National Conference on Engineering Innovations and Solutions (NCEIS – 2018)
International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2018 IJSRCSEIT | Volume 4 | Issue 6 | ISSN : 2456-3307

Malware Detection Using Machine Learning

Shruthi M Sullad[1], Shamanth B[1], Prashanth M[1], Rohan Agasthya B R[1], Mrs. Chandini S B[2]
[1]Student, Department of ISE, Vidyavardhaka College of Engineering, Mysuru, Karnataka, India
[2]Assistant Professor, Department of ISE, Vidyavardhaka College of Engineering, Mysuru, Karnataka, India

ABSTRACT

Malware is any software or a computer program that performs malicious actions on a legitimate user's computer system such as information stealing and spying. While the number of malware attacks is rapidly increasing, the major task of cyber security is to protect computer systems from malware attacks which can be done through efficient malware detection. Currently used signature-based methods for malware detection do not provide accurate results in the case of polymorphism or zero-day attacks. Hence, this paper focuses on detecting malware using machine learning techniques. A program can be trained to identify if certain software is malicious or not. By using a Python script, we train a classifier such that it can detect whether Portable Executable (PE) format files are malicious or non-malicious. Five different classification algorithms – Gaussian Naive Bayes, AdaBoost, Gradient boosting, Decision tree, Random Forest classifiers are applied and the best classifier is chosen for prediction by comparing their results in terms of accuracy. The overall best performance is expected to be given by Random Forest classifier with accuracy above 95%.

I. INTRODUCTION

With the growth of the Internet, malware has become one of the major cyber threats nowadays. Malware, or malicious software, is any program or file that disrupts a computer system by performing malicious actions such as information stealing, spying, espionage. Kaspersky Labs define malware as "a type of computer program designed to infect a legitimate user's computer and inflict harm on it in multiple ways." While the diversity of malware is increasing, anti-virus scanners cannot fulfil the demands of security, resulting in millions of PCs being attacked. According to a survey conducted by Kaspersky Labs in the 2016, more than 6 million different hosts were attacked and around 4 million malware objects were identified.

Adding to the increase in malware attacks, and because of the high availability of anti-detection techniques and online attacking tools, the skill level that is required to write a malicious code or build a software that contains malware is decreasing,. Hence, it is very necessary to protect computer systems from malware attacks as even a single attack can result in data leaks and irreversible loss.

The main agenda of this technical paper is understanding the various types of malware, analysis of these malwares, and detection methods.

II. TYPES OF MALWARE

Before dealing with malware analysis and detection, it is important to know the various types of malwares that are at large in the world today. The classes are as follows:

1. Virus - This is the simplest form of software. It is any piece of software that is loaded and launched without user's permission while reproducing itself or infecting (modifying) other software.
2. Worm - This malware type is very similar to the virus. The difference is that worm can spread over the network and replicate to other machines.
3. Trojan - This malware class is used to define the malware types that aim to appear as legitimate software.
4. Adware - The only purpose of this malware type is to display advertisements on the computer. Often adware can be seen as a subclass of spyware and it will very unlikely lead to dramatic results.
5. Spyware - As it implies from the name, the malware that performs espionage can be referred to as spyware. Typical actions of spyware include tracking search history to send personalized advertisements.
6. Rootkit – It's functionality enables the attacker to access the data with higher permission than is allowed. Root kits always hide its existence and quite often are unnoticeable on the system, making the detection and therefore removal incredibly hard.
7. Backdoor - The backdoor is a type of malware that provides an additional secret "entrance" to the system for attackers. By itself, it does not cause any harm but provides attackers with broader attack surface. Because of this, backdoors are never used independently. Usually, they are preceding malware attacks of other types.
8. Keylogger - The idea behind this malware class is to log all the keys pressed by the user, and, therefore, store all data, including passwords, bank card numbers and other sensitive information .
9. Ransomware - This type of malware aims to encrypt all the data on the machine and asks the victim to transfer some money to get the decryption key. Usually, a machine infected by ransomware is "frozen" as the user cannot open any file, and the desktop picture is used to provide information on attacker's demands.

III. MALWARE ANALYSIS AND DETECTION

All malware detection techniques can be divided into signature-based and behaviour-based methods. Before going into these methods, it is essential to understand the basics of two malware analysis approaches: static and dynamic malware analysis. As it implies from the name, static analysis is performed "statically", i.e. without execution of the file. In contrast, dynamic analysis is conducted on the file while it is being executed for example in the virtual machine.

Static analysis often relies on certain tools. Beyond the simple analysis, they can provide information on protection techniques used by malware. The main advantage of static analysis is the ability to discover all possible behavioral scenarios. Static analysis is safer than dynamic, since the file is not executed and it cannot result in bad consequences for the system. On the other hand, static analysis is much more time-consuming. Because of these reasons it is not usually used in real-world dynamic environments, such as anti-virus systems, but is often used for research purposes.

In dynamic analysis, unlike static analysis, the behaviour of the file is monitored while it is being executed and the properties and intentions of the file are inferred from that information. Usually, the file is run in the virtual environment, for example in the sandbox. During this kind of analysis, it is possible to find all behavioral attributes, such as opened files, created mutexes, etc. Moreover, it is much faster than static analysis. On the other hand, the static analysis only shows the behavioral scenario relevant to the current system properties.

Now, with the knowledge of malware analysis, malware detection methods can be defined. The

signature-based analysis is a static method that relies on predefined signatures. These can be file fingerprints, SHA1 hashes, static strings, file metadata. The scenario of detection, in this case, would be as follows: when a file arrives at the system, it is statically analyzed by the antivirus software. If any of the signatures is matched, an alert is triggered, stating that this file is suspicious. Often this kind of analysis is enough since well known malware samples can be detected based on hash values.

However, attackers started to develop malware in a way that it can change its signature. This malware feature is referred to as polymorphism. Such malware cannot be detected using purely signature-based detection techniques. Moreover, new malware types cannot be detected using signatures, until the signatures are created. Therefore, anti-virus vendors had to come up with another way of detection – behaviour-based also referred to as heuristics-based analysis. In this method, the actual behaviour of malware is observed during its execution, looking for the signs of malicious behaviour: modifying host files, registry keys, establishing suspicious connections. By itself, each of these actions cannot be a reasonable sign of malware, but their combination can raise the level of suspiciousness of the file. There is some threshold level of suspiciousness defined, and any malware exceeding this level raises an alert.

The accuracy level of heuristics-based detection highly depends on the implementation. The best ones utilize the virtual environment, e.g. the sandbox to run the file and monitor its behaviour. Although this method is more time consuming, it is much safer, since the file is checked before actually executing. The main advantage of behaviour-based detection method is that in theory, it can identify not only known malware families but also zero-day attacks and polymorphic viruses. However, in practice, taking into account the high spreading rate of malware, such analysis cannot be considered effective against new or polymorphic malware.

Malware detectors that are based on signatures can perform well on previously-known malware, that was already discovered by some antivirus vendors. However, it is unable to detect polymorphic malware, that has an ability to change its signatures, as well as new malware, for which signatures have not been created yet. In turn, the accuracy of heuristics-based detectors is not always sufficient for adequate detection, resulting in a lot of false-positives and false-negatives.

Need for the new detection methods are dictated by the high spreading rate of polymorphic viruses. One of the solutions to this problem is reliance on the heuristics-based analysis in combination with machine learning methods that offer a higher efficiency during detection.

IV. FEATURE SELECTION

In the domain of machine learning, feature selection, also called as variable selection, attribute selection or variable subset selection, is the process of selecting a subset of relevant features for use in model construction. Feature selection methods are used for 4 purposes:

- ✓ simplification of models to make them easier to interpret by researchers/users,
- ✓ reduced training times,
- ✓ enhanced generalization by reducing overfitting.

The primary concern when using a feature selection technique is that the data may contain many features that are either redundant or unnecessary, and can thus be discarded without facing much loss of information. Redundant or unnecessary features are two distinct notions, since one necessary feature may be redundant in the presence of another necessary is strongly inter-dependant distinct notions, since one necessary feature may be redundant in the presence of another necessary feature with which it is strongly dependant.

The three main categories of feature selection algorithms are wrappers, filters and embedded methods.

In wrapper methods, we try to use a subset of features and train a model using them. Based on the inferences that we draw from the previous model, we decide to add or remove features from your subset. The problem is essentially reduced to a search problem. These methods are usually computationally very expensive.

Filter methods are generally used as a pre-processing step. The selection of features is independent of any machine learning algorithms. Instead, features are selected on the basis of their scores in various statistical tests for their correlation with the outcome variable. The correlation is a subjective term here. Embedded methods combine the qualities of filter and wrapper methods. It's implemented by algorithms that have their own built-in feature selection methods.

1) DebugSize: Denotes the size of the debug-directory table. Usually, Microsoft-related executable files have a debug directory. Hence many clean programs may have a non-zero value for DebugSize.

2) ImageVersion: Denotes the version of the file. It is user definable and not related to the function of the program. Many clean programs have more versions and a larger image-version set. Most malware have an ImageVersion value of 0.

3) IatRVA: Denotes the relative-virtual address of the import-address table. The value of this feature is 4096 for most clean files and 0 or a very large value for virus files. Many malware may not use import functions or might obfuscate their import tables.

4) ExportSize: Denotes the size of the export table. Usually, only DLLs, not executable programs, have export tables. Hence the value of this feature may be non-zero for clean files, which contain many DLLs, and 0 for virus files.

5) ResourceSize: Denotes the size of the resource section. Some virus files may have no resources. Clean files may have larger resources.

6) VirtualSize2: Denotes the size of the second section. Many viruses have only one section and the value of this field is 0 for them.

7) NumberOfSections: Denotes the number of sections. The value of this feature varies in both virus and clean files and it is not clear from inspection how this feature helps separate malware and clean files.

V. MODEL SELECTION BY ALGORITHM COMPARISON

Selection of candidate model to find .optimal classifier among the following algorithms.

Gaussian Naive Bayes - Naive Bayes is the classification machine learning algorithm that relies on the Bayes Theorem. It can be used for both binary and multi-class classification problems. The main point relies on the idea of treating each feature independently. It evaluates the probability of each feature independently, regardless of any correlations, and makes the prediction based on the Bayes theorem.

AdaBoost – AdaBoost, short for adaptive boosting, is a machine learning algorithm that can be used in conjunction with other machine learning algorithms to enhance the performance. The output of the other learning algorithms is combined into a weighted sum that represents the final output of the boosted classifier.

Gradient Boosting – Gradient boosting is a machine learning technique which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. It builds the model in the same way the other boosting methods do.

Decision trees - Decision Trees are a type of Supervised Machine Learning where the data is

continuously split according to a certain parameter. It tries to solve using a tree representation where each internal node corresponds to an attributes and each leaf node corresponds to class label. The goal is find the most accurate result with least number of decisions.

Random Forest – Random forest is a supervised classification algorithm. It is basically a collection decision trees. There is a direct relationship between the number of trees in the forest and the result it can get: larger the number of trees more accurate the result.

VI. CONCLUSION

A large number of malicious attacks have been reported in recent days. Machine learning can be another approach to deal with evolving malicious attacks. In this paper, Random Forest algorithm has been used for feature selection. The above mentioned features have been selected as feature subset. The dataset is fit onto five classification algorithms and the results are compared measuring the accuracy. The overall best performance is expected to be given by Random Forest classifier with accuracy above 95%.

VII. REFERENCES

[1] Karthik Raman, "Selecting Features to Classify Malware", 601 Townsend Street, San Francisco, CA 94103.

[2] Jyoti Landage, Prof. M P Wankhade, "Malware and Malware Detection Techniques: A Survey", International Journal of Engineering Research & Technology (IJERT), ISSN: 2278-0181, Vol. 2 Issue 12, December 2013.

[3] Igor Santos, Carlos Laorden and Pablo G. Bringas, "Collective Classification For Unknown Malware Detection", Computing Deusto Institute of Technology, University of Deusto Avenida de las Universidades 24, 48007, Bilbao, Spain.

[4] Yanhui Guo, Qiaokun Wen, Xiaoxi Lin, "Malware Family Classification Method Based on Static Feature Extraction", 2017 3rd IEEE International Conference on Computer and Communications.

[5] Ivan Firdausi, Charles Lim, Alva Erwin, "Analysis of Machine Learning techniques used in behavior-based Malware Detection", 2010 Second International Conference on Advances in Computing, Control, and Telecommunication Technologies.