Fourier Transform as Feature Extraction for Malware Classification

Stanislav Ponomarev, Nathan Wallace, Travis Atkison
Louisiana Tech University
Ruston, LA, 71270
{spo013, nsw004, atkison}@latech.edu

***Abstract*—Research efforts to develop malicious application detection algorithms have been a priority ever since the discovery of the first "viruses". Fourier transform is used to extract features from binary files. These features are then reduced by random projection algorithm to create a set of low-dimensional features that are used to classify whether the application is malicious or not. A 99.6% accuracy was reached by Random Forest classifier, while processing various worms, trojan horses, viruses, and backdoors.**

***Index Terms*—Computer security, Virus issues**

I. Introduction

Any Turing-complete machine can run malicious code that is designed to "harm or subvert a system's intended functionality". Applications utilizing such code are known as "malware" [1], [2]. Turing-complete machines include a vast set of devices - from personal computers and cell phones, to machinery automation and utility distribution controllers. According to CTIA, by the end of 2012, there were 326.4 million cell phone subscribers in the US alone [3]. Personal computers, laptops, and tablets expand the list of possible malware carriers. All of these devices now communicate over a network, which means they can get a malicious code without any user interaction. Non-networked devices can also be compromised by such code through user interaction - connecting it to computer, inserting a flash drive that contains infected files, or transferring data in any other means.

There have been many studies of detection and protection against malicious applications initiated by both the industry and university labs. Many classes of malicious applications have been defined. The most common ones include "viruses" - "a program that can 'infect' other programs by modifying them to include a possibly evolved copy of itself" [4], "worms" - "a program that self-propagates across a network exploiting security or policy flaws in widely-used services" [5] , and "Trojan horse" - a term derived from Greek mythology describing a software that masks itself as having useful features for the user [6].

Cohen was the one of the first researchers to study the defense against malicious software [4]. He also recognized the risks of a widespread "infection", which was much harder in the time his publication was written, due to low network connection count of the computers in that time period. Currently, many types of anti-virus software exist. They utilize static and dynamic analysis, neither of which are perfect [7], [8].

Static analysis refers to a set of algorithms that can determine whether a code is benign or malicious by looking at its signatures [7]. These algorithms typically compare the signatures of the scanned files with a database of known malicious code signatures. If the signatures match, the file is marked as malicious. This approach results in two major problems. Only malicious code that has already been "captured" and proven to be malicious will be in the database. Which means "zero day viruses" (viruses that were just discovered) have had a potential to stay hidden on users' machines for years or until the security experts find a copy of such a virus.

Dynamic analysis typically runs an executable inside a virtual environment to determine whether it is malicious or not [8]. However, dynamic analysis can be obfuscated by certain conditional statements such as malicious code execution only on a certain date. Furthermore, it is possible to determine if a code is running inside the virtual environment and it is even possible to write code that can escape virtual environment into the host system [9].

According to the Symantec 2013 Internet Threat Report, there was a malware application in every 1 of 291 emails sent in the year of 2012. One of the primary reasons for such an abundance of malware are "attack kits" - a set of tools that allow almost anyone with some computer knowledge to create or modify new viruses almost instantly. Attack kits allow the class of malware writers to grow past the people with computer penetration knowledge, and include the average users, who might try writing viruses for various reasons from unintentional to active attacks. Combining such tools with simple execution obfuscation techniques allows attackers to create a new strand of a virus by simply morphing the old one. Christodorescu refers to it as a "game between malicious code writers and researchers

working on malicious code detection" [7].

The principles of static analysis and data mining are used in this ongoing research effort to create a set of trained classifiers that are more robust in detecting malicious applications than signature based detection methods commonly used in modern anti-virus applications. By doing so, it is possible not only to detect malicious software that has already been known, but a malicious software that has been obfuscated by various methods.

Atkison was first to suggest the use of random projection in combination with n-gram analysis and data mining algorithms to classify computer applications [10]. Durand then further analyzed different parameters of n-gram analysis and the target feature count of random projection algorithm in combination with several common classifiers. He determined that the 4-gram analysis, using 1500 features as a target feature count, and Support Vector Machines classifier have the highest accuracy of classification. This research evaluates variations of these algorithms to create a code that can be efficiently run on common computers. Fourier transform is also evaluated as a possible replacement for n-gram analysis.

II. Background

The problem of malicious application detection is very popular and well-studied, and has gathered a significant body of research. All research ventures can be categorized as either static analysis or dynamic analysis. Static analysis refers to the process of determining whether an application is malicious without actually running the program in question. Dynamic analysis describes the process of determining whether a program is malicious by monitoring the behavior of a suspect program by executing it, usually within a virtual environment. Neither one of these approaches is a complete solution in itself, but each has a part to play in producing better malware detection systems.

A. n-gram Analysis

When dealing with information retrieval or data mining, the features extracted from the data set play a pivotal role in the success of the prediction process. The information retrieval technique of n-gram analysis has proven to be a valuable tool for feature extraction in several research efforts which focus on the detection and/or classification of malicious applications [2], [11]–[14]. An n-gram is any substring of length n [15]. Since n-grams overlap, they do not just capture statistics about sub-strings of length n, but also implicitly capture frequencies of longer sub-strings [16]. However, due to the high dimensionality of n-gram feature sets, the gathered data is a subject to the "curse of dimensionality" [17]. Many of these research efforts use some form of dimensionality reduction to curb these large feature sets in order to mitigate the effects. For this particular experiment, n-gram of size 4 is used, as it is shown to yield higher classification accuracy as shown in [18].

B. Fourier Transform

As a possible replacement for the n-gram analysis technique, Fourier transform can be used to provide the information about frequency of byte patterns in malicious applications. Unlike n-gram analysis, Fourier transform does not require a built a set of all the n-grams, nor the creation of sparse matrices, which means a large decrease in RAM usage of the software. Fourier transform also simplifies the comparison of the features in files of different length - same frequencies are reported for different lengths of data.

C. Random Projection

Though the feature selection technique of mutual information has been very popular in reducing the feature sets of related research efforts, another dimensionality reduction technique known as random projection has been recently applied to the field of malware detection. Random projection is a feature extraction technique which embeds a high dimensional feature set into a "low-dimensional subspace using a random matrix whose columns have unit length" [19], thus creating a completely new set of features. Random projection feature extraction technique was first introduced to the realm of malicious application detection in [20]. Similarly to Kolter, in [21], a vector space model was used with n-gram analysis to produce weighted feature vectors from binary executables [22]. Every dimension of these vectors represented a unique n-gram which could be extracted from the corresponding executable. Generated feature vectors were then used as input to random projection algorithms in order to produce feature vectors of a reduced dimension. Random projection used Achlioptas' matrix multiplication with a random matrix of values of 0, +1, or -1 following a probability distribution of 2/3, 1/6 and 1/6 respectively to reduce the feature vectors [23]. Previous findings have shown the use of random projection to reduce the feature set to 1500 features to result in higher accuracy [18].

D. Classification algorithms

Nine classification algorithms were chosen for this research: Naïve Bayes, SVM, Simple Logistic, Bagging, Ridor, Decision Stump, J48, LMT, and Random Forest. Previous research results used some of these methods to classify malicious applications [18]. A new set of bagging type classifiers was chosen because they perform well with training sets containing large noise [24].

III. Methodology

Methodology of the previous research effort was used as a control. The dataset of malicious and benign executables was united into corpus. A 4-gram analysis and a random projection reduction to 1500 features was applied to the dataset [18], and the result was used to create data in Attribute-Relation File Format that was processed through Waikato Environment for Knowledge Analysis (WEKA), which used classifiers to determine whether given application is malicious or benign [14]. 10-fold validation was used by the trained classifier to determine average classifier accuracy.

After the control data generated, the algorithm was modified to use Fourier transform instead of 4-gram analysis. The rest of the algorithm remained untouched. The results of the Fourier transform were randomly projected to 1500 features and WEKA was used to train the classifiers. Fourier transform is commonly used in digital signal processing to transform the signal from time domain to frequency domain by following the equation 1. By treating the content of binary executables as a raw waveform, this research was able to transform the data from byte offset to byte pattern frequency, which allowed for an easy comparison of features in files of various sizes.

$$\hat{f}(\sigma) = \int_{-\infty}^{\infty} f(x) e^{-2\pi i x \sigma} dx \tag{1}$$

IV. Experiment

This research effort targets the accuracy aspects of n-gram analysis, random projection, and Fourier transfer methods. The data set described in the next section was first processed by Fourier analysis or n-gram analysis on an xServe G-5 cluster (PPC970FX cpu, 2GB RAM per node), then the extracted features were reduced by random projection and the result was uploaded to a test machine running on Intel Core i7-3770K, 16 GB RAM, 1TB HDD that ran machine learning algorithms to classify the data.

A. Data set

The data set for this experiment consisted of 5124 windows executables in a PE format, with .exe extension.

4270 executables in the data set were malicious applications that break into 854 different Viruses, Backdoors, Trojan Horses, Worms, as well as 854 different instances of Zeus Trojan binary. These executables were obtained from various web-sites online, such as http://www.trojanfrance.com, http://vx.netlux.org, and http://zeustracker.abuse.ch. Previous research efforts used some of the same malicious applications [18].

854 of the executables in this data set were benign. They were obtained by installing an instance of Windows operating systems as well as Office environments in the virtual machine with a disconnected network adapter. The host computer was behind a NAT firewall, as well as the firewall of Louisiana Tech University. As executables were extracted from virtual machines, their MD5 sums were also recorded to make sure they did not get infected during the transfer process. All the executables and their MD5 sums were compressed in an archive, and copied to the research server. Once on the research server, the files were extracted into a dataset folder, given a read only access, and verified versus their MD5 checksums.

V. Results

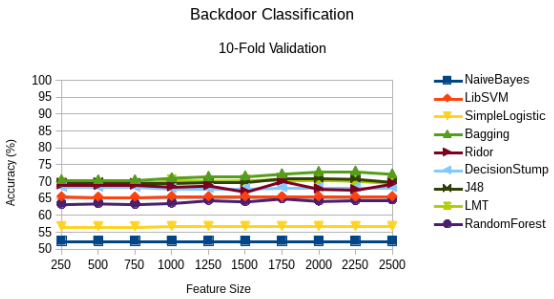

Fig. 1: Backdoors 10-fold classification

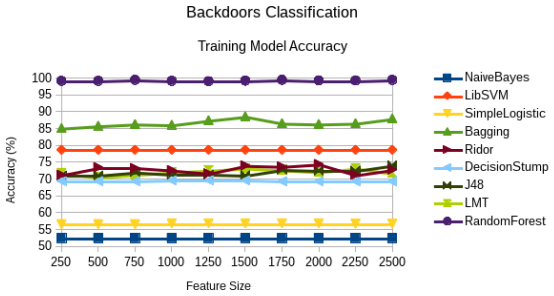

Fig. 2: Backdoors Training Model classification

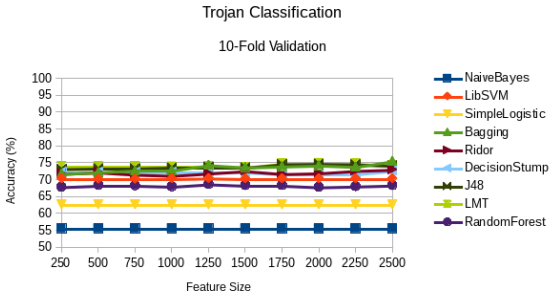

Fig. 3: Trojans 10-fold classification

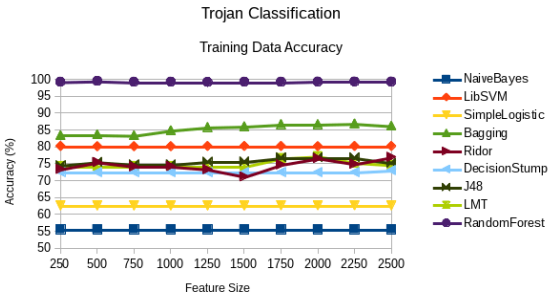

Fig. 4: Trojans Training Model classification

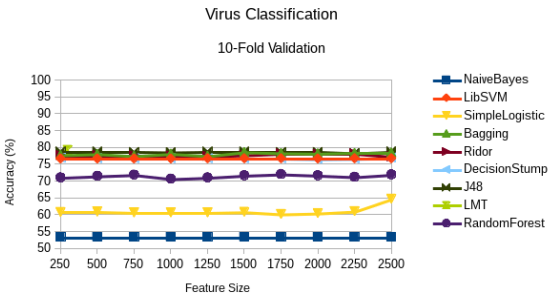

Fig. 5: Viruses 10-fold classification

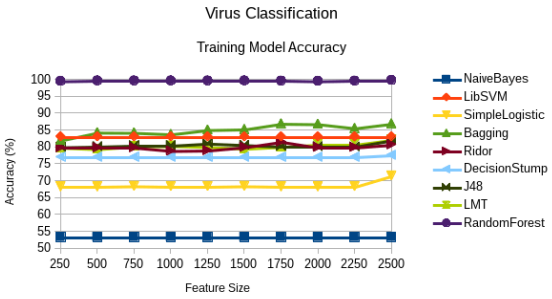

Fig. 6: Viruses Training Model classification

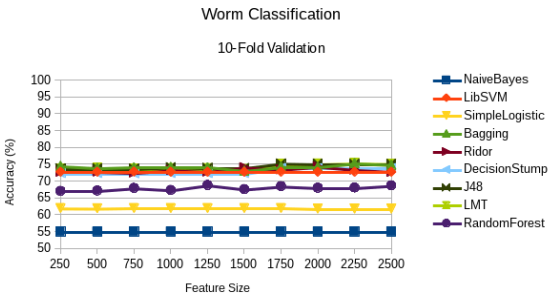

Fig. 7: Worms 10-fold classification

The figures (Fig. 1 - 10) show classifier accuracies based on the amount of features generated by processing

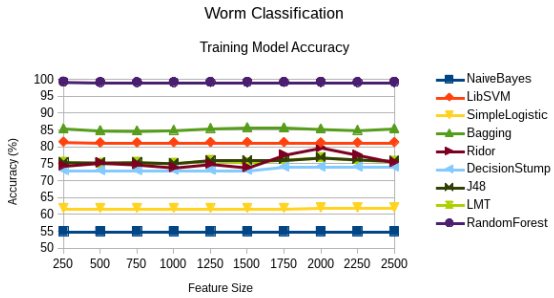

Fig. 8: Worms Training Model classification

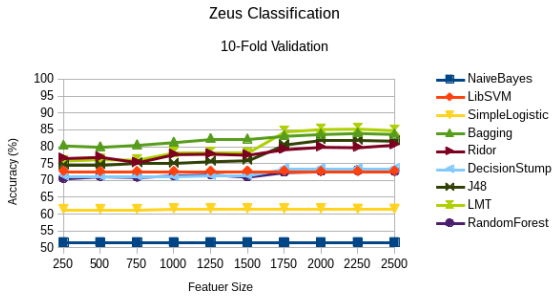

Fig. 9: Zeus Trojan 10-fold classification

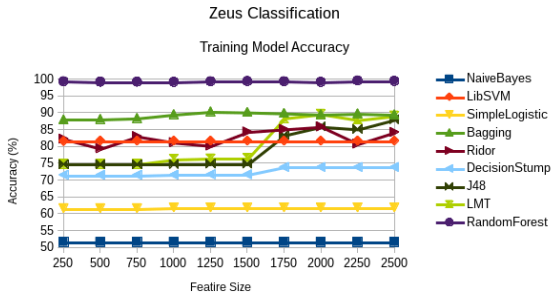

Fig. 10: Zeus Trojan Training Model classification

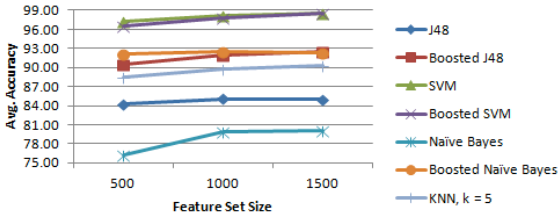

Fig. 11: Classification accuracy using n-gram analysis

the data with Fourier transform, and random projection. Malicious files were separated into worms, trojans, viruses, and backdoors sections. Zeus trojan section was separately created from each individual instance of Zeus Trojan horse. The accuracy of each classifier was

graphed in relation to the amount of features generated by the random projection feature reduction technique.

Random forest classifier was able to acheive more then 99% accuracy with every type of malicious application while evaluating the training model. During cross validation, bagging and LMT classifiers performed the best with a set of Zeus trojan horse malware, reaching 85% accuracy. Bagging also reached the highest accuracy while cross validating backdoor classification, reaching 74%. J48 reached the highest accuracy 80% while classifying viruses, and 75% while classifying trojans and worms.

VI. Conclusions

Random projection has been proven to work well in reducing the amount of features in a dataset for malicious application detection. In conjunction with Fourier transform, these algorithms allow for an accurate classification of malicious applications in various categories, without relying on specific signatures.

An added benefit of using Fourier transform instead of n-gram analysis is much lower memory footprint, and an ability to process new files without restructuring the feature set. N-gram analysis has to use large sparse matrices to generate features, which can take gigabytes to store. Processing files with Fourier transform and random projection for use with machine learning classifiers allows security researchers to detect zero day malicious applications before they have time to damage any critical infrastructure.

VII. References

References

[1] A. Hodges, "Alan turing and the turing machine," in *The Universal Turing Machine A Half-Century Survey*. Springer, 1995, pp. 3–14.

[2] G. McGraw and G. Morrisett, "Attacking malicious code: A report to the infosec research council," *Software, IEEE*, vol. 17, no. 5, pp. 33–41, 2000.

[3] C.-T. W. Association *et al.*, "Wireless quick facts," 2013.

[4] F. Cohen, "Computer viruses: theory and experiments," *Computers & security*, vol. 6, no. 1, pp. 22–35, 1987.

[5] N. Weaver, V. Paxson, S. Staniford, and R. Cunningham, "A taxonomy of computer worms," in *Proceedings of the 2003 ACM workshop on Rapid malcode*. ACM, 2003, pp. 11–18.

[6] C. E. Landwehr, A. R. Bull, J. P. McDermott, and W. S. Choi, "A taxonomy of computer program security flaws," *ACM Computing Surveys (CSUR)*, vol. 26, no. 3, pp. 211–254, 1994.

[7] M. Christodorescu and S. Jha, "Static analysis of executables to detect malicious patterns," DTIC Document, Tech. Rep., 2006.

[8] B. Le Charlier, A. Mounji, M. Swimmer, and V. T. Center, "Dynamic detection and classification of computer viruses using general behaviour patterns," in *International Virus Bulletin Conference*, 1995, pp. 1–22.

[9] P. Ferrie, "Attacks on more virtual machine emulators," *Symantec Technology Exchange*, 2007.

[10] T. Atkison, "Aiding prediction algorithms in detecting high-dimensional malicious applications using a randomized projection technique," in *Proceedings of the 48th Annual Southeast Regional Conference*. ACM, 2010, p. 80.

[11] T. Abou-Assaleh, N. Cercone, V. Keselj, and R. Sweidan, "Detection of new malicious code using n-grams signatures." in *PST*, 2004, pp. 193–196.

[12] O. Henchiri and N. Japkowicz, "A feature selection and evaluation scheme for computer virus detection," in *Data Mining, 2006. ICDM'06. Sixth International Conference on*. IEEE, 2006, pp. 891–895.

[13] I. Santos, Y. K. Penya, J. Devesa, and P. G. Bringas, "N-grams-based file signatures for malware detection." in *ICEIS (2)*, 2009, pp. 317–320.

[14] I. H. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2005.

[15] R. Baeza-Yates, B. Ribeiro-Neto *et al.*, *Modern information retrieval*. ACM press New York, 1999, vol. 463.

[16] T. Abou-Assaleh, N. Cercone, V. Keselj, and R. Sweidan, "N-gram-based detection of new malicious code," in *Computer Software and Applications Conference, 2004. COMPSAC 2004. Proceedings of the 28th Annual International*, vol. 2. IEEE, 2004, pp. 41–42.

[17] R. Bellman, *Adaptive control processes: a guided tour*. Princeton university press Princeton, 1961, vol. 4.

[18] S. Ponomarev, J. Durand, N. Wallace, and T. Atkison, "Evaluation of random projection for malware classification," in *Software Security and Reliability-Companion (SERE-C), 2013 IEEE 7th International Conference on*. IEEE, 2013, pp. 68–73.

[19] N. Goel, G. Bebis, and A. Nefian, "Face recognition experiments with random projection," in *Defense and Security*. International Society for Optics and Photonics, 2005, pp. 426–437.

[20] T. Atkison, "Applying randomized projection to aid prediction algorithms in detecting high-dimensional rogue applications," in *Proceedings of the 47th Annual Southeast Regional Conference*. ACM, 2009, p. 23.

[21] J. Durand and T. Atkison, "Using randomized projection techniques to aid in detecting high-dimensional malicious applications," in *Proceedings of the 49th Annual Southeast Regional Conference*. ACM, 2011, pp. 166–172.

[22] J. Z. Kolter and M. A. Maloof, "Learning to detect and classify malicious executables in the wild," *The Journal of Machine Learning Research*, vol. 7, pp. 2721–2744, 2006.

[23] D. Achlioptas, "Database-friendly random projections," in *Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. ACM, 2001, pp. 274–281.

[24] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An introduction to statistical learning*. Springer, 2013.