International Journal of New Technology and Research (IJNTR)
ISSN:2454-4116, Volume-2, Issue-12, December 2016 Pages 47-53

A Survey on Android Malware Detection

Nirmala Yadav,Aditi Sharma,Amit Doegar

***Abstract*— Malwares are spreading around the world and infecting not only the end users but also large organizations and service providers. Android operating system seems to have attracted the most attention from malicious code writer due to its popularity. Earlier, Signature based detection techniques were used to detect unknown malwares. But it was insufficient because these techniques were not able to detect unknown malwares (0-day attack). To analyze the malwares, static and dynamic techniques are used. Static analysis has advantage of being undetectable, as malware cannot modify its behavior during analysis. Despite number of detections and analysis techniques are in place, high detection accuracy of new malwares are still a critical issue. This survey paper highlights the existing detection and existing analysis methods used for the android malicious code.**

***Index Terms*— Android malware,Machine Learning, Mobile Application, signature based.**

I. INTRODUCTION

The rapid growth of mobile devices and remarkable advances in 4G/5G mobile networking technologies have both inspired and facilitated by many mobile applications. In support of these mobile devices, various mobile operating systems have been developed, such as Android, iOS, and BlackBerry, etc. Among these mobile operating systems, Android has become the most popular one, because of its light weight, cost effective, open source and numerous mobile apps it provides. Unfortunately, smartphones running Android have been increasingly targeted by attackers and infected with malicious apps. According to the Kaspersky mobile threat report of 2015 is that the48.15% of attacks targeting financial data (Trojan-SMS and Trojan-Banker), Most Risky threat were ransomware, and the Number of infected smartphone were almost three times that of in 2014[26]. A compromised smart phone can cause severe damages to both users and the service provider.

Nirmala Yadav, PG Scholar/ Department of CS, NITTTR, Chandigarh, India
Amit Doegar, Assistant Professor/ Department of CS, NITTTR, Chandigarh, India.
Aditi Sharma, Technologist, DSD, Chandigarh, India
.

The Android Detection techniques are based on Signature based, De-compilation based, Rule based, and Machine learning techniques. Signature based techniques uses the characteristic, properties or signature of the malware for the detection of malicious codethat takes almost 48 hours to detect new malware and fails at the time of unknown malware. De-compilation based technique uses de-compilation of android app which is used for the recovery of source code and then applies the semantic patterns, control flow, structure flow and data flow analysis. But the missing of control and data flow increases the false alarm. Some good application used bad coding practice, this technique is not able to differentiate good application and malevolent application. Rule based certification technique checks the presence of undesirable operations in application suspected as malicious. It starts from general functionality requirement and then analyzed the required permissions that can create the conflicting operation which are used by malicious program. The rule based certification application must be run all the time in order to verify the status of downloaded application. Machine learning techniques is based on extracted features. These features are extracted from manifest.xml (definition file) and .dex (code based) file. The learned features are used as an indicator of malicious application. The accuracy of machine learning algorithm is depend on the classifier method used. This approach is automated and can enable the static detection of malware application.

II. RELATED WORK

In this section gives the overview of android architecture and the survey the machine learning malware detection techniques will be highlightedaccording to survey papers of android malware, non-android based papers, permission based and the papers that are based on both manifest and code based file.

A.Overview of android

Android is an operating system which uses Linux as its base. Android app developers develop apps in Java and control their operation using Java Libraries designed by Google. Android software stack has number of different elements, shown in Figure 1.1.

Software stack is a collection of Linux kernel and libraries of C/C++ that are exposed through app framework. This app framework provides different services and management at run time. Core services are handled by Linux kernel. It also provides an abstraction between hardware and remainder of the stack. Libraries includes number of distinct C/C++ core libraries such as libc and SSL. Android Run Time includes Dalvikvirtual machine and core libraries. App Framework provide classes used to create Android apps, generic abstraction for hardware access, manages user interface and app resources. All types of Android apps are built on app layer. Both native and third party apps uses the same type of API libraries.

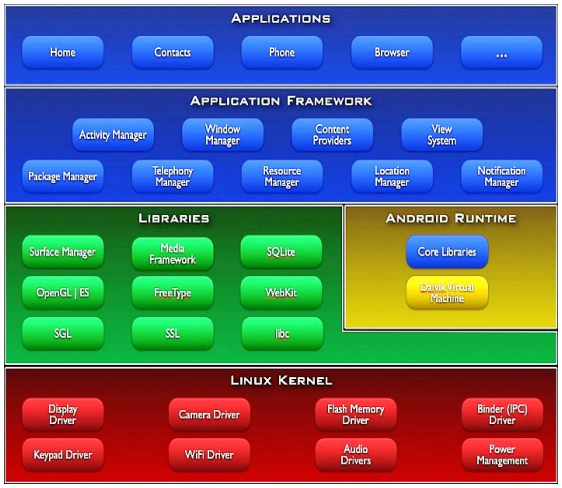

Figure 1.3 Android System Architecture [8]

B.ANDROID MALWARE SURVEY

The security of Android Apps has become a critical issue due to which it becomes active field of research. As with the detection of the first malicious programs, now a days a number of new malwares are being detected every day. This section covers Android malware survey papers.

There are several related work, Author **[17]** provided one of the first survey on mobile malware and analyzed a total of 46 samples. These samples were of iOS, Symbian and Android and they were collected from 2009 to 2011. Main target of attacker were the Android Smartphone market. There were many reasons to write a mobile malware because of their incentives. Author presented the current and future incentives and also examined the cause of incentives. [15]Evaluated Android malware to find out botnet behavior. The purpose of identifying botnet behavior was to find specific trends and characteristics. These trends had extracted from android code and structure. As a result, Author identified characteristics and explored them in terms of Android botnet invention process. This process includes infection, propagation and execution of malware. Finally this process lead to botnet maturity model for Android.[13]Focused on the Android platform and characterize existing Android malware in 49 distinct malware families. Authors performed a time line study to characterize various malwares based on their installation process, activation mechanism and nature of payloads. The data set however contains almost 1260 Android malware samples being collected between 2010 and 2011. Based on the evaluation of four antivirus software, they achieved best case detection rate 79.6% and worst case detection rate 20.2%.[7]Discussed the recognition and analysis of Android malware classes. For analysis authors collected 1485 malware samples from 58 malware families. The author selected the characteristic of Android malware from end users and proposed a solution as recommendation to users before installing apps. The recommended result was the ultimate desire to mitigate the damage happened with Android users. [3]Suggested that a deep knowledge of characteristics of malware is the initial step to prevent from many unwanted consequences present in the app. The rapid increase of malicious apps at Google play store created new possibilities of threat for Google and end user. Author discussed the number of default security mechanism provided by Google for Android Apps. Author also focused on properties found in familiar malware apps, and also reviewed the mitigation technique of android malware.

C.MALWARE DETECTION TECHNIQUES

This section covers non-android based malware detection literature. Authorexamined 45 different malware detection techniques and also presented the scope to compare these detection techniques to determine decision rules. These decision rules lead to secure app development system. Though quality of these system depends on the utilized technique [23]. The previous malware protection researches were not based on malicious activities. They were depended on the functionality limitations of mobile phones. Author discussed the gap between these two and threw some light on present detection and analysis techniques and their pros and cons. Author also discussed how one can improved upon these techniques in current malware detection and analysis techniques[8]. Authorshowed that solution available for malware detection traditionally based on signature-based techniques. But these techniques were declined due to some obfuscation techniques utilized by malware rese and utilized De-obfuscation and Unpacking technique as anti-evasion approach of malware. De-obfuscation

techniques finds the status of obfuscation and unpacking technique is the process of analyzing at the code that gives the exact information of dynamic behavior.Author also used "bi-feature technique" rather than static Mono-features analysis [11].

D. MANIFEST FEATURES BASED ANDROID APPS ANALYSIS

This section covers Authorsused executable to find out the function calls present in code. The readelf command is utilized to extract these function calls. The obtained function call list is correlated with malware executables for classification. Author used Partial Decision Tree algorithm, Prims and KNN. Further they showed a combined malware detection approach to enhance the results[21]. Authorsdissecting the API calls of 940 apps and the causes of over privileging in Android apps; author found that app writer tried to follow minimal privilege set but sometimes failed due to errors that could be attributed to insufficient API documentation. Author of this paper build a weapon to detect over privilege features in Android apps. Author used his weapon "Stowaway" on almost 940 apps. As a result author found that one third apps were over privileged[18]. Authorestablished a detailed mapping of Android API calls to required permissions of Android apps. In this experiment author discovered that almost all developer were not aware of using correct permission set. This was due to lack of security awareness of Android apps. The apps were either had over-specify permission set or under-specify permission set. This study was based on large number of 141,372 Android apps [16]. Authorproposed an approach for categorizing Android apps. The categorization was based on machine learning techniques. The proposed method extracted different permissions present in each app. Permissions set was the main focus of this study [14]. PUMA was based on permission usage to detect malware in Android. The author of this paper analyzed only permissions set features. PUMA is based on machine learning technique which provided high accuracy by encompassing permissions only [9]. Authorsdeveloped a two-layered permissions-based detector to detect malicious Android apps. Author of this paper also compared their method with the previous by considering requested permission set as additional factor. The used permissions features were used to improve detection accuracy. To evaluate their approach, author used 28548, 1536 non malicious apps and malicious apps respectively [5].

E. MANIFEST AND CODE BASED DETECTION AND ANALYSIS

Authors [19]introduced ded decompile, which recovers Android app source code by utilizing installation image directly. The author of this paper used a static investigated 21 million lines of recovered instructional code to design and study of Smartphone apps. It uncovered both dangerous functionality and vulnerabilities like message of phone identifiers, botnet characteristic or use of advertisement libraries by studying 1,100 popular free Android apps. DroidMat[12]extracted features were used permissions, deployment of components, API calls and intent message passing to find out the behavior of Android apps. Author proposed and developed a system known as Droid Mat. The extracted the information like requested permission, intent message passing are from manifest file and API calls are from code based file. Then applied K-means clustering and KNN classification technique. Clustering is used to enhance the malware modeling capability and classification is used for classify the application benign and malicious. Author [10]presented an effective Bayesian classification approach to handle Android malware. Author developed and analyzed an approach which is independent of signature based technique. Author used static analysis, and large data set to uncover previously unknown Android malwares. Static analysis was based on permission detector, command detectors and API calls detectors Author analyzed total 2000 apps, 1000 from benign data set and 1000 from malicious data set. Out of 2000, 1600 were used for training and 400 (200 each from benign and malware samples) for testing the designed model. The implemented approach evaluated against real malware samples.Authors [4]proposed Drebin, a light weight method. This method able to identified malicious apps directly on Android platform. This paper used static analysis technique and collected as many as possible features of an app. For the evaluation author collected 123,453 total apps, among which 5,560 are malicious apps. It detected 94% of the malicious apps with lower FPR. In addition to this author of this paper provided explanation to each relevant property of the detected malware. Drebin required on an average 10 seconds for an analysis.Authors developed android malware detection model using SVM. The author used similarity score which was calculated in term of API calls of every malicious and non-malicious android app. The similarity score is the multiplication of IDF(inverse document frequency) and TF (term frequency). This score and requested permission became the final feature set that is used for analysis. The accuracy of this model was 86%. This study is based on small number of android apps. With the increase of data set, score

calculation overhead also increased [2].The system ICCDectector was Based on Inter Component Communication patterns. The SVM classifier was trained with ICC patterns that are extracted from benign and malicious apps before the outputs of detection model. ICC patterns were component (activity, service, broadcast and receiver), intents (explicit, implicit) and intent filter. For the evaluation author collected 5264 malware and 12026 benign samples. Performance of ICCDetector is compared with the benchmark technique which detects malware according to required permissions. Its accuracy is 10% higher than the benchmark technique [1].

F.INFERENCES

This section covers critical facts extracted from literature. Signature based techniques fails to detect unknown malwares (0-day attack) [12]. Malware damages an infected device within seconds, so signature based techniques are not more effective [3]. Static analysis is more efficient than dynamic approach, as malware cannot modify its behavior during static analysis [18]. Dynamic analysis cannot be performed on the smartphones themselves to directly identify malware because they usually incur a large amount of computational overhead [2]. Permission-based model is not only sufficient to identify malicious apps. Both Manifest and Code properties are required to find the exact behavior of an app [9]. Reverse engineering (in case of Android) is a beneficial process to extract the features. It provides the directory structure and usable files (as dex files, AndroidManifest.xml files, and smali codes) [11]. Machine learning technique is capable to detect new malware [10]. Hybrid classifier gives more accurate result than single classifier [12].

III. CONCLUSION

The growing rate of Android malware created a difficulties in life of Android users. User feels insecure as with risk like hanging of phone on receiving a call, personal information stealing (contacts, pictures, video etc), and large amount of bill while connecting to internet and many more. The available Android malware detection approaches has not been able to provide better accuracy. Most of approaches are based on permission-set only which was insufficient to detect new Android malware. Few approaches consider few code properties but they were not able to provide good accuracy.

Table1: Survey of existing work

[Reference No.], Author`s Name [Year]	Technique used	Advantages	Disadvantages
[1] K. Xu.et al. [2016]	• Extracted Features: Inter Component communication Patterns • SVM • Accuracy	• Gives 10% higher Accuracy than Benchmark Technique	• Inspected only ICC patterns.
[2] L. Wenjia et al. [2015]	• Similarity Score in term of API Calls and Risky permissions were used as a feature • SVM • Accuracy	• Focused on permission set as well as on API calls	• Score calculation produce Overhead • sometimes very low accuracy
[4] D.Arp et al. [2014]	• Static Analysis • SVM • Detection Rate	• Light weighted method • Explain each relevant property of the detected malware. • Detected up to 94% malware with few false alarm	• Based on top features of malware families • Model quality depended on malware samples.
[5] X.Liu et al. [2014]	• Features- requested Permission, Used Permissions and Permission Pairs • J48 classifier • Detection rate, Accuracy	• Used permission helps to improve detection rate	• Focused Only permission based components • Overhead
[9] I. Santosh et al. [2013]	• Based on Extracted permissions • Bayesian, J48, and random tree • K-cross fold validation • Accuracy	• Bayesian based classifier achieved higher accuracy than others.	• Focused on only permission based features • Compromised detection Rate
[10] S.Y. Yerima et al. [2013]	• Extracted features were API calls, system commands and permissions. • Bayesian classification • Detection rate	• Better detection rate then the signature based antivirus software.	• 15-20 features are not sufficient to decide the malicious behavior of an app. • Compromised accuracy.
[12] D. J. Wu. et al. [2012]	• Extracted features: Intent message passing and API Calls, Requested permissions. • K-means and KNN for classification. • Accuracy	• It is effective, scalable, and efficient, Higher Precision Value • Achieved accuracy up to 97.87%	• Not focused on complete feature set • Low Recall value

[14] B. Sanz et al [2012].	• Permission, Market Features of app • Machine learning techniques • AUC	• Better performance • Bayesian gives 0.93 of AUC	• Focused only permission based • Market features introduce overhead
[18] A.P. Felt et al. [2011]	• Static analysis tool • Stowaway: Detects over privilege	• Identified API calls, Content Provider to determine over privileged apps	• Lack of permission set causes error • It fails in case of reflective calls
[19] W. Enck et al. [2011]	• Deddecompiler to recover source code. • Control flow, data flow, structural and semantic analysis were used • Accuracy	• Uncovered dangerous functionalities and vulnerabilities.	• Studied Apps are biased towards popularity • The tool cannot compute the data and control flow for IPC between Component • Missing data and control flows lead to false negatives
[21] A.D. Schmidt et al. [2009]	• Extract function calls using command readelf • PART classification • FNR	• Reduce false negative ratio	• Lack of semantical information

REFERENCES

[1]K. Xu, Y. Li, and R. Deng, "ICCDetector: ICC-Based Malware Detection on Android," in Proc. of IEEE Transaction on Information Forensics and security, vol. 11, no. 6, June 06, 2016.

[2] L.Wenjia, D. Guqian, "Detecting Malware for Android Platform: An SVM Based approach," in Proc. of IEEE 2nd Int. Conference on Cyber Security and Cloud Computing, 2015.

[3] V.N. Cooper, H. Shahriar, and H.M. Haddad, "A Survey of Android Malware Characteristics and Mitigation Techniques," in Proc. of 11th IEEE Int. Conference on InfoTech: New Generations, USA, pp. 327-332, April 2014.

[4] D. Arp, H. Gascon, M. Spreitzenbarth, M. Hübner and K. Rieck, "DREBIN: Effective and Explainable Detection of Android Malware in Your Pocket," in Proc. of 21st Network and Distributed System Security Symposium (NDSS), California, pp. 1-15, Feb. 2014.

[5] X. Liu and J. Liu, "A Two-Layered Permission-based Android Malware Detection Scheme," in Proc. of 2nd IEEE Int. Conference on Mobile Cloud Computing, Services, and Engg., UK, pp. 142-148, 2014.

[6] J. Landage, and M.P. Wankhade, "Malware and Malware Detection Techniques: A Survey," Int. Journal of Engg. Research & Technology (IJERT), vol. 2, issue 12, pp. 1-8, 2013.

[7] H. L. Thanh, "Analysis of Malware Families on Android Mobiles: Detection Characteristics Recognizable by Ordinary Phone Users and How to Fix it," Journal of Information Security (JIS), vol. 4, issue 4, pp. 213-224, Oct. 2013.

[8] V.B. Mohata, D.M. Dakhane, and R.L. Pardhi, "Mobile Malware Detection Techniques," Int. Journal of Computer Science and Engg. Technology, India, vol. 4, issue 4, pp. 2229-3345, April 2013.

[9] B. Sanz, X.U. Pedrero, I. Santos, C. Laorden, P.G. Bringas, and G. Álvarez, "PUMA: Permission Usage to Detect Malware in Android," Int. Joint Conference, Heidelberg, vol. 189, issue 1, pp 289-298, 2013.

[10] S.Y. Yerima, S. Sezer, and G. McWilliams, "A New Android Malware Detection Approach Using Bayesian Classification," in Proc. of 27th IEEE Int. Conference on Advanced Information Networking and Applications, Ireland, pp. 121-128, March 2013.

[11] K. Mathur and S. Hiranwal, "A Survey on Techniques in Detection and Analyzing Malware Executables," Int. Journal of Advanced Research in Computer Science and Software Engg., India, vol. 3, issue 4, pp. 422-428, 2013.

[12] D.J Wu, C.H. Mao, T.E. Wei, H.M. Lee, and K.P. Wu, "DroidMat: Android Malware Detection through Manifest and API Calls Tracing," in Proc. of 7th Asia Joint Conference on Information Security, Tokyo, pp. 66-69, August 2012.

[13] Y. Zhou and X. Jiang, "Dissecting Android Malware: Characterization an Evolution," IEEE Symposium on Security and Privacy, San Francisco, pp. 95-109, 2012.

[14] B. Sanz, I. Santos, X.U. Pedrero, C. Laorden, and P.G. Bringas, "On the Automatic Categorisation of Android Applications," in Proc. of 9th IEEE Consumer Communications and Networking Conference (CNC), USA, pp. 149-153, 2012.

[15] H. Pieterse and M.S. Olivier, "Android botnets on the rise: Trends and Characteristics," in Proc. of IEEE Int. Conference on Information Security for South Africa (ISSA), Johannesburg, South Africa, pp. 1-5, 2012.

[16] R. Johnson, C. Gagnon, Z. Wang, and A. Stavrou, "Analysis of Android Apps"s Permissions," in Proc. of 6th IEEE Int. Conference of Software Security and Reliability Companion, Maryland, pp. 45-46, 2012.

[17] A.P. Felt, E. Chin, M. Finifter, S. Hanna, and D. Wagner, "A Survey of Mobile Malware in the Wild," in Proc. of 1st ACM Conference of Security and privacy in Smartphone and Mobile devices (SPSM), USA, pp. 3-14, 2011.

[18] A.P. Felt, S. Hanna, E. Chin, D. Song, and D. Wagner, "Android Permissions Demystified," in Proc. of 18th ACM Conference on Computer and Communications Security, USA, pp. 627-638, 2011.

[19] W. Enck, P. McDaniel, D. Octeau, and S. Chaudhuri, "A Study of Android Application Security," in USENIX Security Symposium, USA, vol. 2, issue 2, pp. 2-17, 2011.

[20] V. J. V. Hese, "Dissecting Andro Malware, " in SANS Institute InfoSec Reading Room, UK, pp. 1-38, Sept. 2011.

[21] A.D. Schmidt, R. Bye, and H.G. Schmidt, "Static analysis of executables for collaborative malware detection on android," in Proc. of IEEE Int. Conference Communications, Germany, pp. 1–5, June 2009.

[22] R. Meier, "Professional Android Application Development," Published by Wiley publishing, ISBN: 978-0-470-34471-2, Chapter Hello, Android, pp. 1-17, 2009.
[23]N. Idika and A.P Mathur, "A Survey of Malware Detection Techniques," Department of Computer Science, Purdue University, West Lafayette, IN 47907, pp. 1-48, Feb. 2007.
[24] J. Rabek, R. Khazan, S. Lewandowski, and R. Cunningham, "Detection of injected, dynamically generated, and obfuscated malicious code," In Proc. of the ACM Workshop on Rapid Malcode, USA, pp. 76–82, 2003.
[25] C. Bodnar, Malware Classifications, Online available: https://blog.kaspersky.co.in/a-malware-classification/, Last accessed: Oct. 03, 2015.
[26], Last accessed 20 June, 2016. Kaspersky Lab Report Online available: https://www.kaspersky.com/about/news/virus/2016/The_Volume_of_New_Mobile_Malware_Tripled_in_2015

Authors Profile

Nirmala Yadavreceived the **B.E.** degree in computer science & engineering from Rajasthan University Jaipur, Rajasthan, India and **M.E.** in computer science and engineering from Panjab University, Chandigarh, India. Research interest includes Network Security and Malware Analysis.

Amit Doegarreceived the **B.E.** degree in computer science &engineering from Karnatak University, Dharwar, India and **M.E.** in computer science and engineering from Panjab University, Chandigarh, India. Research interest includes Computer Networks, Image Processing, Virtual Learning, and Open Source Technology.

AditiSharma received the **B.E.** degree in computer engineering from Punjabi University Patiala, Patiala, India and **M.E.** in computer science and engineering from Panjab University, Chandigarh, India. Research interest includes Network Security and malware analysis.