DATA PROTECTION AGAINST MALWARE USING MACHINE LEARNING TECHNIQUES

[1] Shoraimov Husanboy Uktamboyevich, [2]Tashpulatova Nadira Botirovna, [3]Akbarova Marguba Xamidovna.

Teacher of the Department, "Systematic and Practical Programming", Tashkent University of InformationTechnologies named after Muhammad Al-Khwarizmi, UZBEKISTAN[1] , Dotsent of the Department, "Systematic and Practical Programming", Tashkent University of Information Technologies named after Muhammad Al-Khwarizmi, UZBEKISTAN[2] , Dotsent of the Department, "Systematic and Practical Programming", Tashkent University of Information Technologies named after Muhammad Al-Khwarizmi, UZBEKISTAN[3]

ANNOTATION

Cyber attacks on confidential data have become a serious threat around the world due to the growing use of computers and information technology. Cyber criminals daily release new malware or viruses over the Internet in an attempt to destroy or steal sensitive data. Consequently, data protection research is of great interest in the Cyber community, to deal with new malware variants, machine learning techniques can be used to accurately classify and detect. This article proposes an efficient scheme for detecting malicious threats using data mining and machine learning methods. Experimental results show that the proposed approach gives better performance compared to other similar methods.

Keywords *– cybersecurity, malware protection, training, data discovery.*

INTRODUCTION

Malicious software or virus programs violate the privacy and integrity of data and cause unauthorized information leakage. In recent years, the number of cyber attacks has been alarming due to the emergence of new applications for computers and the Internet. Hundreds of thousands of new malware are being released by cybercriminals over the Internet in an attempt to steal or destroy sensitive data. Therefore, effective detection and prevention of intruders to protect valuable data is critical in the computer user community.

Typical malware detection approaches can be divided into two categories: (i) signature-based approach and (ii) anomaly-based approach [1]. Signature-based methods are very efficient and quickly detect known malware [2]. Most modern commercial anti-malware methods use a set of signatures to detect malware. A signature can be a sequence of bytes in a file, or a cryptographic cache of a file or its sections. In a signature-based malware detection system, the signature or fingerprint of a file is analyzed and then compared to a signature database of known malicious files. If the signature is not available in the data set, it means that the file is not malicious (malicious) but starts (clean program). However, most malware can generate new variants every time they are executed and a new signature is generated. Thus, signature-based approaches do not detect unknown malware that is not in the database. In addition, another limitation of signature-based detection methods is that they require human knowledge to update the signature database with new signatures [3].

On the other hand, anomaly detection approaches use sequences of API calls instead of matching sequences of bytes [3, 4]. Although anomaly detection approaches use knowledge of normal behaviors and perform better than the signature-based approach, they nevertheless have a high false positive rate. In addition, behavior-based methods are slow, like real-time detectors on the end host, and often require virtual machine technologies.

In recent years, machine learning has been proposed to get around the problems of traditional malware detection methods. Machine learning has been proven to be able to detect new variants of malware [5]. Machine

learning methods used to detect and classify malicious objects include support vector machines, decision tree, random forest, and naive bayes [6]

However, the disadvantages of increasing the false alarm rate are due to poor feature selection and inefficient classifier generation. Thus, accurate malware detection is still a key challenge for the cyber community. This paper proposes a comprehensive framework for detecting and classifying malware in order to protect data from their attacks using a classification algorithm based on machine learning, learning approaches.

The rest of the paper is organised as follows. Section II provides an overview of related work. The proposed malware detection system is described in Section III. The experimental results are presented and discussed in Section IV. Finally, Section V concludes the article.

RELATED WORKS

Many researchers have proposed machine learning approaches for malware detection. Machine learning algorithms used association classifiers, support vector machines, decision tree, random forest, and naive bayes classifier.

In this section, we provide several links to illustrate such methods. Malware classification includes Wang et al [7] a method based on a support vector machine related to static and dynamic analysis of file samples. Static analysis of the samples was performed by dumping the program import table from the Portable structure. Executable (PE) and Records of Used DLLS and API Function Calls o Live analysis ran each sample in a VMWare sandbox environment and tracked changes to the registry, system folders/files, and network states. The results of each type of analysis were used as features in the support vector machine, and information gain was calculated to observe the effect of feature selection.

Chavan and Zende [8] proposed an approach to spyware detection using data mining and supervised machine learning. They extracted n -program features from sample files and used supervised algorithms to classify the file as both spyware and safe. N -grams of various sizes (in the center n = 5) and several supervised learning algorithms were compared to create a more efficient solution than existing antivirus software.

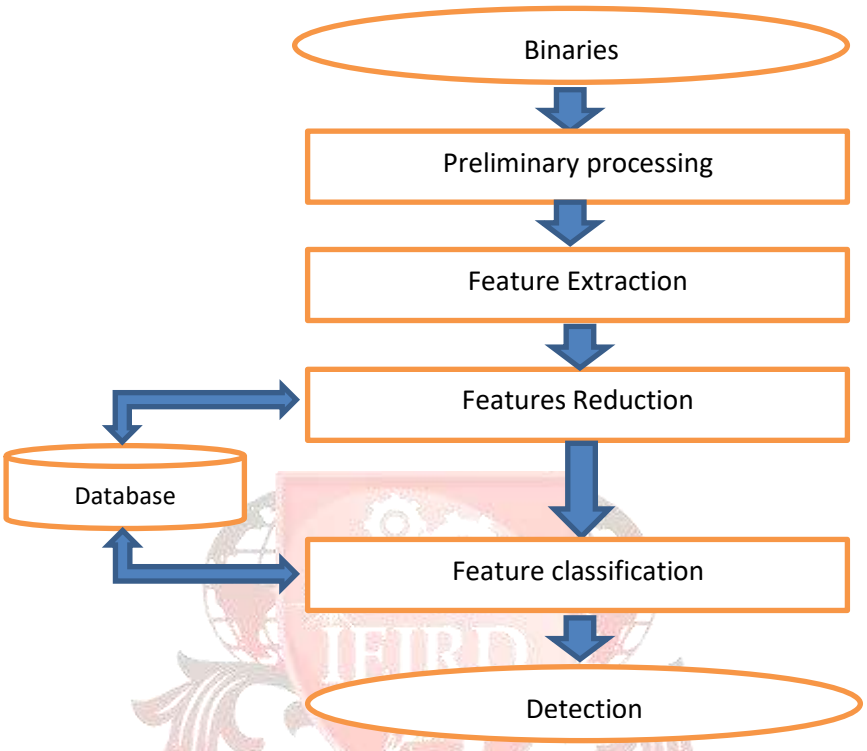
Colter and Maloof [9] proposed a scheme using enhanced n -gram decision trees that gave better results than Naive Bayes classifier and support vector machines. The authors of [10] used association rules for Windows Execution Followers Yu API to distinguish between malware and pure program files.

Chuchane et al. [11] used Hidden Markov Models in their scheme to determine whether or not a given program file is a variant of a previous program file. To achieve a similar goal, Stamp et al. [12] applied Hidden Markov Models, which had previously been used with great success for sequence analysis in bioinformatics.

The ability of neural networks to detect polymorphic malware in [13]. Yoo [14] used self-organizing maps to identify patterns in the behavior of executable viruses in Windows files . Some malware categorization schemes use clustering methods [15]. However, these classification methods require a large number of training samples to build classification models. Moreover, machine learning approaches can have disadvantages associated with increased false positive rates due to poor feature selection and inefficient classifier generation. Hence, accurate malware detection is still a key challenge for the cyber community. This paper proposes a comprehensive framework for detecting and classifying malware to protect data from threats using a machine learning classification method.

PROPOSED APPROACH

The proposed malware detection system consists of the following main components: (1) data collection, (2) pre-processing. (3) Feature extraction, (4) Feature reduction, (5) Classification and (6) Detection. The architecture of the proposed approach is shown in Figure 1.



Picture 1. Architecture of the proposed malware detection system.

A. Data collection.

In the course of this work, we collected a total of 52,185 executable files, including 41,265 of the latest malicious files and the rest of the harmless files. Table 1 shows the malware and cleanware datasets used in this experiment. Malicious files are collected from VX Heaven [6] while safe files are collected from online sources: Download.com and Softpedia.com. The malware collection consists of trojans, backdoors, hacking tools, rootkits, worms, and other types of malware.

File type		Number of files
Malware _	Backdoor	6 280
	Virus	12 340
	rootkit	521
	Worm	9882
	trojan	11 341
	Exploit	289
	Another	679
Clean software		11 289
Total		52 621

Table 1. Malicious and clean software data set

B. Pre-treatment.

The collected files are raw executable files, they are stored in the file system as a binary code, to make them suitable for our work, we have pre-processed them. First, we unpack the executables in a limited environment called a virtual machine (VM). We use the PEid tool [17] to automatically unpack packaged executables.

C. Feature extraction.

After pre-processing the collected files of malware and clean programs, we extract two types of common functions from the file cache: n -grams and portable Windows executables (PE) based on their API calls .

1) Features of N -gram: N -gram is a sequence of substrings of length n -gram. The advantage of using the n-gram method is that it can capture the frequency of words that are n-grams long [18]. We extract n-gram (5-gram) features from every malware and clean programs executable. Through experience, we have found that n-grams of size 5 (each sequence of exactly 5 bytes) give the most accurate results. Hence, we decided to evaluate our experiment with an n-gram size of 5. Term frequency (TF) is used to estimate the frequency of n-gram features that appear in the file. A matrix is created containing a file of malicious and clean programs with vectors of n-grams TF.

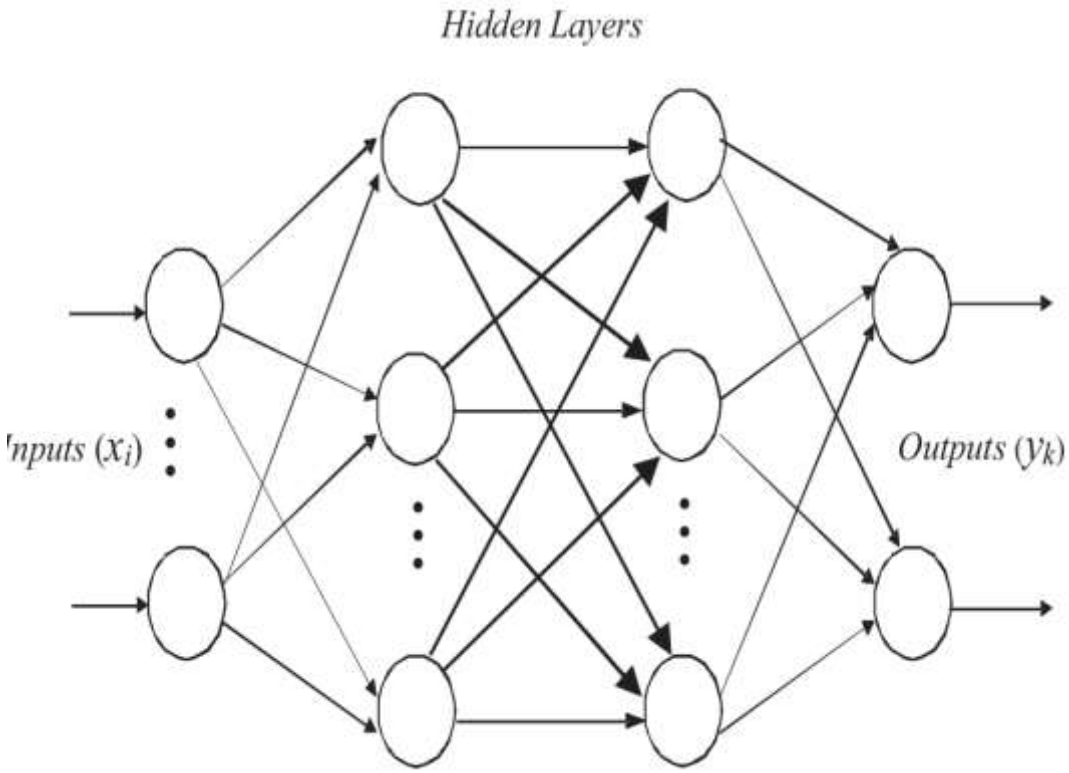
2) N-gram functions: The Portable Executable (PE) header contains information about how the operating system manages the resources allocated to the program. The elements from the PE header are extracted. PE feature mining is done using a package called "pedump" [9].

D. Feature reduction .

After feature extraction, an important step is feature reduction. In this step, the most informative features are selected and the best one is checked based on the accuracy calculation, in this step, the most informative features are selected and the best one is checked based on the classifier accuracy calculation, which corresponds to the number of features selected using different classifier methods, which corresponds to the number of features selected using different feature selection methods. In this work, we use Principal Component Analysis (PCA) [19] for feature selection. PCA is used to increase computational speed due to its ability to reduce dimensionality. It is based on transforming a large number of variables into a smaller number of uncorrelated variables by finding several orthogonal linear combinations of the original variables with the largest variance.

E. Classification and detection.

After processing the binary files, we generate training and test datasets with malicious and clean program files. The classification process is divided into two stages: training and testing. During the training phase, the system is provided with a training set of malicious and safe files. The learning algorithm trains the classifier. The classifier learns from labeled data samples. At the testing stage, a set of new malicious and safe files are loaded into the classifier, which are not used at the training stage. 80% of the malware and cleanware files are used for training, while the remaining files (20%) are used for testing. In this work, we use an artificial neural network (ANN) with a feedforward perceptron to build a classifier. Figure 2 shows the architecture of the proposed ANN classifier. The training set is passed to the classifier to train it and then validate the performance with the test dataset. The number of epochs for this experiment is 35,000 and the minimum error is 0.002.



Picture 2. Neural network architecture for malware classification.

EXPERIMENTAL RESULTS AND DISCUSSION.

In this section, we present the experimental results of our proposed malware classification and detection approach. Experiments are conducted with the collected datasets of both malware and clean programs. To evaluate the effectiveness of the proposed method, we evaluate the following evaluation metrics:

Accuracy = (TP + TN) / (TP + FP + TN + FN) (1)

TPR = TP / (TP + FN) (2)

FPR = FP / (FP + FN) (3)

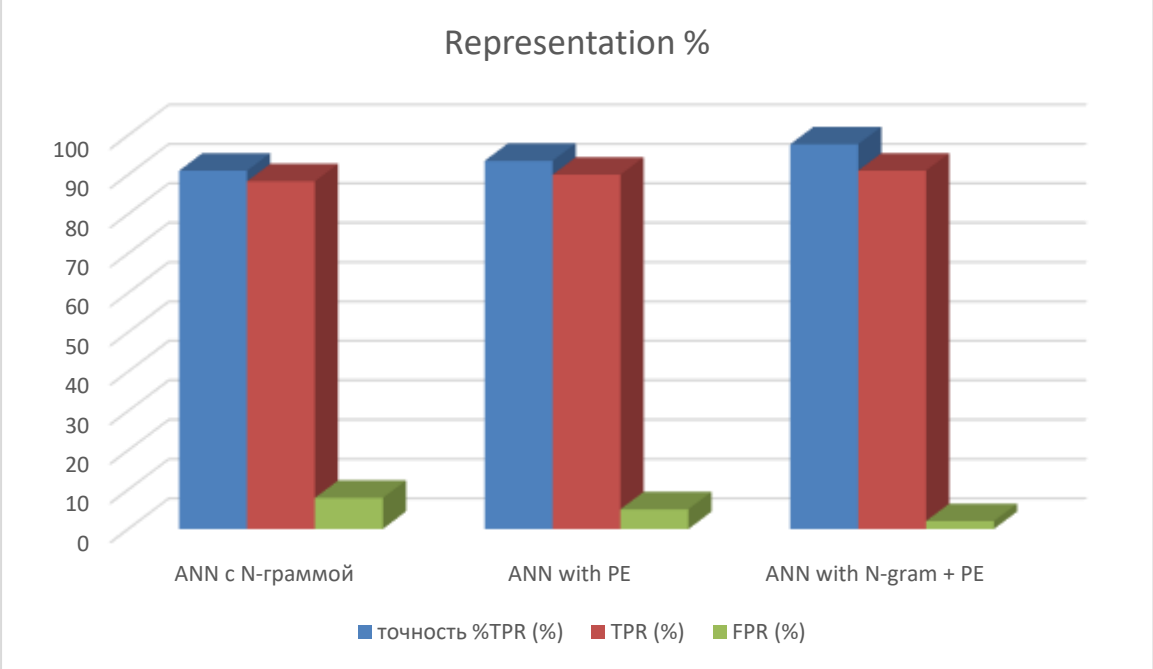
Where, TPR is the true positive rate, also known as sensitivity: and FPR is the false positive rate.

TP (true positive) = number of malware files correctly identified as malware.

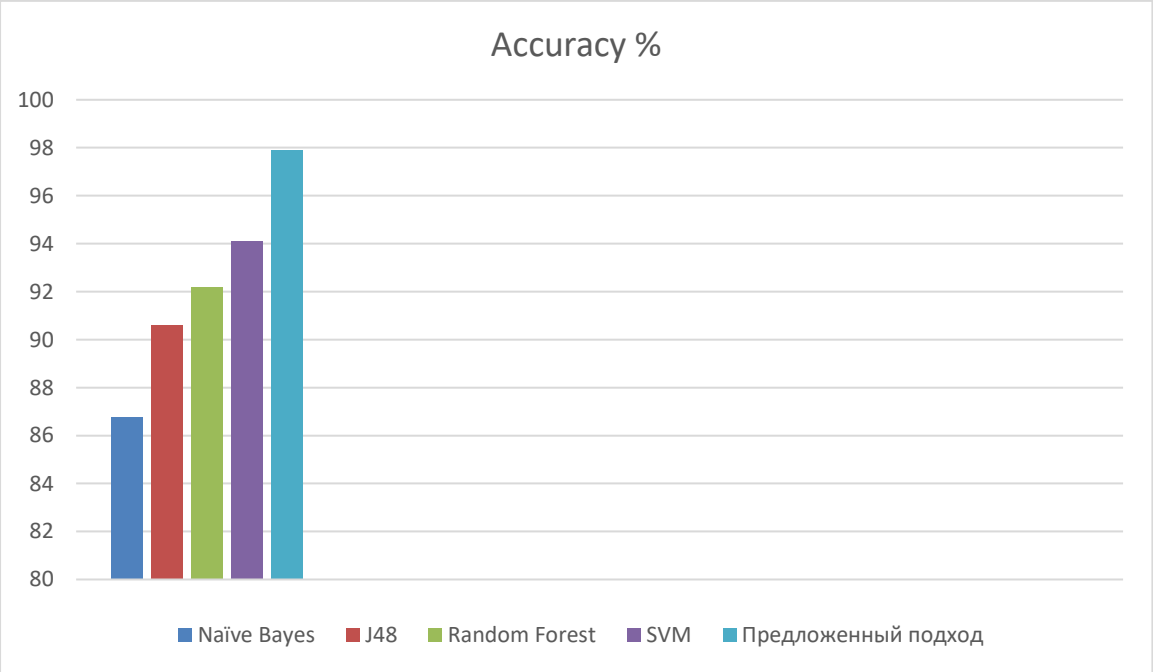
FP (false positive) is the number of malware files misidentified as clean software.

TN (true negative) is the number of clean software files correctly identified as clean software. F N (false negative) - No clean program files incorrectly identified as malware.

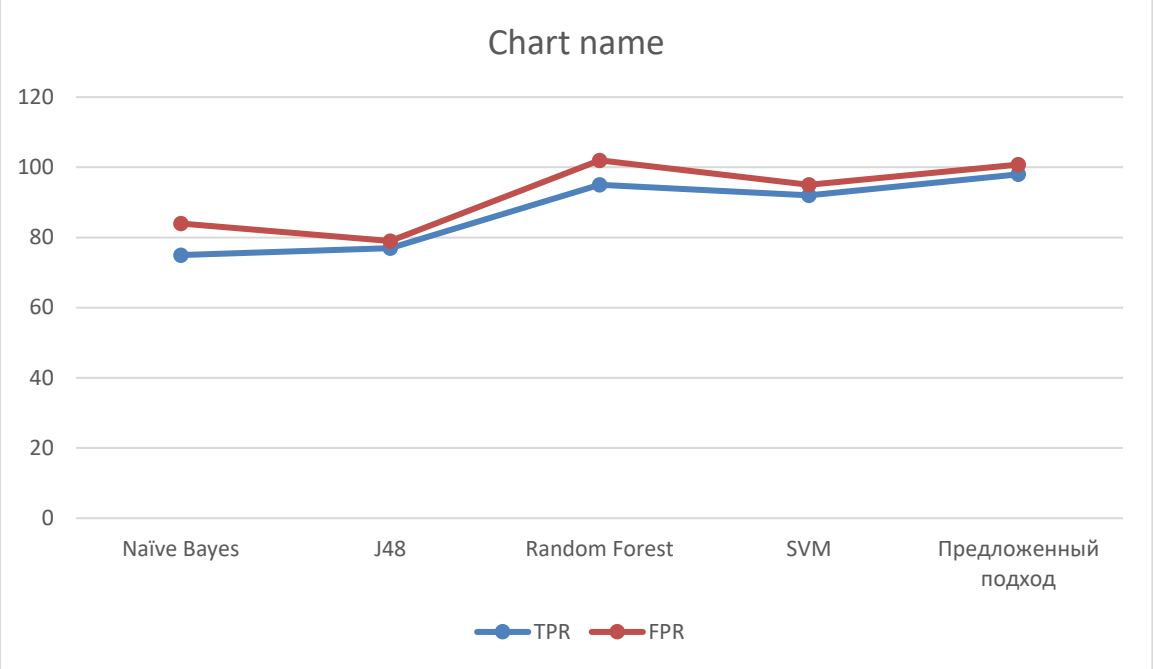
We compare our proposed method with other similar ones. We run experiments using combined functions and single functions to test our approach. The experimental result presented in Figure 3 clearly shows that the proposed classifier gives higher accuracy when integrated features are used. Methods including Support Vector Machine (SVM), Decision Tree (J48), Naive Bayes and Random Forest using similar functions. The experimental results are presented in Figures 4 and 5. The results clearly show that our proposed method provides better performance compared to other similar methods.



Picture 3. Performance of the proposed method with individual features, integrated features.



Picture 4. Comparison of malware detection accuracy by different methods.



Picture 5. Comparison of TPR and FPR for different methods.

CONCLUSION

This article proposes an efficient and reliable malware detection system using a machine learning classification algorithm. We investigated parameter variations and their influence on detection accuracy. Our approach combines the use of n-grams and PE functions to achieve greater accuracy. Experimental evaluation confirms that our proposed method provides better performance compared to other similar methods. In future work, we aim to use more different features in combination with each other in order to achieve greater detection accuracy and reduce the number of false positives.

LITERATURE

[1] Islam R, Tian R, Batten LM, and Versteeg S (2013). Classification of malware based on integrated static and dynamic features. Journal of Network and Computer Applications 36:646–656.

[2] K. Tang, M.T. Zhou, Z. Z-H (2010). An enhanced automated signature generation algorithm for polymorphic malware detection, J. of Electronic Science and Technology of China, 8:114–121.

[3] I . Gurrutxaga , Evaluation of Malware clustering based on its dynamic behavior . Seventh Australasian Data Mining conference, Australia, pp. 163–170, 2008.

[4] Tian R, Islam R, Batten L, Versteeg S. Differentiating malware from cleanware using behavioral analysis. Int. conference on malicious and unwanted software: MALWARE 2010; 2010.p. 23-30.

[5] Hadžiosmanovi _, D., Simionato , L., Bolzoni , D., Zambon , E., and Etalle , S. 2012. N-Gram Against the Machine: On the Feasibility of the N-Gram Network Analysis for Binary protocols. Research in Attacks, Intrusions, and Defenses. Springer. 354-373.

[6] Chan PK and Lippmann R. Machine learning for computer security. Journal of Machine Learning Research, vol. 6, pp. 2669–2672, 2006.

[7] Wang, T., Horng , S., Su, M., Wu, C., Wang, P., & Su, W. (2006). A surveillance spyware detection system based on data mining methods. IEEE Congress on Evolutionary Computation, Sheraton Vancouver Wall Center Hotel, Vancouver, BC, Canada. 3236-3241.

[8] Chavan , M. k., & Zende , D. A. (2013). Spyware solution: Detection of spyware by data mining and machine learning technique. International Conference on Advanced Research in Engineering and Technology, Vijayawada, India

[9] JZ Kolter and MA Maloof, "Learning to detect and classify malicious executables in the wild," Journal of Machine Learning Research, vol. 7, pp. 2721–2744, December 2006, special Issue on Machine Learning in Computer Security.

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

[11] M . R. Chouchane , A. Walenstein , and A. Lakhotia , "Using Markov Chains to filter machine-morphed variants of malicious programs," in Malicious and Unwanted Software, 2008. MALWARE 2008. 3rd International Conference on, 2008, pp. 77–84.

[12] M . Stamp, S. Attaluri , and S. McGhee, "Profile hidden markov models and metamorphic virus detection," Journal in Computer Virology, 2008.

[13] R . Santamarta , "Generic detection and classification of polymorphic malware using neural pattern recognition," 2006.

[14] I. Yoo , "Visualizing Windows executable viruses using selforganizing maps," in VizSEC /DMSEC '04: Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security. New York, NY, USA: ACM, 2004, pp. 82–89.

[15] Kolter JZ, Maloof MA. Learning to detect malicious executables in the wild. In: Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining. New York, NY, USA: ACM; 2004. p. 470–8.

[16] VX Heaven collection, VX Heaven website, available at: http://vx.netlux.org

[17] PEid Unpacker. http://www.peid.info/

[18] Jain, S. and Meena , YK 2011. Byte Level n–Gram Analysis for Malware Detection. Computer Networks and Intelligent Computing. Springer. 51-59.

[19] Xu, X. and Wang, X. 2005. An Adaptive Network Intrusion Detection Method Based on PCA and Support Vector Machines. Advanced Data Mining and Applications. Springer. 696-703.