International Journal of Scientific Research in Science, Engineering and Technology (www.ijsrset.com)
© 2020 IJSRSET | Volume 7 | Issue 1 | Print ISSN: 2395-1990 | Online ISSN : 2394-4099
DOI : https://doi.org/10.32628/IJSRSET207141

Malicious Code Variant Detection : A Survey

K V Sreelakshmi*, Dileesh E D
Department of Computer Science Engineering, Government Engineering College Idukki, Idukki, Kerala, India

ABSTRACT

Malicious codes have become one of the major threats to computer systems. The malicious software which is also referred to as malware is designed by the attackers and can change their code as they propagate. The existing defense against malware is highly affected by the diversity and volume of malware variants that are being created rapidly. The variants of malware families exhibit typical behavioral patterns exhibiting their origin and purpose. The behavioral patterns can be exploited statically or dynamically to detect and classify malware into their known families. This paper provides a detailed survey of techniques to detect and classify malware into their respective families.

Keywords : Malware Detection, Static Analysis, Dynamic Analysis.

I. INTRODUCTION

Malware is the software that fulfils the harmful intent of an attacker. These are intended to gain access to network resources, personal computers, to thwart computer operations and to obtain personal information without taking the system owner's permission. Malware is of different types such as Trojan, Virus, Worm, Rootkit, Spyware, Adware, etc.
To date, more than 10 thousand malicious codes have been reported. Attackers use vulnerabilities of web browsers, operating systems or social engineering techniques to make users run the malicious codes on their system to spread malware. Malware authors use different types of obfuscation techniques like register assignment, instruction substitution, dead code insertion, code transposition, etc., to evade detection by existing defences like antivirus, firewalls, and gateways which uses signature-based technique and are unable to detect previously unseen malware.

Users of malware are known by various names with the most popular as black hats, hackers, and crackers. The individuals or organizations who take on the above- mentioned names include an external or internal threat, a foreign government or a spy. Malware is usually inserted in the software at two main phases in its lifecycle. The two phases are known as the pre-release phase and the post-release phase. The pre-release phase is a point in the lifecycle of a software before it is released to the end user. At this point, it is only an internal threat or an insider who can insert the malware into the software. An insider is the person working within the organization of a certain software that is normally scheduled to be released to the end users. On the other hand, individuals or organizations who take up the role of a hacker can only insert the malware during the post-release period of the software. Thus, the post-release is the point at which the software is released to the intended audience. While coming up with a new malware, the hackers normally use one or both of the following methods: obfuscation and behaviour addition or modification that helps in bypassing the detectors of the malware. Obfuscation tries to conceal the actual intentions of the malicious code without exhibiting the actual behaviours of the malware. Behaviour addition or modification, on the other hand, creates a new malware, even though the significance of the malware may remain unchanged. The extensive application of the above techniques by malware coders together with those outlined by different researchers posit that a major component towards development of new malware is the reused code. This lead plays a key role in certain malware detection, or in other cases termed as misuse detection techniques as shall be seen later on.

The malware detection system is a system that is used to determine whether a program has malicious intent or not. The detection system includes two tasks – analysis and detection. A malware detector is used as a tool to defend against malware. The qualities of such detectors are determined by the techniques it uses. Malware detection relies on analysis of the features of the malware. There are basically two types of analysis: static and dynamic analysis. Static analysis analyzes malicious codes without executing it. The static analysis uses different types of patterns for analysis such as syntactic library call, string signature, control flow graph, etc. The malicious executable is decrypted before analyzing its execution logic. Dynamic analysis analyzes the behavior of malicious code during runtime. Different techniques used for dynamic analysis include function call monitoring, information flow tracking, instruction traces, etc.. In the dynamic analysis, the malware is executed in a safe virtual environment for analyzing its behavior.

The goal of this paper is to gain an understanding of malware detection using static and dynamic analysis. In particular, we would like to determine the advantages of using static analysis for detecting malware using image processing techniques.

The rest of this paper is organized as follows. In section II we discussed classification of malware. In section III we discussed the malware analysis techniques and conclude in section IV.

II. MALWARE CLASSIFICATION

Malware comes in different forms and categories. These are categorized based on their propagation method and action performed by them on an infected machine using the designed malware program.

Adware: The malicious software that presents unwanted to the user of a computer. The common sources of adware are free games, peer to peer clients, etc.

Botnet: the group of computers that are connected in a particular fashion to perform some malicious activities. The computer is called a bot. these computers are connected in a network that is controlled by a third party and is used to launch an attack or transmit malicious or spam programs.

Ransomware: Type of malware that locks the data on the victim's computer by using encryption and demands for payment to decrypt the data and returning access to the victim.

Rootkit: It is a set of software that provides unauthorized users access to the computer to launch an attack. Once the rootkit is installed the controller can change the data and system configuration of the host computer.

Spyware: These are the software that spies on a user of a computer. It is designed to capture web browsing and other activities and may also capture sensitive information as banking details, credit card information, etc.

Trojan: the malicious program that masks themselves to appear as legitimate. It can destroy the data and can extract sensitive information like bank details, passwords, etc.

Virus: It is a malicious program that travels from one program to another or one computer to another by inserting their code into other programs.

Worm: these are self-replicating programs that spread from one computer to another by transmitting its copy via network relying on the security failures on target computers to access, steal or delete the data.

III. MALWARE ANALYSIS TECHNIQUES

A. Static Analysis

Static analysis is also called as code analysis. It is the process of analyzing the program by examining it i.e. software code of the malware is observed to gain the knowledge of how malware's functions work. In this technique, reverse engineering is performed by using a disassemble tool, decompile tool, debugger, source code analyzer tools such as IDA Pro and Ollydbg to understand the structure of malware. Before the program is executed, static information is found in the executable including header data and the sequence of bytes is used to determine whether it is malicious. The disassembly technique is one of the techniques of static analysis. With static analysis, the executable file is disassembled using disassemble tools like XXD, Hexdump, NetWide command, to get the assembly language program file. From this file, the opcode is extracted as a feature to statically analyze the application behavior to detect the malware. In some of the malware detection techniques, the code is converted into images and then features are extracted. Static analysis is fast and safe. It can analyze multipath malware and produces less false positive that is has greater accuracy.

Kancherla et al. [1] proposed a visualization-based approach for malware detection in which the program executable are converted to a gray scale image called Byteplot. Low-level features like intensity-based and texture-based features are extracted and Support Vector Machine is used for classification. The limitation of this approach is that very fewer features are extracted for classification hence false positive is high.

Ye et al. [2] proposed a system for malware detection using Object Oriented Association (OOA) mining-based classification. The Program Executable files are converted into the API execution sequence using a PE Parser. These generated API sequences are then grouped into a 32-bit ID which represents the corresponding API functions. API calls are then used as signatures for PE files. Using OOA mining algorithm class association rules are generated. The API calls and the association rules are passed to the malware detection module to perform association-based classification.

Makander et al. [3] proposed a malware image classification method using Support Vector Machine in which multi resolution and wavelet are used to build effective texture feature vector using Gabor Wavelet, GIST and Discrete wavelet Transform and other features.

Makander et al. [4] proposed a method for detecting and classifying malware. The malware is in the form of an image that is normalized. These normalized samples are then passed to discrete wavelet transform with three-level decomposition using db4 wavelet family and effective energy coefficients are extracted from the image and stored in the feature vector for classification. The feature vectors are then passed to the training set and testing set and classification is done using Support Vector Machine.

Zhao et al. [5] proposed a virus detection method where the program executables are converted into assembly language by using a disassembler. These assembly languages are then converted into a control flow graph. The features are extracted from the control flow graph using rapid arithmetic method and then trained and classified using Decision Tree, Bagging and Random forest algorithm. The virus is detected using these trained classifiers.

Nataraj et al. [6] proposed a malware detection method by visualizing and classifying malware using image processing techniques. The malware binaries are converted into a grayscale image. The grayscale image is then passed through a bank of Gabor filter from which several filtered images are obtained, and texture-based features are extracted. The texture features are computed by using the GIST algorithm. The malware is classified using KNN and Euclidean Distance and malware are classified efficiently.

Cui et al. [7] proposed a malware detection method based on deep learning. The dataset is a 25 family of malware samples. To address the data imbalance problem in the dataset bat algorithm is used. These samples are converted into a grayscale image and CNN is used for automatic feature extraction and classification of the samples.

Han et al. [8] proposed a method for visually analyzing malware binary information to quickly identify, detect and classify malware and malware families. The malware binary information generated by static analysis is converted into colored image matrices. Using selective area matching the similarity of malware image matrices are determined and malware is classified to their corresponding malware families.

B. Dynamic Analysis

Dynamic analysis is also called behavioral analysis. It analyses the behavior of the malware by executing it in a simulated environment such as a virtual machine, simulator, emulator, sandbox, etc. during the execution of the file in the virtual environment, its system interaction, behavior and effect on the machine are monitored. Dynamic analysis is not safe and is much time-consuming. It cannot analyze multipath malware. Dynamic analysis can detect unknown and known malware but has a high level of false positive.

Trinius et al. [9] proposed a parametrized method for malware detection in which the malware samples are run in a controlled virtual environment to observe its behavior during runtime. This collected information about the behavior of each malware sample is then represented using two visualization techniques: Treemaps and Thread graphs. Treemaps display the distribution of the individual operations performed by the sample. Thread graphs represents the temporal behavior of individual threads in the sample. By using both these representation a human analyst detects the maliciousness of the sample and classifies malicious behavior.

Tobiyama et al. [10] proposed a malware detection system based on analyzing data traffic. The dataset was generated using Cuckoo Sandbox to run the malware files in a controlled virtual environment. The malware behavior was traced to determine generated and injected processes. The log files are created about the behavior of the malware analyzed during runtime. Features are extracted using RNN, based on log files. Features were extracted and converted into images which were then given to CNN for training the image features. The process of validation was evaluated using the trained model. The limitation of the paper is it used a small dataset and the malware can detect the presence of a virtual environment and can behave differently resulting in high false positive.

Chun et al. [11] proposed a technique in which the API log is collected by running the malware samples in a virtual environment. From the API log, the features are extracted and given for classification. The classifiers used in this paper are Naïve Bayesian, J48 Decision Tree and Support Vector Machine. These classifiers then classify whether the sample is a malware or a benign. The limitation of this paper is it has monitoring overhead, and tracing of API is less accurate.

Anderson et al. [12] presented a malware detection algorithm based on the analysis of graphs constructed from dynamically collected instruction traces. A modified version of Ether malware analysis framework is used to collect data. The method uses 2-grams to condition the transition probabilities of a markov chain (treated as a graph). Machinery of graph kernels is used to construct a similarity matrix between instances in the training set. Kernel matrix is constructed by using two distinct measures of similarity: a Gaussian kernel, which measures local similarity between the graph edges and a spectral kernel which measures global similarity between the graphs. From the kernel matrix, a support vector machine is trained to classify the test data. The performance of multiple kernel learning method used in this work is demonstrated by discriminating different instances of malware and benign software. Limitation of this approach is that the computation complexity is very high, thus limiting its use in real world setting.

Rieck et al. [13] proposed a framework for automatic analysis of malware behavior using machine learning. This framework collected large number of malware samples and monitored their behavior using a sandbox environment. By embedding the observed behavior in a vector space, they apply the learning algorithms. Clustering is used to identify the novel classes of malware with similar behavior. Assigning unknown malware to these discovered classes is done by classification. Based on both, clustering and classification, an incremental approach is used for behavior-based analysis, capable of processing the behavior of thousands of malware binaries on daily basis.

Zolkipli et al. [14] presented an approach for malware behavior analysis. They used HoneyClients and Amun as security tools to collect malwares. Behaviors of these malwares are identified by executing every sample on both CWSandbox and Anubis on virtual machine platform. The results generated by both of these analyzers are customized using human based behavior analysis. Then the malwares are grouped into malware families Worms and Trojans. The limitation of this work is that customization using human analysis is not possible for today's real time traffic which is voluminous and having a variety of threats.

C. Hybrid Analysis

Hybrid analysis gathers the information from static and dynamic analysis. It provides the benefits of both static and dynamic analysis and increases the accuracy of detection. As both the static and dynamic analysis has its advantages and disadvantages. Static analysis is fast, cheap and safer than dynamic analysis. But malware evades it by using obfuscation techniques. Dynamic analysis is reliable, resistant to obfuscation techniques and can detect unknown malware. But however, it is time-intensive and resource-consuming.

Schultz et al. [15] introduced the concept of malware detection using data mining. Three different types of static features were used for malware classification: Portable Executable (PE), strings and byte sequences. In PE approach features like the list of DLL function call, list of DLL used by the binary and the number of system calls used within each DLL are extracted from DLL information inside PE files. Based on the text strings that are encoded in program files, strings are extracted from the executables. The byte sequence uses the sequence of n bytes which are extracted from executable files. A rule induction algorithm called Ripper was applied to find the pattern in the DLL data. Naïve Bayes algorithm is used to find the patterns in the string data and the n-gram of byte sequence is used as input data for the Multinomial Naïve Bayes algorithm. The Naive Bayes algorithm, taking strings as input data, gives the highest classification accuracy of 97.11%. The authors claimed that the rate of detection of malware using data mining method is twice as compared to signature-based method.

Choi et al. [16] has proposed a framework for malware classification using both static and dynamic analysis. The features of malware are defined using an approach called Malware DNA. In this a debugging based behaviour monitor and analyser is used to extract the dynamic features.

Islam et al. [17] to classify the executables into malicious and benign files using both static and dynamic features. The static features used in this work include function length frequency and printable sting information and dynamic features used are API function names and API parameters. The experiment was conducted using 2939 executable files including 541 clean files separately for every feature i.e. function length frequency, printable string information and API function calls and then for integrated method for meta classifiers SVM, IB1, DT and RF. The obtained results showed that all meta-classifiers achieve highest accuracy for integrated features and meta- RF is the best performer for all cases. The authors also compared their integrated method accuracy with those of the existing ones and found that their approach is showing the best results.
Eskandari et al. [18] developed and analyzed a tool that they call HDM Analyser. This tool uses both static analysis and dynamic analysis in the training phase, but performs only static analysis in the testing phase. The goal is to take advantage of the supposedly superior fidelity of dynamic analysis in the training phase, while maintaining the efficiency advantage of static detection in the scoring phase.

Santos et al. [19] proposed a hybrid unknown malware detector called OPEM, which utilizes a set of features obtained from both static and dynamic analysis of malicious code. The static features are obtained by modelling an executable as a sequence of operational codes and dynamic features are obtained by monitoring system calls, operations and raised exceptions. The approach is then validated over two different data sets by considering different learning algorithms for classifiers Decision Tree, K-nearest neighbor, Bayesian network, and Support Vector Machine and it has been found that this hybrid approach enhances the performance of both approaches when run separately.

Anderson et al. [20] proposed a method, in which multiple data sources (the static binary, the disassembled binary file, its control flow graph, a dynamic instruction trace & system call trace, and a file information feature vector) are used. For the binary file, disassembled file, and two dynamic traces, kernels based on the Markov chain graphs are used. For the control flow graph, a graphlet kernel is used and for the file information feature vector, a standard Gaussian kernel is used. Then multiple kernel learning is employed to find a weighted combination of the data sources and support vector machine classifier is used to classify the dataset into malicious and benign. It is tested on a dataset of 780 malware and 776 benign instances giving an accuracy of 98.07%.

IV. CONCLUSION

Malware is posing a severe threat to the internet and computer system. Detection and analysis of malware have become very important to create anti-malware software that can detect all kinds of malware and can detect malware with obfuscation. This survey paper describes different techniques used for malware detection under static, dynamic and hybrid analysis. Static analysis is safe and fast than dynamic analysis as the static analysis does not require malware to be executed. Dynamic analysis can detect malware but has low accuracy, unsafe and slower than static analysis-based malware detection schemes. Hybrid analysis combines both static and dynamic analysis for malware detection. It gives a detailed review of malware detection using data mining, deep learning, image processing, and machine learning. The malware features used in most of the previous research works are opcode, API calls, control flow graph, image features, etc. malware can be efficiently detected and classified using these features and accuracy of the system depends on the number of features taken for classification.

V. REFERENCES

[1] Kancherla K., Mukkamala S., "Image visualization based malware detection". In Proc. 2013 IEEE Symp. Computational Intelligence in Cyber Security, CICS, pp. 40–44, 2013.

[2] Y. Ye, D. Wang, T. Li, and D. Ye, "Imds: intelligent malware detection system," in KDD, P. Berkhin, R. Caruana, and X. Wu, Eds., pp. 1043-1047,ACM 2007.

[3] A. Makandar and A. Patrot, "Detection and retrieval of malware using classification," International Conference on Computing, Communication, Control and Automation (ICCUBEA), 2017.

[4] A. Makandar and A. Patrot, "Malware Class Recognition Using Image Processing Techniques," International Conference on Data Management, Analytics and Innovation (ICDMAI) Zeal Society, Pune, India, Feb 24-26, 2017.

[5] Z. Zhao, "A virus detection scheme based on features of Control Flow Graph." 2nd International Conference on Artificial Intelligence, Management Science and Electronic Commerce (AIMSEC), pages 943-947, 2011.

[6] L. Nataraj, S. Karthikeyan, G. Jacob, and B. Manjunath. "Malware images: visualization and automatic classification". In Proceedings of the 8th international symposium on visualization for cyber security page 4. ACM, 2011.

[7] Cui, Z., Xue, F., Cai, X., Cao, Y., Wang, G., & Chen, J. (2018). "Detection of Malicious Code Variants Based on Deep Learning". IEEE Transactions on Industrial Informatics, 14(7), 3187-3196, 2018.

[8] K. Han, J. H. Lim, and E. G. Im. "Malware analysis method using visualization of binary files". In Proceedings of the 2013 Research in Adaptive and Convergent Systems, pages 317–321. ACM, 2013.

[9] P. Trinius, T. Holz, J. G¨obel, and F. C. Freiling, "Visual analysis of malware behavior using treemaps and thread graphs". In Visualization for Cyber Security, 2009. VizSec 2009. 6th International Workshop on, pages 33-38. IEEE, 2009.

[10] S.Tobiyama, Y. Yamaguchi, H. Shimada, T. Ikuse, and T. Yagi, "Malware detection with deep neuralnetwork using process behavior," in Computer Software and Applications Conference (COMPSAC), 2016 IEEE 40th Annual, vol. 2. IEEE, pp. 577–582, 2016.

[11] C. Fan, H.W. Hsiao, C.H. Chou and Y.F. Tseng, "Malware Detection System Based on API Log Data Mining". IEEE 39th Annual International Computers, Software Applications Conference, 2015.

[12] Anderson, B., Quist, D., Neil, J., Storlie, C., & Lane, T. "Graph- based malware detection using dynamic analysis". Journal in Computer Virology, 7(4), 247–258, 2011.

[13] Rieck, K., Trinius, P., Willems, C. and Holz, T. (2011) Automatic Analysis of Malware Behavior Using Machine Learning. Journal of Computer Security, 19, 639-668.

[14] Zolkipli, M.F. and Jantan, A. (2011) An Approach for Malware Behavior Identification and Classification. Proceeding of 3rd International Conference on Computer Research and Development, Shanghai, 11-13 March 2011, 191-194.

[15] Schultz, M., Eskin, E., Zadok, F. and Stolfo, S. (2001) "Data Mining Methods for Detection of New Malicious Executables". Proceedings of 2001 IEEE Symposium on Security and Privacy, Oakland, 38-49, 2001.

[16] Choi, Y H et al. : "Towards extracting malware features for classification using dynamic and static analysis". Computing and Networking Technology (ICCNT), Gueongju, South Korea, pp.126-129.

[17] Islam, R., Tian, R., Battenb, L. and Versteeg, S. (2013) Classification of Malware Based on Integrated Static and Dynamic Features. Journal of Network and Computer Application, 36, 646-556.

[18] Eskandari, M., Khorshidpour, Z., Hashemi, S.: HDM-Analyser: A hybrid analysis approach based on data mining techniques for malware detection. J. Comput. Virol. Hack. Techn. 9(2), 77–93 (2013).

[19] Santos, I., Devesa, J., Brezo, F., Nieves, J. and Bringas, P.G. (2013) OPEM: A Static-Dynamic Approach for Machine Learning Based Malware Detection. Proceedings of International Conference CISIS'12-ICEUTE'12,189,271-280.

[20] Anderson, B., Storlie, C. and Lane, T. (2012) Improving Malware Classification: Bridging the Static/Dynamic Gap. Proceedings of 5th ACM Workshop on Security and Artificial Intelligence (AISec), 3-14.

Cite this article as :

K V Sreelakshmi, Dileesh E D, "Malicious Code Variant Detection : A Survey", International Journal of Scientific Research in Science, Engineering and Technology (IJSRSET), Online ISSN : 2394-4099, Print ISSN : 2395-1990, Volume 7 Issue 1, pp. 245-251, January-February 2020. Available at doi : https://doi.org/10.32628/IJSRSET207141
Journal URL : http://ijsrset.com/IJSRSET207141