ANALYSIS OF ANDROID MALICIOUS SOFTWARE DETECTION TECHNIQUES AND TOOLS

Dr.C.Shanthi
Associate Professor,
Department of Computer Science VISTAS, Chennai, Tamil Nadu, India.
Dr.K.Sharmila(Corresponding-author)
Associate Professor,
Department of Computer Science VISTAS, Chennai, Tamil Nadu, India.
: **Dr.R.Devi**
Associate Professor,
Department of Computer Science VISTAS, Chennai, Tamil Nadu, India.
Dr.J.Jebathangam
Associate Professor,
Department of Computer Science VISTAS, Chennai, Tamil Nadu, India.
E-mail: shanthi.scs@velsuniv.ac.in, , devi.scs@velsuniv.ac.in, jthangam.scs@velsuniv.ac.in
Corresponding-author : sharmila.scs@velsuniv.ac.in

ABSTRACT

The custom of Smartphone is expanding the prevalence of Android is additionally expanded very fast. At present Android offers a cumbersome figure of uses in liberated from cost to be downloaded and accessed. The private and classified data put away in Android phones are more vulnerable to malware. Here the investigations of various types of malware and their identification methods with advantages and disadvantages and their possibility scope. To discriminate malware from a great many Android applications muddled static and dynamic investigation apparatuses to consequently distinguish and characterize pernicious applications. It is thusly imperative to imagine significant methods to look at and distinguish these dangers. This article presents an extensive stage on driving Android malware examination and disclosure techniques, and their sufficiency against creating malware. The danger introduced by convenient malware convinces the progression of powerful and exact assessment strategies. The system can be improved to redesign security and exactness later.

Keywords: Android, Mobile malware detection, Dynamic code loading, Static analysis, Dynamic analysis.

INTRODUCTION

The mobile devices have been increasing and manipulators can simply make banking, shopping, webpage navigation, gaming, and other comparable transactions without computers. In Android platform applications have become the objective of vindictive programming engineers in light of the fact that the Android platform allowed and accessed open-source working framework, and at whatever point an application is included the Google Play Market, the application isn't inspected in detail turning out to be progressively mainstream and different associations have built up an assortment of uses to outfit as per advertise patterns.

Malware or Malicious software that depicts any malevolent program or code that is insensitive to frameworks. Examining malware is a procedure that requires making a couple of strides. Perhaps the most straightforward approaches to survey a dubious program is to examine it with completely computerized apparatuses. Malware identification and classification are difficult issues, particularly on mobile platforms.

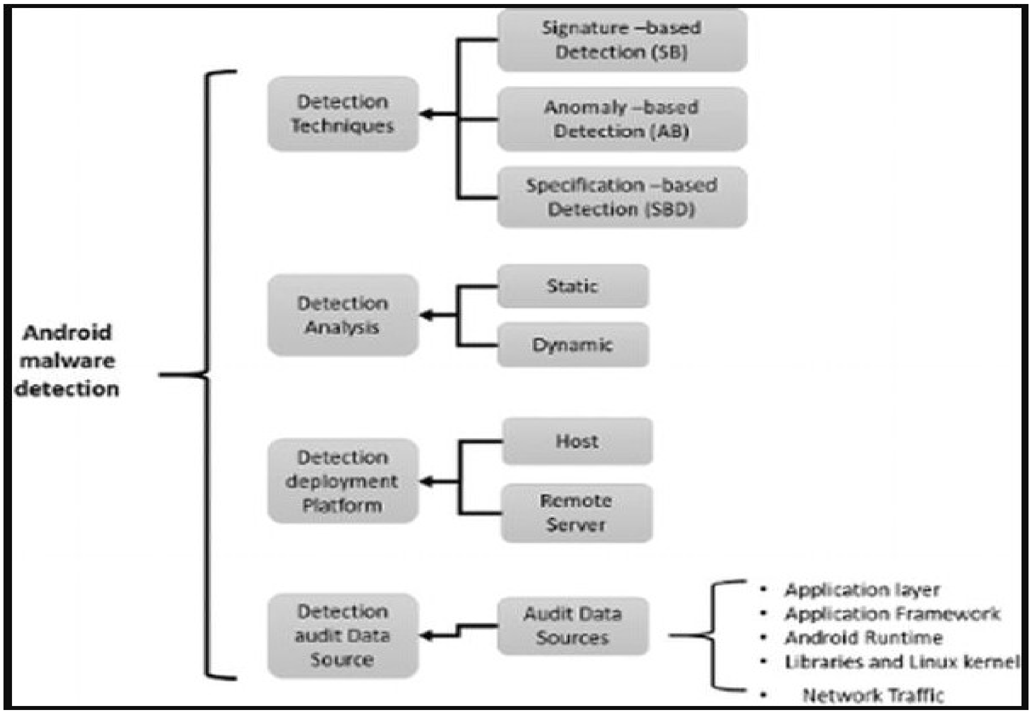

Android malware detection
Detection Techniques
Signature –based Detection (SB)
Anomaly –based Detection (AB)
Specification –based Detection (SBD)
Detection Analysis
Static
Dynamic
Detection deployment Platform
Host
Remote Server
Detection audit Data Source
Audit Data Sources
• Application layer
• Application Framework
• Android Runtime
• Libraries and Linux kernel
• Network Traffic

Figure 1: Approaches and Classification of Android Malware Detection

In the above figure three detection techniques has been classified: signature-based (SB), anomaly-based (AB), and specification-based (SPB) detection. Inconsistency based location screens standard activities in the gadgets and searches for any lead that goes wrong from the ordinary model. Practically identical to AB location, SPB identification furthermore screens for any deviation yet rather than perceiving the occasion of unequivocal assault designs; it screens for deviation of their lead from the conventional detail.

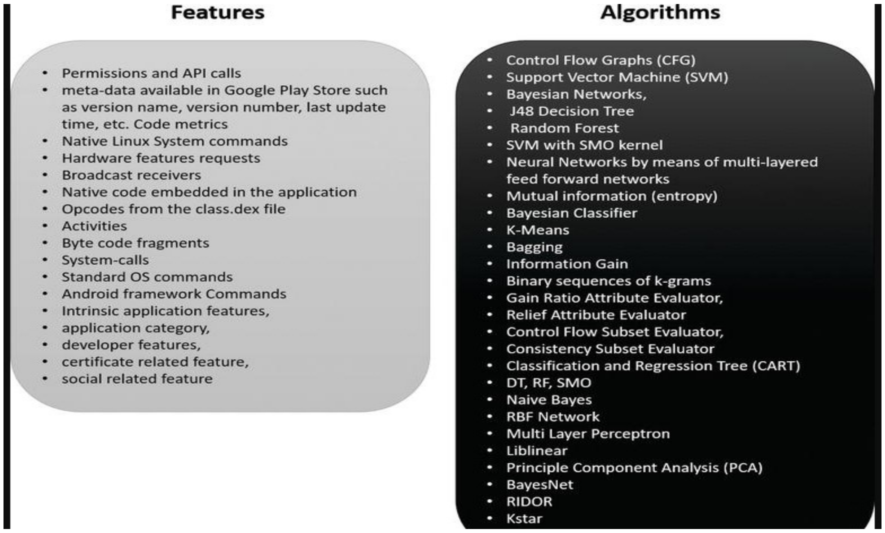

Features
Permissions and API calls
meta-data available in Google Play Store such as version name, version number, last update time, etc. Code metrics
Native Linux System commands
Hardware features requests
Broadcast receivers
Native code embedded in the application
Opcodes from the class.dex file
Activities
Byte code fragments
System-calls
Standard OS commands
Android framework Commands
Intrinsic application features,
application category,
developer features,
certificate related feature,
social related feature
Algorithms
Control Flow Graphs (CFG)
Support Vector Machine (SVM)
Bayesian Networks,
J48 Decision Tree
Random Forest
SVM with SMO kernel
Neural Networks by means of multi-layered feed forward networks
Mutual information (entropy)
Bayesian Classifier
K-Means
Bagging
Information Gain
Binary sequences of k-grams
Gain Ratio Attribute Evaluator,
Relief Attribute Evaluator
Control Flow Subset Evaluator,
Consistency Subset Evaluator
Classification and Regression Tree (CART)
DT, RF, SMO
Naive Bayes
RBF Network
Multi Layer Perceptron
Liblinear
Principle Component Analysis (PCA)
BayesNet
RIDOR
Kstar

Figure 2: Static Analysis features and algorithms that are used to process research approaches.

The two categories of examination could be performed on behaviour-based recognition, that is, static analysis and dynamic analysis. Static analysis can get highlights subject to examination on groupings of headings procured using sorting out. In a word, static analysis has the potential gains of profitability and high code incorporation, and it is a generally lightweight procedure when contrasted with the dynamic analysis. The three procedures originate in the composing used as examination methodologies for Android pernicious programming.

Unfortunately, static and signature-based analysis techniques can be avoided by malware applications utilizing strategies, for example, polymorphism, transformation, and dynamic code stacking.

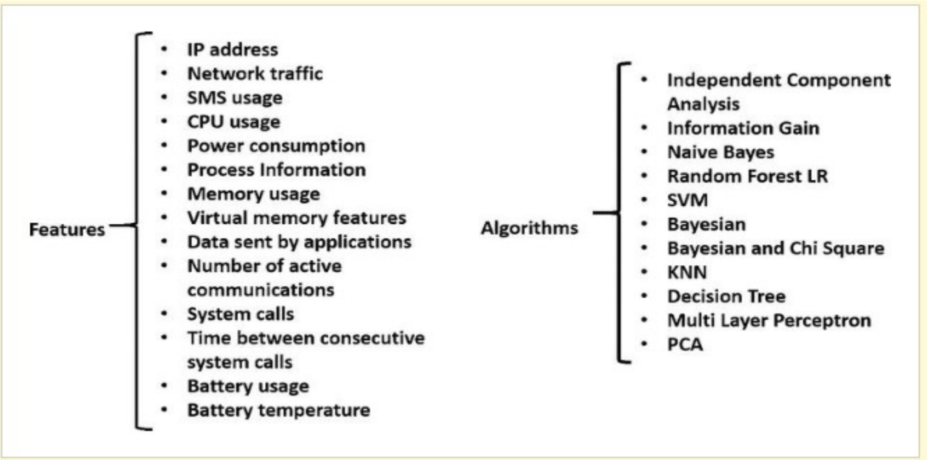


Figure 3: Dynamic Analysis features and algorithms that are used to process research approaches.

The dynamic analysis can separate item code disarray somewhat by checking the run-time conditions of an application, which runs in a secured sandbox mode, yet it entails a great deal of figuring resources and has low code consideration.

DETECTION TECHNIQUES

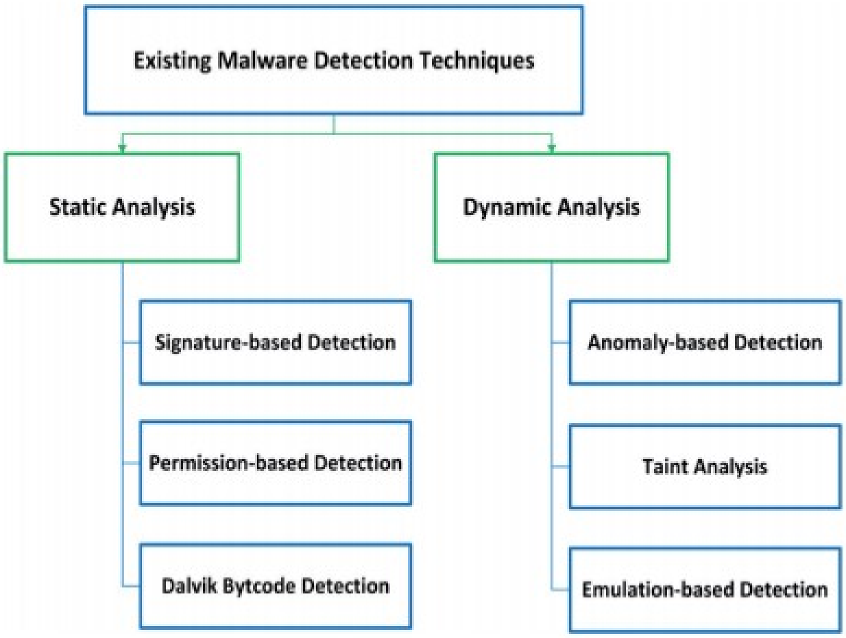

Existing Malware Detection Techniques
Static Analysis
Dynamic Analysis
Signature-based Detection
Permission-based Detection
Dalvik Bytcode Detection
Anomaly-based Detection
Taint Analysis
Emulation-based Detection

Figure 4: Malware detection techniques

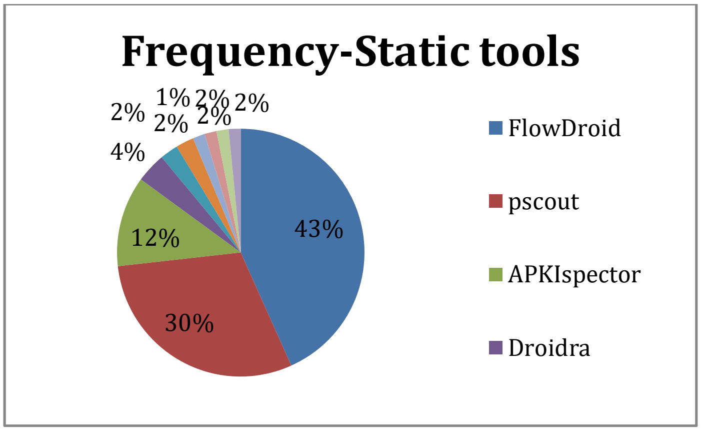
Figure 4 shows the vibrant flow of Malware detection Technique, Static Analysis and Dynamic Analysis. A malware signature is made by eliminating matched models from an example. Nevertheless, this philosophy has at any rate two critical disservices. In any case, this system is deficient for recognizing dark perils, that is, zero-day attacks, as no recently produced signature could exist.

This is costly as additional techniques are required to recognize the danger, make another signature, and scatter it. Second, malware can without a doubt avoid signature-based distinctive verification by changing small amounts of its item without affecting the semantics.

Signature-based detection which depends concerning set aside for known malwares doesn't take the option to distinguish unidentified malware signatures. The permission- based detection, accommodates application which could be idea of, erroneously, as a malware on account of the trivial collection of the referenced approvals from the first and the malware application. Every one of the recently referenced instruments focuses on in any event one feature.

Static Analysis

Static analysis takes a gander at a program without executing any code. Even though it may reveal every comprehensible method of execution, there are a couple of requirements. In this investigation system, Android applications are analysed and the features found in the archive prior to presence on the device.

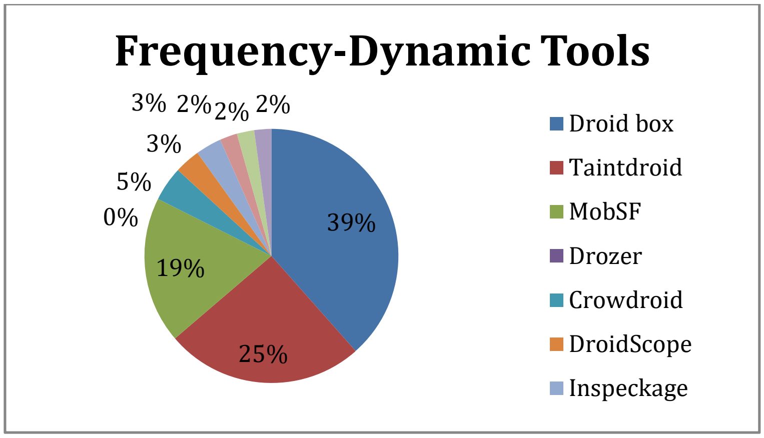


Graph 1: Frequency- static Tools

In Graph 1 the statistical study shows the methods, occurrence of the used approaches, datasets, and tools in the existing systems. The graph represents the tools which has developed frequency usage for static tools.

Dynamic Analysis Method

The dynamic analysis methodology is used to recognize malicious programming by coordinated instances of employments. The attested framework traffic instances of dissimilar applications have similar models with a similar limit, and the consequences avowed the case.



Graph 2: Frequency- static Tools

In Graph 2 represents the tools which have higher frequency usage for Dynamic tools. To detect malicious applications a system is developed by tools used to detect network activity. Alternatively, a wider set of available tools used to detect application features may be analysed statically or dynamically or classify, malicious applications. To improve the efficiency and detection model, a compression procedure is also used for tools which are easy to detect features for the malware.

A framework is created to identify malicious applications by devices used to recognize network movement. On the other hand, a more extensive arrangement of accessible tools used to distinguish application highlights might be examined statically or dynamically or classify, malicious applications. To improve the efficiency and detection model, a pressure technique is likewise utilized for tools which are anything but difficult to detect features for the malware.

Conclusion

The elucidation is to keep and protect Android customers and the devices used, must be secured from the risks of different security attacks. The consolidated stages and the method for downloading Android applications, to deliver the source code and decoding them, also screened to remove static analysis or dynamic analysis important features to be applied and removed. Suitable data valuation and data mining techniques could be applied to take a gander at the application and gathering it as liberal or malware with high precision. The malware revelation organization could be executed and given similarly as a versatile application that will grant the checking results to the customer in a neighbourly way.

REFERENCES

1) Y. Duan, M. Zhang, A. V. Bhaskar, H. Yin, X. Pan, T. Li, X. Wang, and X. Wang,“Things you may not know about android (un) packers: a systematic study based on whole-system emulation,” in 25th Annual Network and Distributed System Security Symposium, NDSS, 2018, pp. 18–21.
2) “Alan-android-malware.com,” http://seclist.us/alan-android-malware-evaluating-tools-released. html, April 2019.
3) W. Xu, Y. Qi, and D. Evans, “Automatically evading classifiers,” in Proc. Of NDSS, 2016.
4) S. Wang, Z. Chen, L. Zhang, Q. Yan, B. Yang, L. Peng, and Z. Jia, “Trafficav: An effective and explainable detection of mobile malware behavior using network traffic,” in Proc. of IWQoS. IEEE, 2016, pp. 1–6.

5) “Virusshare.com,” https://virusshare.com/, December 2017.

6) Y. Tsutano, S. Bachala, W. Srisa-an, G. Rothermel, and J. Dinh, “An efficient, robust, and scalable approach for analyzing interacting android apps,” in Proc. of ICSE, Buenos Aires, Argentina, May 2017.

7) F. A. Narudin, A. Feizollah, N. B. Anuar, and A. Gani, “Evaluation of machine learning classifiers for mobile malware detection,” Soft Computing, vol. 20, no. 1, pp. 343–357, 2016.

8) Alenezi M, Almomani I. Empirical analysis of static code metrics for predicting risk scores in android applications. In: Proceedings of the 5th Symposium on Data Mining Applications (SDMA2018); 21-22 March, 2018; Riyadh, KSA.

9) Chen J, Wang C, Zhao Z, Chen K, Du R, Ahn G. Uncovering the face of android ransomware: Characterization and real-time detection. IEEE Transactions on Information Forensics and Security. 2018;13(5):1286-1300.

10) Arnatovich L, Wang L, Ngo N, Soh C. A comparison of android reverse engineering tools via program behaviors validation based on intermediate languages transformation. IEEE Access. 2018:12382-12394. DOI: 10.1109/ access.2018.2808340.

11) Abubaker H. Analytics on malicious android applications. International Journal of Advanced Software Computer. 2018;10:106-118. ISSN 2074-8523.

12) Almomani I, Alkhayer A. Android applications scanning: The guide. In: Proceedings of the IEEE International Conference on Computer and Information Sciences (ICCIS); 3-4 April 2019; Saudi Arabia. Jouf: IEEE; 2019.

13) Doğru, İ. A., & KİRAZ, Ö. (2018). Web-based android malicious software detection and classification system. *Applied Sciences*, *8*(9), 1622.

14) www.intechopen.com