Machine Learning and Malware Classification

Amit Raut
College of Computer and Information Sciences
Northeastern University
Boston MA
raut.am@husky.neu.edu

Pranav Sharma
College of Computer and Information Sciences
Northeastern University
Boston MA
sharma.pran@husky.neu.edu

***Abstract*— The exponential growth of malware and its impact on the computer systems has created need for malware classification to determine how many malwares have unique features, what makes them unique. Anti-malware tools use blacklisting approach to stop malwares but they are not effective enough. In this project, unlike anti-malware tools we are using whitelisting approach to perform malware classification. Whitelisting approach means the classifier will allow only known good behavior and deny unknown behavior by default. The key advantage of using whitelisting approach is that it can prevent computer systems from zero day attacks. Our previous work has classified Windows malware samples using static and dynamic analysis. The results for dynamic analysis were very accurate and can be used in malware classification. The results of static analysis were not very accurate because it is difficult to perform classification without running malware. The results are combination of results from static and dynamic analysis using weighted average.**

Keywords— Malware Classification, Malware Analysis, Windows PE, Static and Dynamic Analysis, python

I. INTRODUCTION

One of the biggest threat to computer industry is malware. With programming languages becoming easier to code, malware writers can do more damage with fewer lines of coding. Hence the research we are doing becomes more important as the sheer number of malware is overwhelming but the number of benign executables is still fewer when compared to the malware.

With the rising number of malware, the size of the database containing signatures of malware also increases. To keep up with this, there is a need to update the malware signatures regularly. The biggest risk associated with this approach is a zero-day malware as this approach becomes useless because the signatures are not known to classifier and hence the file is assumed to be benign even when file is malicious leading to compromise of the system.

Using whitelisting methodology, nature of an executable file could be determined. As the malware are growing rapidly the hackers or people with malicious intent make the malware so close to the actual executable file, it has become almost impossible to manually classify them. The goal is to make a classifier that can know if the file is harmful and avoid possible compromise to the system.

The biggest challenge faced was, sometimes a benign file can be altered to incorporate malicious functionality. For this reason, both static and dynamic analysis should be performed.

II. LITERATURE REVIEW

There have been many approaches to solving the problem of malware classification and researchers around the world are trying to find what makes the malware tick. The secret has yet not been discovered that would help analyst and researchers to classify malware. Different angles have been used to tackle this problem. Much research has been done on malware classification based on the behavior factor analysis, instruction frequency etc. Some of the research papers are as follows:

The preliminary work related to malware classification based on the instruction frequency is done by Kyoung Soo Han, Boojoong Kang and Eul Gyu Im [1]. The short paper written by the authors "Malware classification using instruction frequencies" is an approach to classifying the malware variants. The malware variants tend to evade the antivirus signature by abstracting the malicious behavior in some way. With the instruction frequency based classification, malware variants can be identified and classified into malware families.

The research was done by Jiyong Jang, David Brumley and Shobha Venkataraman in their paper "BitShred: Feature Hashing Malware for Scalable Triage and Semantic Analysis" is very important as the growth of malware is exponential [2]. In their paper authors presented BitShred, a system for large-scale malware similarity analysis and clustering, and for automatically uncovering semantic inter and intra-family relationship within clusters. The key idea behind BitShred is the use of feature hashing which dramatically speeds up the malware triage tasks. The clustering can be done in parallel for improved performance.

Another preliminary research is done by Hengli Zhao, Ming Xu, Ning Zheng in their paper "Malicious executables classification based on behavioral factor analysis" [3]. This research focuses on rapid and automated detection and classification of malicious software based on the behavioral analysis. A trace report is generated by characterizing malware behavior profiles after behavioral analysis. The trace report contains the status change caused by the executable and event which are transferred from corresponding Win32 API calls and their certain parameters. The behavior unit strings feature and distinguishes different malware families' behavior patterns. This

feature vector space is fed to Support Vector Machine (SVM) for classification. This method of classification classifies malware into different malware families with higher accuracy and efficiency.

In our research project, we are taking advantage of the fast classification technique to create a feature set out of benign application binaries. The approach used to analyze and classify binary executables is very different than previous work. The research mainly focuses on identifying unique features out of large set of benign Windows binary executables. The feature extraction will use "BitShred: Feature Hashing" method for efficiency. The overlapping consecutive n-gram instructions will be considered as a unique feature. The set of unique features will define a feature set or whitelist. Once whitelist is generated the classifier will use it and predict whether a binary is benign or malicious. The important work of this project lies in determining the important features out of large set of features. This research will determine whether to use a whitelisting approach in malicious binary analysis. If the whitelisting approach is effective and accurate then we will focus our further research on improving the performance of feature extractor and classifier and determine new feature vectors. Based on our previous work we determined that the use of whitelisting approach can successfully predict malicious behavior of the malwares.

As a lot of research has been with supervised learning with malicious software and many supervised learning techniques have used a large set of malware, but comparably smaller sets of benign classification and regression. Since there is a difference between the way a malicious and benign executables work, the research which we are pursuing is different from the previous work, as all the previous research are based upon the classification of malware using blacklists. What paper intend to use an approach in which malware would be classified using a whitelisting method. A whitelist is a list or register of entities that are acceptable. Whitelisting is the reverse of blacklisting, the practice of identifying entities that are denied, unrecognized, or ostracized. Finally, paper outlines cost-benefits analysis to see the extent of success of the whitelisting approach.

III. Problem Statement

The approach for the discovery of malware stems from the blacklisting approach where a known malware is studied and the actions that affect the system/network are added to a blacklist which is used when classifying an executable. The approach this project has is whitelisting approach where instead of studying malware to train the classifier benign files were used. This whitelist is used to classify if an unknown file is a malware or not.

This is different from traditional malware classifiers used in the present-day scenario. As the number of malware is very large when compared with a benign executable used by a person/server. A single benign file can be changed in any number of ways to do various other things that the original author of the application never intended to do. Hence, by containing only the signature of the right file, or the right method to use a file this approach would save a lot of space in terms of storage of the list and help protect from unknown attacks.

IV. Methods and Procedure

The biggest problem faced today in cyber security domain is to differentiate benign executables from malicious ones. This project aims to implement a whitelisting approach which would determine with a fair amount of certainty if the file is malicious or benign.

Empirical Research – The underlying research for the project is based on BitShred: Feature Hashing Malware for Scalable Triage and Semantic Analysis. The analysis reveals that the feature hashing increases the speed when doing comparison when dealing with large scale analysis. The key idea in this was to reduce high-dimensional feature and space in malware analysis. The approach makes clustering and finding nearest neighbor up to 2,365 time faster. And their research extracted features provide insight into the fundamental differences and similarities between and within malware data sets.

This system uses the afore mentioned research to calculate the features of the executables both known benign and unknown samples. These features were then hashed to speed the process of comparison. The original research was scoped around malwares but this project uses it for executables in general.

Constructive Research – "Malware classification using instruction frequencies" [2] research uses instruction frequency as an effective method for fast classification and even malware detection with low computational overheads. False positives were a major concern by the authors of this research paper and to overcome that, the authors suggested combining other malware analysis or detection methods with this approach. And this is one of the main approaches we have used in this project by making use of both static and dynamic analysis.

Procedures Overview – The approach taken include performing static and dynamic on benign executables and create a training set. The method used is shown in following diagrams-

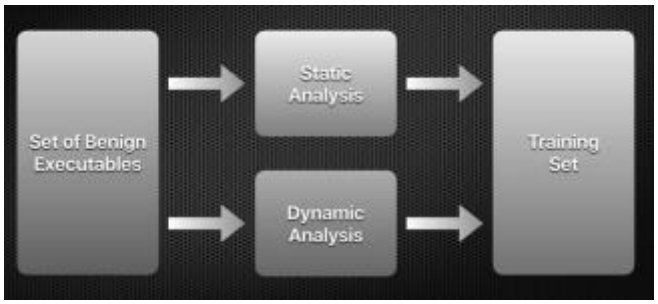

Figure 1: Approach to Build Training Set

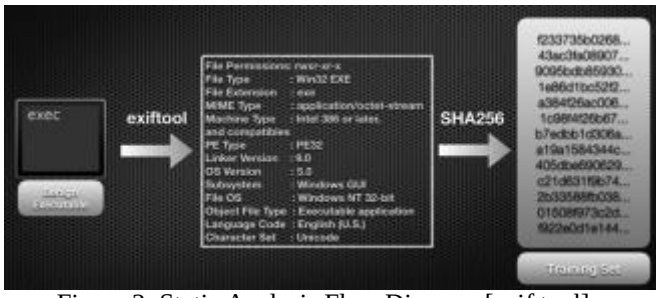

Figure 2: Static Analysis Flow Diagram [exif tool]

Figure 2.1: Static Analysis Flow Diagram [readpe tool]

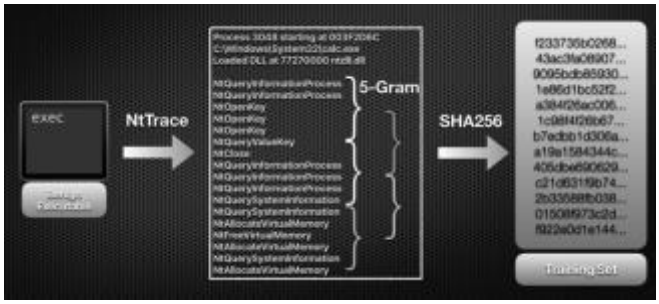

Figure 3: Dynamic Analysis Flow Diagram

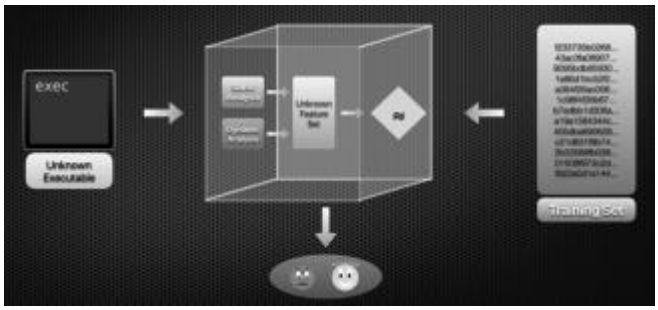

Figure 4: Classifier Flow Diagram

Classifier is used to find the nature of the file in question. This file is provided as an input to the classifier which then compare the SHA-256 checksum. For a Windows, executable SHA256 checksum along with output from each line from ExifTool is used to compare against the data in training set for static analysis to find similarity to being file. Other static analysis tools like readpe from PEV open-source framework for Windows PE static analysis is used. Collectively 49 unique static analysis features from different tools were used towards classification.

For dynamic analysis, the training set was built by running the benign executables under NtTrace tool. NtTrace listed out all system calls for the executable. This research used consecutive overlapping n-gram system calls from the NtTrace output and considered it as one unique feature. The training set for dynamic analysis was built by extracting n-gram features out of many benign samples. To improve accuracy 3-gram, 4-gram, 5-gram, 6-gram, and 7-gram features are extracted, hashed, and stored in the training set.

The classifier will extract the features of the unknown file and calculate hashes for the feature and compare them with feature hashes obtained earlier from the samples of benign executable files i.e. features from training set. The classifier would then find out the similarity between the features of an unknown file and the features of known files i.e. features from training set.

V. Result

Testing indicated that it is important to use both static and dynamic analysis while classifying malware. For Windows by using 3100 benign samples to train we classified 25,000 malwares by doing dynamic analysis. Both static and dynamic analysis were done separately. The dynamic analysis was not successful in some cases because some malwares were making use of the same system calls to abuse the system as benign files. Here training with a larger number of benign samples would have helped to narrow down the false positives. The number of false positives was low by using just dynamic approach, hence it was found that by using static analysis along with dynamic one could reduce the number of false positives.

Static analysis was done by training the classifier with 3000 benign samples and classifying 25000 samples. Static analysis made use of ExifTool which is a perl based program and readpe from PEV framework [10], to read 49 tags and then find the SHA256 signature. These signatures are then stored into a training set. It is found that the approach used was not very accurate in classifying the malware as the parameters read by the ExifTool and readpe can be modified by ease. This points to the fact that for static analysis to be successful multiple methodologies to analyze the unknown file should be used

For future scope, by just static analysis, it could be determined if the file was already present in the repository of benign files. One can then use a detailed matrix to allocate scores to the unknown file based on the static parameter read and the how deterministic the parameter is in specifying if the unknown file is a malware. Dynamic analysis is then performed and the score obtained could then be added to give a final score to the executable.

If one can spot anomalies in the executable with preliminary analysis, it will just give a sense that malwares is not very sophisticated. If the malware was designed by a skilled programmer, they can forge flags and parameters to avoid detection through static analysis. Hence using a multi-faceted static analysis would help to identify a malware by covering

more ground. Based on both dynamic and static analysis one can build a matrix and assign scores to each of these parameters which can then be used to find out the nature of the malware. The score could point out the percentage of maliciousness of the file.

Use of dynamic analysis approach alone was time consuming and generated false positives. To avoid false positives, considering use of static approach was essential. For this, testing was done on the same malware samples set. By using the static features from benign exactable the unknown executables were analyzed statically and their static features were compared with the entries in stored in the training set.

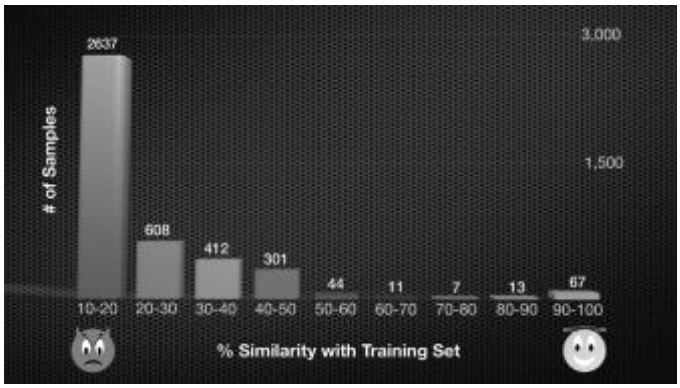

Figure 5: Results

The graph above shows result of the research. It is evident from the graph that the malware samples are categorized as malware and it has very less similarity with the features present in the training set. The results are weighted average of the results obtained from static and dynamic analysis. It is important to note that this research gave higher weightage to the results obtained from dynamic analysis than that of the static analysis. Static analysis sometimes give false sense of accuracy because the executable never ran on the system [4].

VI. Discussion

Considering the results from the dynamic analysis, it is evident that whitelisting approach for malware classification is efficient. It is very safe to execute the benign executables on the system for malware classification than to run actual malwares. Today malware is created in large magnitude and with different evasion techniques. Determining which important features to be used for malware classification is crucial to get the best results. Testing the classification with different feature set can be helpful in determining perfect features. As static analysis involves extracting metadata of the executable and using values in metadata as unique feature, it is observed that the unique features do not have enough entropy. Thus, using static analysis with more number of features is not sufficient to get accurate results. Combinational approach i.e. using static and dynamic analysis should be used in practice. The further steps in the project include moving the classifier to a virtualized environment, using other techniques for malware classification. Another approach in this project will be to create families of the benign samples who share same features. These families can be used to further analyse the malwares based on their behaviour.

VII. Conclusion

Using whitelisting approach for malware classification can be very effective in systems that need very high degree of security. The advantages of using whitelisting approach include assurance of high degree of system security and prevention of zero day attacks. The disadvantages of using this approach include constant updating of the training set and functional inflexibility of using benign application which do not have features like the features in training set.

Malware classification is difficult problem. Static analysis should be preferred over dynamic analysis when dealing with large volume of malware samples but since static analysis could be bypassed by a skilled hacker a balance between dynamic and static analysis should be maintained. At present no perfect approach is present for malware classification. Using combinational approach i.e. blacklisting and whitelisting approach should be preferred for effective malware classification. Sophisticated malwares that use encryption are very difficult to classify and such kind of malware should be tackled using various unencrypting approaches available like the one mentioned in "A Fast Flowgraph Based Classification System for Packed and Polymorphic Malware on the Endhost" [9]. This paper talks about how control flow has been proposed as an alternative signature that can be identified across such variants as signature and string matching have been proven ineffective against polymorphic malware.

VIII. References

[1] Han Kyoung Soo, Kang Boojoong, Im Eul Gyu, Malware classification using instruction frequencies, Proceedings of the 2011 ACM Symposium on Research in Applied Computation, November 02-05, 2011, Miami, Florida

[2] Jang Jiyong, Brumley David, Venkataraman Shobha, BitShred: feature hashing malware for scalable triage and semantic analysis, Proceedings of the 18th ACM conference on Computer and communications security, October 17-21, 2011, Chicago, Illinois, USA

[3] Kinable, J., Kostakis, O.: Malware classification based on call graph clustering. Journal in Computer Virology, 33–45 (2011)

[4] Kruegel Moser, C., and Kirda E. Limits of static analysis for malware detection. In Proceedings of Anual Computer Security Application Conference (ACSAC), Miami Beach, FL, USA, 2007a. ACM Press.

[5] M. Christodorescu and S. Jha. Static analysis of executables to detect malicious patterns. In Proceedings of the 12th USENIX Security Symposium (Security'03), pages 169-186. USENIX Association, USENIX Association, Aug. 2003.

[6] Rieck K., Holz T., Willems C., D¨ussel P., and Laskov P., "Learning and classification of malware behavior," in DIMVA '08: Proceedings of the 5th international conference on Detection of Intrusions and Malware, and Vulnerability Assessment. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 108–125.

[7] Santos, Igor; Laorden, Carlos; Bringas, Pablo G., "Collective classification for unknown malware detection," in Security and Cryptography (SECRYPT), 2011 Proceedings of the International Conference on , vol., no., pp.251-256, 18-21 July 2011

[8] Veeramani, R., Nitin, R.: Windows API based Malware Detection and Framework Analysis. International Journal of Scientific Engineering Research. 3, (2011) [10] Zhao H., Xu M., Zheng N., Yao J., Hou Q., Malicious executables classification based on behavioral factor analysis. e-Education, e-Business, e-Management and eLearning, International Conference on,0:502-506, 2010.

[9] Cesare, S.; Yang Xiang, "A Fast Flowgraph Based Classification System for Packed and Polymorphic Malware on the Endhost," in Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on , vol., no., pp.721-728, 20-23 April 2010

[10] **pev** the PE file analysis toolkit http://pev.sourceforge.net