieve a window of size 100, then characterize each window by means of an attribute vector and then slide the window a step of 100 elements to continue with the same procedure in a third step, we build a model that captures the commonality in the sequence of attribute vectors, representing the original log.

B. Exploring Permission-Induced Risk in Android Applications for Malicious Application Detection [2]:

This paper introduce, framework to identify malapps through analyzing the permission usage patterns, as intuitively an app's behavior is characterized by the permissions it requests.

- First, it employs three feature ranking techniques to evaluate the risk of granting each permission, based on which the permissions are ranked from most to least risky.
- Second, permission sets, instead of individual permission, are evaluated by feature subset selection methods for investigating the risk introduced by the collaboration of several permissions.
- Third, the detection of malapps based on risky permissions is formulated as a classification problem and executed by building classifiers. Last, in order to explicitly characterize the risk caused by permission requests and use it to report malapps, detection rules are extracted from malapps detectors, then employ the detection rules to detect unknown malapps.

C. A New Android Malware Detection Approach Using Bayesian Classification[3]:

In this paper, perspective approach based on proactive method aimed at uncovering known families as well as unknown malware so as to reduce incidents of malware in marketplaces from evading detection. The Bayesian classification based approach can complement signature-based scanning thus enabling harmless apps obtained from Android marketplaces to be verified whilst isolating suspicious samples for further scrutiny.

Bayesian classification is solution for the problem filtering large amounts of apps as it can perform relatively fast classification with low computational overhead on curtained. Android applications is the ability to model both an 'expert' and 'learning' system with relative ease compared to other machine learning techniques. Bayesian method allows the incorporation of prior probabilities(expert knowledge) even before the training phase. This hybrid property can be exploited to optimize the classifier's performance without incurring additional computational overhead. However, in this research, only its applicability as a pure 'learning' method is explored.

D. Detection and identification of Android malware based on information flow monitoring [4]:

To detect an application infected with a known malware, we look for a match between the information flows involving its data and the edges of the malware profiles.

The two following conditions must be fulfilled to consider that a flow matches an edge. First, the information flow and the edge must involve the same information. Second, the source and destination nodes of the edge must describe the source and destination containers of the flow. If these conditions are fulfilled, we consider that the analysed application is infected with the malware of which profile enables us to find the match.

Therefore consider the application as infected with the malware of which profile enables us to find the match.

E. Catch me if you can: Evaluating Android anti- malware against transformation attacks [5]:

In this paper approach is to evaluate anti-malware products for Android regarding their resistance against various transformation techniques in known malware. For this purpose, approach developed Droid Chameleon, a systematic framework with various transformation techniques.

- Gives methodology through the series of transformations applied to Droid Dream and Fake player samples and the detection results on various anti-malware tools. Empty cells in the tables indicate positive detection while cells with 'x' indicate that the corresponding anti-malware tool failed to detect the malware sample after the given transformations were applied to the sample. Each transformation is applied to a malware sample and the transformed sample is passed through anti-malware. If detection breaks with trivial transformations, then stop.
- Next, it applies all the DSA transformations. If detection still does not break, we apply combinations of DSA transformations. In general there is no well-defined order in which transformations should be applied.

III. PROPOSED WORK

In this section, the brief discussion on proposed theory followed by proposed architecture.

The system takes android .apk file as an input and then decompiles it. After decompilation process, Parser takes code as an input performs parsing to build data structure to give structural representation of input checking for correct syntax in the process. In next phase feature extraction which transforms the large data input into reduced data set called feature vector. Then will apply three classification techniques to this feature vector. Once done with classification, analysis of each one with different algorithms/methodology is to be made such as Behavioural analysis, neural network, Naïve-byes. Then collect result from all three and fused it. Lastly, will get final output that is malware detected in android app.

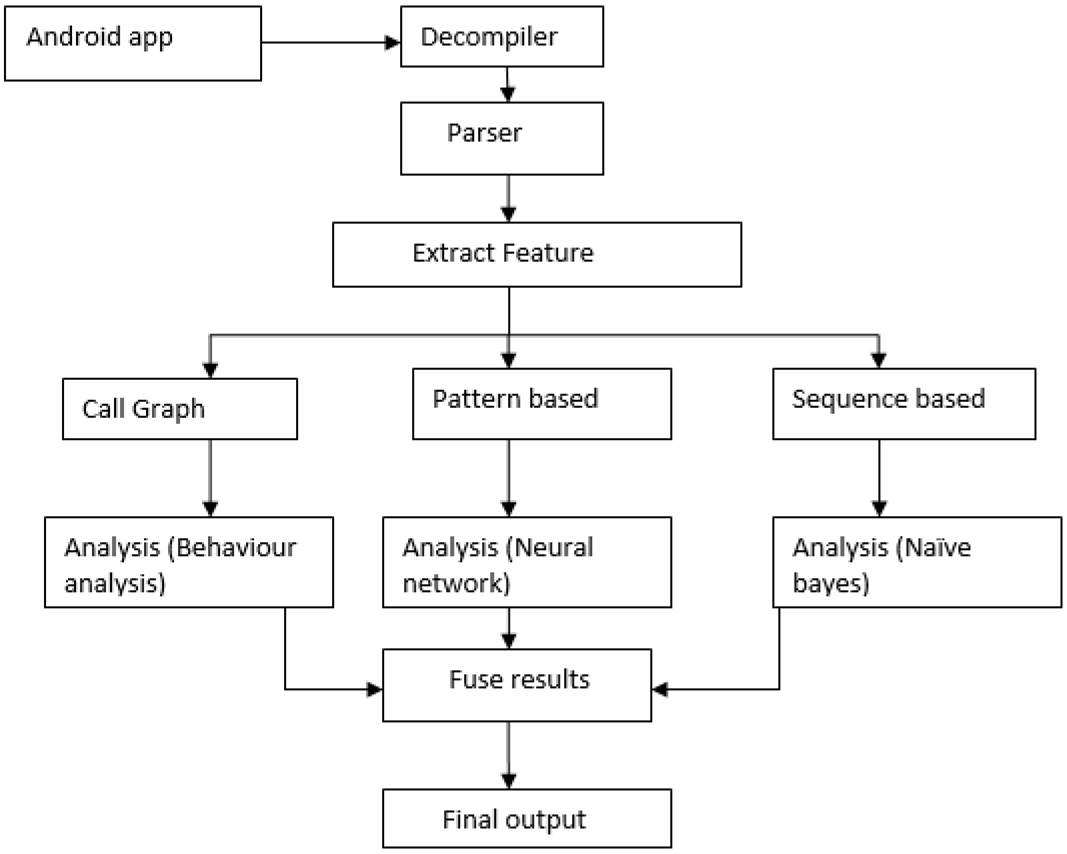

Fig. 1: Proposed Architecture

- Decompiler: In this phase it takes android app as an input and transforms it into code and decompiles it.
- Parser: It take code as input and perform parsing build data structure, giving structural representation of input, checking for correct syntax in the process.
- Feature extraction: It transforms the large data input into reduced data set called feature vector. Will apply classification technique to this feature vector.
- Call Graph based: In this technique will get graph representing relation between subroutines in code. Behavioural analysis will be used.
- Pattern based: In this technique classification based on pattern matching. Neural network algorithm will be used.
- Sequence based: In this technique sequence is transformed in feature vector and each sequence will have class label. Naive byes algorithm will be used.
- Analysis**:** The methods or algorithm will be applied to each of three techniques:

1) Behavioral analysis: It considers flow of function call and display function call graph.
2) Neural network: It is back propagation algorithm divided into two phases: Propagation and weight update.
3) Naïve byes: It is probability based algorithm. It consider feature vector and assign class label.

- Fuse result: In this phase it will fuse result from all three techniques and will contain graph of recall and precision of each technique
- Final output: Lastly final result that will give malware if present in app.

IV. CONCLUSION

The proposed system provides an effective approach as three classification technique such as Call-graph, Naïve-byes and NN based is used to spot malware in android application. All three results is fused for better performance, by this it helps to spot complex malware in application. It is generalized approach can be applied to any android application. The proposed approach is more reliable as multi-classification is used to detect malware and can be used in any android device. This approach extended with more additional feature or information to increase accuracy. The work can be extended to spot malware on direct application running on mobile device.

REFERENCES

[1] Karen A. Garc´ıa, Ra´ul Monroe, Luis A. Trejo, Carlos Mex-Perera, and Eduardo Aguirre, "Analyzing Log Files for Post mortem Intrusion Detection," IEEE transactions on systems, man, and cybernetics—part c: applications and reviews, vol. 42, no. 6, November 2012.
[2] Wei Wang, Xing Wang, "Exploring Permission-Induced Risk in Android for Malicious Application Detection," Information forensics and security, vol. 9, no. 11, November 2014..
[3] Suleiman Y. Yerima, Sakir Sezer, Gavin McWilliams, "A New Android Malware Detection Approach Using Bayesian Classification" IEEE 27th International Conference on Advanced Information Networking and Applications, 2013.
[4] R. Andriatsim and efitra and V. V. T. Tong, "Detection and identification of Android malware based on information flow monitoring," in Int. Conf. on Cyber Security and Cloud Computing, 2015, pp. 1–4.
[5] V.Rastogi, Y. Chen, and X. Jiang, "Catch me if you can: Evaluating Android anti- malware against transformation attacks," IEEE Trans. On Information Forensics and Security, vol. 9, no. 1, pp. 99–108, Jan. 2014.
[6] Shabtai, U. Kanonov, Y. Elovici, C. Glezer, and Y. Weiss, "Andromaly: behavioural malware detection framework for Android devices," Journal of Intelligent Information Systems, vol. 38, no. 1, pp. 161–190, 2012.
[7] Bose, X. Hu, K. G. Shin, and T. Park, "Behavioural detection of malware on mobile handsets," in Proc. Int. Conf. on Mobile Systems, Applications, and Services, 2008, pp. 225–238.
[8] P. Faruki, V. Ganmoor, V.Laxmi, M. S.Gaur, and A. Bharmal,"AndroSimilar: robust statistical feature signature for Android malware detection," in Proc. Int. Conf. on Security of Information and Networks,2013, pp. 152–159.
[9] W. Mazurczyk and L. Caviglione, "Information hiding as a challenge for malware detection," Security & Privacy, vol. 13, no. 2, pp. 89–93, 2015.