International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Zero Day Attack by using Machine Learning Algorithm

Sadhana Rajendra Patil.

Department of Computer Science Engineering, SSVPS Bapusaheb Shivajirao Deore College of Engineering Dhule

ABSTRACT

It is difficult to identify malware with a new signature with current antivirus software, which is efficient against existing infections. Detection relying on signatures is ineffective in the face of zero-day attacks. Malware can infiltrate a system until an anti-malware database has been updated with a new signature for that infection. There are many different types of malware, each of which has a distinct purpose in mind when it comes to a computer's security. Software that isn't malicious and isn't harmful to the user is called legitimate. Antimalware software may be improved by using machine learning techniques to identify previously undiscovered malware, zero-day attacks, and other threats. In this paper, we utilized a feature selection algorithm with machine learning techniques to identify the malware classification for the zero-day attack. Machine learning classifiers like Logistic regression, Decision Tree and Random Forest with feature extraction mechanism had been trained and tested on the dataset. We had found that the Decision Tree with feature selection and 10-fold cross-validation had given the best result of 99.99% accuracy.

Keywords: ***Zero-Day Malware, Machine Learning, Feature selection, cross-fold validation***

1. INTRODUCTION

Malware is described as software that is intended to enter or harm a computer system without the owner's explicit permission. Malware is a catch-all term for all kinds of computer dangers. Malware is classified into two types: file infectors & stand-alone malware. Another method of categorizing malware is by its specific action: worms, backdoors, trojans, rootkits, spyware, adware, and so on. Malware detection using standard, signature-based methods is becoming increasingly difficult, as all current malware apps tend to have several polymorphic layers to evade detection or use side processes to automatically update themselves to a newer version at quick intervals to avoid detection through any antivirus software. Emulation in a virtual environment is an example of dynamical file analysis for malware identification. Traditionally, most anti-virus software relies on the "Signature-Based Methodology" that has been around for over a decade. Signature-based malware detection can only identify "known" malware. The properties of every item may be utilized to establish a distinctive signature. Scanning an object yields the digital signature. Once malware is detected by an anti-malware application, its fingerprint is uploaded to a database malware.

There are hundreds of millions of malicious object signatures in these databases. Unfortunately, the rising number of additional versions of malicious code that surface daily makes it difficult to stay on top of things. These newly disclosed kinds of malware may be differentiated from an innocuous file using Supervised Machine Learning Taxonomy. It also helps to overcome some of the disadvantages of signature-based malware detection. Three types of features have been regularly employed in experiments: n-grams over machine code instructions, API call sequence, and PE32 header data. To categorize files as dangerous or clean when utilizing machine learning, a labelled dataset must first be created for training. Specific file characteristics are utilized to determine file features, and each file is then classified either as benign or malicious. A model is produced by examining training records with learning algorithms; this model maps the connection of file attributes and labels.

The classifier/learning method is used to determine if the new file is authentic or malicious.Using the 32-bit Portable Executable PE32 header data, we will be able to identify malicious executable files. The PE32 format is used for 32-bit Windows executable files. Several fields in the executable's header data indicate its metadata (such as how long the executable component of the code is and when it was created, for example). Analysis of header data from benign and malicious PE32 files reported by Yonts [3] in 2012 showed that the header data of harmful and nonmalicious programmes differed on a statistical basis.

Because many header elements are inherent to a program's structure, it is difficult for attackers to manipulate the programme without compromising its functionality. As a result, the attacker will be unable to abuse the detection rules merely by altering the header data. A Machine Learning-based antivirus tool may detect new malware if its structure is comparable to that of known malware.

2. LITERATURE REVIEW

Malware detection has previously been attempted on several occasions. Research publications have employed a variety of methods. The classifier was trained and tested on a collection of 1971 benign and 1651 malicious Windows PE executables, compiled by Maloof and Kolter [1]. To decode each programme, they employed the Hexdump software. N-grams with an average size of 4 were chosen as highlights from the top 500 n-grams. The n-grams were used to create the Waikato Environment for Knowledge Acquisition (WEKA) Java implementation, which included IBk, naive Bayes, a support vector machine (SVM), and a decision tree. In their tests, Boosted J48 detected 98% of the new malware, beating out all other approaches. Additionally, IBk and SVM performed well.

As demonstrated by Schultz et al. [2], data mining may be utilized to determine whether or not a new executable is malicious. The data collection includes 4266 applications, of which 3265 were malicious binaries and 1001 were clean programmes. Each sample of the data set had a binary profile taken from it, and features were derived from these binary profiles. The binary's static attributes were extracted, but the files themselves were not run. Other features collected from executable files included system resource information as well as strings and byte sequences. GNU's Bin–Utils were used to get resource data from executables. There were three distinct sorts of features: The list of DLLs 2. A list of DLL method calls. The number of separate DLL function calls. Researchers employed a 5-fold cross-validation method in their study. An inductive rule-based learner called RIPPER was used to learn the training dataset's rules from the data. The accuracy of RIPPER when DLL functions were employed was 89.36 percent. In addition to the Nave Bayes and Multi-Nave Bayes algorithms, the malware was detected. With a recognition rate of 97.76, Multi-Naive Bayes outperformed signature-based algorithms by more than double.Based on the header data of PE32 files, Yonts [3] investigated the potential and utility of identifying malware in 2012. For the collection, Yonts gathered 2.5 million malicious PE32 files. Malicious indications were discovered using data mining. The detection criteria were able to discriminate between harmful and non-malicious files based solely on characteristics. A property was tested to see whether it could be used to differentiate between benign and malicious behaviour; however, no machine learning classifier was built using the results.

It takes a long time and a lot of effort to collect and keep track of all virus activity. Many studies in the literature have concentrated on gathering only a portion of the mal-dynamic ware's activity. For example, in the case of botnets or ransomware, a malicious C&C site is used, and the traffic created by the virus is analyzed to detect harmful activity. (Lim et al., 2015) A sequence of network flows is shown using a collection of characteristics extracted from the data. To categorize malware traffic, the authors utilized a variety of sequence alignment methods. Real-world malware analysis yielded results of over 60% accuracy, according to the researchers.Kilgallon et al. applied machine learning and dynamic malware analysis (Kilgallon et al., 2017). The proposed technique gathers register value information and API calls made by the monitored malware bi-naris. The collected information is stored in vector structures and analyzed using a value set analysis method. Then, they used a linear similarity metric to compare unseen malware to known malware binaries. Their experiment showed that the proposed technique could detect malware with an accuracy of up to 98.0%.

3. Equations

Equations and formulae should be typed in Mathtype, and numbered consecutively with Arabic numerals in parentheses on the right hand side of the page (if referred to explicitly in the text). They should also be separated from the surrounding text by one space.

$$\rho = \frac{\vec{E}}{J_c\left(T = \text{const.}\right) \cdot \left(P \cdot \left(\frac{\vec{E}}{E_c} \right)^{m} + \left(1 - P\right) \right)} \qquad (1)$$

4. Online license transfer

All authors are required to complete the Procedia exclusive license transfer agreement before the article can be published, which they can do online. This transfer agreement enables Elsevier to protect the copyrighted material for the authors, but does not relinquish the authors' proprietary rights. The copyright transfer covers the exclusive rights to reproduce and distribute the article, including reprints, photographic reproductions, microfilm or any other reproductions of similar nature and translations. Authors are responsible for obtaining from the copyright holder, the permission to reproduce any figures for which copyright exists.

Acknowledgements

Acknowledgements and Reference heading should be left justified, bold, with the first letter capitalized but have no numbers. Text below continues as normal.

An example appendix

Authors including an appendix section should do so before References section. Multiple appendices should all have headings in the style used above. They will automatically be ordered A, B, C etc.

A.1. Example of a sub-heading within an appendix

There is also the option to include a subheading within the Appendix if you wish.

References

Van der Geer, J., Hanraads, J. A. J., & Lupton, R. A. (2000). The art of writing a scientific article.*Journal of Science Communication, 163*, 51–59.

Strunk, W., Jr., & White, E. B. (1979).*The elements of style* (3rd ed.). New York: MacMillan.

Mettam, G. R., & Adams, L. B. (1999).How to prepare an electronic version of your article. In B. S. Jones & R. Z. Smith (Eds.), *Introduction to the electronic age* (pp. 281–304). New York: E-Publishing Inc.

Fachinger, J., den Exter, M., Grambow, B., Holgerson, S., Landesmann, C., Titov, M., et al. (2004).Behavior of spent HTR fuel elements in aquatic phases of repository host rock formations, 2nd International Topical Meeting on High Temperature Reactor Technology. Beijing, China, paper #B08.

Fachinger, J. (2006). Behavior of HTR fuel elements in aquatic phases of repository host rock formations.*Nuclear Engineering & Design*,*236*, 54.