ADVANCES in NATURAL and APPLIED SCIENCES

ISSN: 1995-0772
EISSN: 1998-1090
2016 April 10(4): pages 534-541

Published BY **AENSI Publication**
http://www.aensiweb.com/ANAS
Open Access Journal

A Mining Approach for Detecting Unknown Malware Using N-Gram and SVM

[1]Muthuvel S, [2]Neelakandan S, [3]Dinesh Kumar M

[1]Associate Professor, Dept. of CSE, Jeppiaar Institute of Technology, India.
[2,3]Assistant.Professor, Dept. of CSE, Jeppiaar Institute of Technology, India.

Received February 2016; Accepted 18 April 2016; Available 25 April 2016

Address For Correspondence:
Muthuvel S, Associate Professor, Dept. of CSE, Jeppiaar Institute of Technology, India
E-mail: smvel21@yahoo.com

ABSTRACT

One of the major and serious threats on the Internet today is malicious software, often referred to as a malware. The malwares being designed by attackers are polymorphic and metamorphic which have the ability to change their code as they propagate. Current Method demonstrates a strategy for static analysis of malware and a way to achieve classification for the same. Signature based malware detection is a popular method but it only identifies variants of malware that have been previously identified. With a signature-based approach, frequent and recurrent updates of the malware signature database are imperative as huge numbers of malware variants are released every day. Therefore, the traditional signature-based detection system is neither efficient nor effective in defeating malware threats. In this project, the indisputable characteristics are used to detect the malware by directly extracting the assembly sequences. By using N-grams, the recurring patterns are evaluated to classify the malware, which is sufficient to detect a zero-day malicious executable. Currently detection of malware using n-grams is not widely implemented. The proposed approach is compared with the dynamic detection approach using dynamic system call sequences. The experimental results show that the proposed approach has higher accuracy and a lower false positive rate than the dynamic detection approach.

KEYWORDS: Dynamic detection, Malware, N-gram, Static detection, SVM.

INTRODUCTION

The rising shadow of Internet economy, malware has developed into one of the major threats to computers and information systems throughout the world. Software like this is being used to compromise computer systems, to destroy the information, and to render them useless. It is also being used to gather information, such as passwords and credit card numbers, and to distribute information, such as pornography, all without the knowledge of a system's users.In recent years, the growth in malware and cyber-crime has been well documented by many sources. According to the computer security company F-secure, the number of samples of malicious code in its database doubled in 2007 alone. It took 20 years to reach this size. Trend Micro, another established computer security company, estimates that the malware industry earns more revenue than US$26 billion.

To Cite This Article: Muthuvel S, Neelakandan S, Dinesh Kumar M., A Mining Approach for Detecting Unknown Malware Using N-Gram and SVM. **Advances in Natural and Applied Sciences**. 10(4); Pages: 534-541

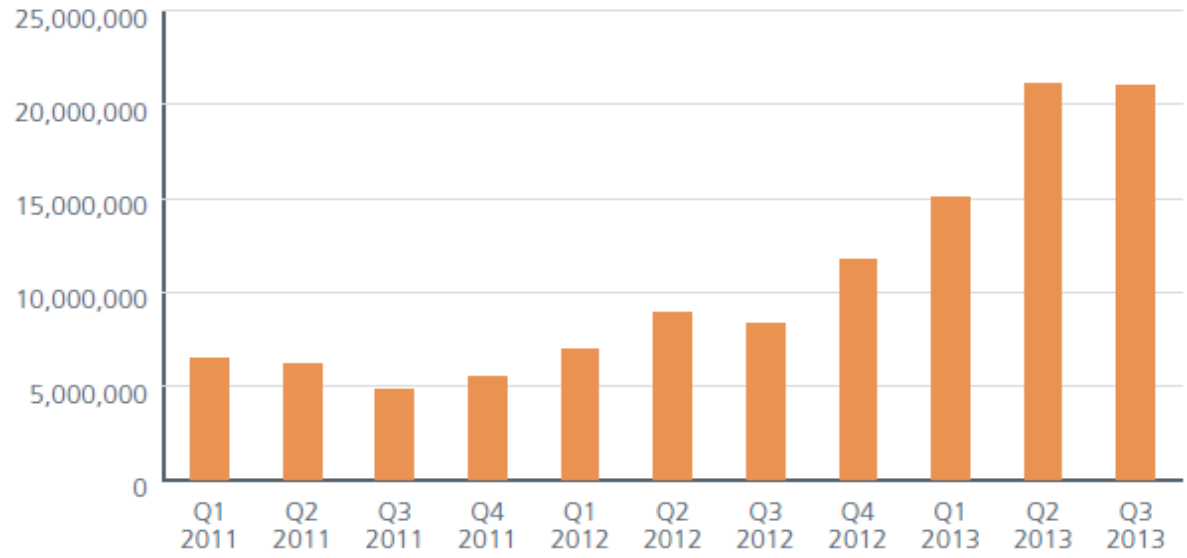

Fig. 1: New malware from 2011 to 2013

Malware and related IT security threats have grown and matured. Malicious code authors are far more adept at camouflaging their work—using the dark parts of the Internet—creating new threats that are smarter, shadier and stealthier. *Malicious code or malware* is "any code added, changed, or removed from a software system to intentionally cause harm or subvert the system's intended function".As a result of malware infection, a victim machine may also unintentionally expose sensitive information, participate in remotely coordinated large-scale attacks, become a spam sender, host phishing websites, etc. As more novice users obtain advanced computers with high-speed connections to the Internet, the potential for further abuse is great. To complicate matters, writing malware programs has become easier. Virus kits are freely available on the Internet. Individuals who write viruses have become more sophisticated, using mechanisms to change or obfuscate their code to produce *polymorphic viruses*. Indeed, researchers have recently discovered that simple obfuscation techniques impede commercial programs for virus detection. These challenges have driven some researchers to consider learning methods for detecting latest or unknown viruses, and more generally, malicious code.

II. Related work:

2.1. Malware classification using N-grams sequential pattern features:

2.1.1. N-Grams Extraction:

An n-gram is an n-character slice of a longer string. To extract n-grams, first IDA-Pro [10], a tool that disassembles files, is used to extract contents of a file into a long string of hexadecimals. The string is then processed into a set of overlapping n-grams. In our study, we explore n-grams of several different lengths. The kfngram tool is employed to generate n-gram slices. In the experiments, our tests are run with n=1, n=2, n=3 and n=4.

2.1.2. Sequential pattern extraction:

There are a very large number of n-grams, but they lack order information necessary to capture characteristics of a program. This information can be provided by sequential patterns. A sequential pattern extraction technique is used to generate frequently occurred sequences of n-grams to represent the data and reduce response time. After being processed, malware is represented by a set of frequent patterns.

2.1.3. Malware family classification:

A classification model accepts a feature vector and returns the family of the malware. Three learning algorithms are studied in this research, which consist of C4.5, multilayer perceptron, and support vector machine [27]. They are available in KNIME [13], a data mining software. The data set is randomly split into two partitions: 80% for training and 20% for testing.

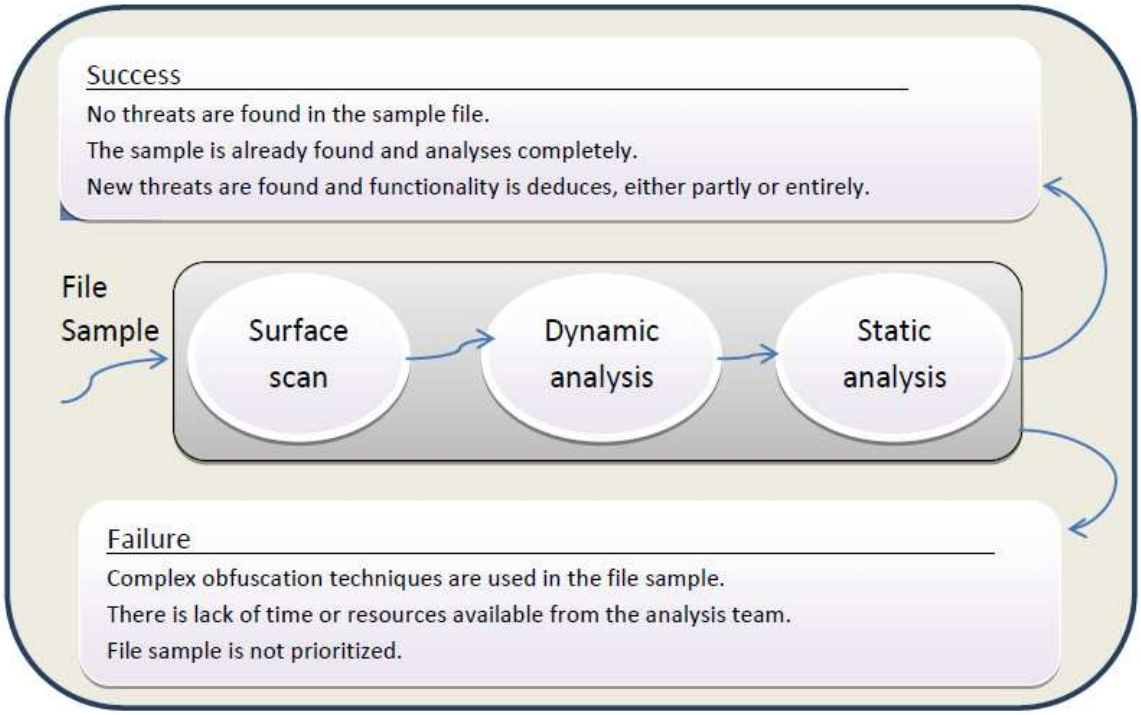

Fig. 2: Graphical representation of the different steps in a malware analysis.

2.2. Malware detection using perceptrons and support vector machines:

A training database was created (henceforth denoted VH), which contains 273133 clean files and 27475 malware files. The malware files were collected from the Virus Heaven website. Since our main target was to create a detection with as few as possible (or even 0) false positives, the VH database has much more clean entries than malware ones. The malware files in this collection are of a variety of types such as: trojans, back doors, hack tools, root kits, worms and other types. The clean files from the training (VH) database are mainly system files (from different versions of operating systems) and executable and library files from different popular applications. We also use clean files that are packed or have the same form or certain geometrical similarities with malware files (e.g. use the same packer) in order to better train and test our machine learning system. From the whole set of feature we created, 308 binary features were selected. Different files may generate equal sets of values for the chosen feature set; therefore they were counted only once. Each one of the left columns represents the percentage of that malware family from the total number of files, while each one of the right columns represents the corresponding percentage from the total number of unique combinations of features.

III. System design:

Correctly classifying malware is an important research issue in a malware detection and classification system. In this proposed system, we use heuristic-based detection method as a malware detection method.

3.1. Heuristic-based detection:

Heuristic-based systems try to detect known and unknown Malware on the basis of rules defined by experts who define behaviour patterns for malicious and benign software. Heuristic malware detection methods use data mining and machine learning techniques to learn the behaviour of an executable file.The following are the features employed in Heuristic Methods:

3.1.1. API/System calls:

Almost all programs use application programming interface(API) calls to send their requests to the Operating System, API call sequences is one of the most attractive way that reflects the behavior of a piece of code like malware.

3.1.2. Control Flow Graph:

Control Flow Graph (CFG) is a graph that represents the control flow of programs and are widely used in the analysis of software and have been studied for many years. CFG is a directed graph, where each node

represents a statement of the program and each edge represents control flow between the statements (i.e. what happens after what).

3.1.3. N-Grams:

N-Grams are all substrings of a larger string with a length of N. For example, the string “VIRUS”, can be segmented into several 3-grams: “VIR”, “IRU”, “RUS” and so on. Over the past decade, several researches have been motivated on the detection of unknown malware based on its binary code content..

3.2. Data mining techniques:

Data mining has been defined as ”the nontrivial extraction of implicit, previously unknown, and potentially useful information from data” and ”the science of extracting useful information from large data sets or databases”. Data mining can be divided into two broad categories:

- Predictive data mining, involves predicting unknown values based upon given data items.
- Descriptive data mining, involves finding patterns describing the data.

Data Mining involves the application of a full suite of statistical and machine-learning algorithms on a set of features derived from malicious and clean programs. The outcome of a data mining process is a model or a collection of models. A number of data mining techniques are available for the model building process. These techniques are not limited to the final stage though. Any previous data mining stage e.g. data pre-processing may use these techniques. The idea of this project is to use variable length instructions sequence, extracted from binary files as the primary classification feature. Unlike fixed length instructions or n-grams, the variable length instructions inherently capture the programs control flow information as each sequence reflects a control flow block.

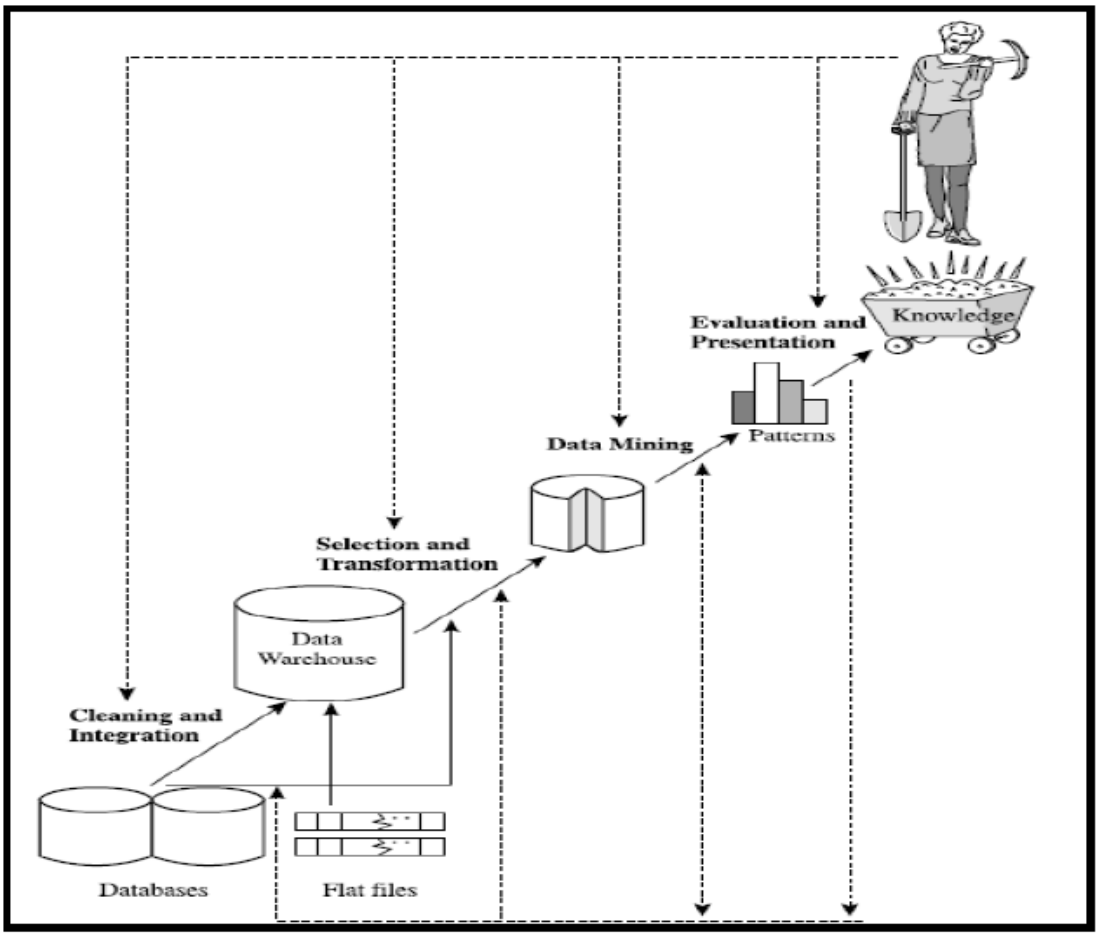

Fig. 3: Data mining as a step in the process of knowledge discovery

3.3. Classification learning algorithms:

The particular problem of labelling a program as malware or clean is an example of the general classification problem, which in turn belongs to a wider class of dependency estimation problems. Learning theory describes one approach to such problems. The implementation details differ, but the main task is to learn

from given data samples by searching through an n-dimensional space to form an acceptable generalization. More formally, given a collection of samples $(x_i, f(x)_i)$, the inductive learning process returns a function h(x) that approximates f(x). The approximating function can be defined in terms of its internal parameters, and a more precise form can be expressed as h(X, w), where X is the input vector and w is the functions parameters. The quality of the approximation can be measured in terms of a function that can compare the output produced by the learning machine to the actual output. Such a function is called the,

loss function L(y, h(X, w)):

where y is the actual output of the system, X is the set of inputs, w is the parameters of the approximation function, h(X, w) is the output produced by the learning machine using X as inputs and w as internal parameters.

The expected value of the loss function is called the *risk functional R(w):*

$$R(w) = \iint L(y, h(X, w))p(X, y)dXdy$$

where L(y, h(X, w)) is the loss function and p(X, y) is the unknown probability distribution of the input X. The *vector space model* was used to create our feature set. In information retrieval, a vector space model defines documents as vectors (or points) in a multidimensional Euclidean space where the axes (dimensions) are represented by terms. Depending upon the type of vector components (coordinates), there are three basic versions of this representation: Boolean, term frequency (TF) and inverse document frequency (IDF). The programs and instruction sequences are mapped to documents and terms, respectively. Using these program vectors, a term-document matrix was created where programs were arranged in rows, while columns represent the potential features (instruction sequences). For feature selection Boolean representation of the matrix where, where a 1 represented presence, while 0 represented absence, of an instruction sequence in a given program. Assume there are n programs p_1, p_2, ... p_n, and m instruction sequences s_1, s_2, ..., s_m. Let $n_{i\,j}$ be the number of times a sequence s_i was found in a program p_j.In order to be selected, a sequence s_i, must have its $N_{i\,j}$ greater than a defined threshold. This threshold was set to 10% of the total number of the programs, as it is a common practice in data mining for defining unary variables.

$$N_{ij} = \frac{n}{10}$$

3.4. Proposed framework:

The process of the design implemented with the system architecture view comprises of the parts of the project work that encapsulates all modules ranging from module to module communication. The system developed here is to classify any PE file as either malware or benign. Data analysis and pre-process is applicable for data mining process and data can be selected based on static analysis.

$$p_j^i = \begin{cases} 0 & if\ n_{ij} = 0 \\ 1 & if\ n_{ij} > 0 \end{cases}$$

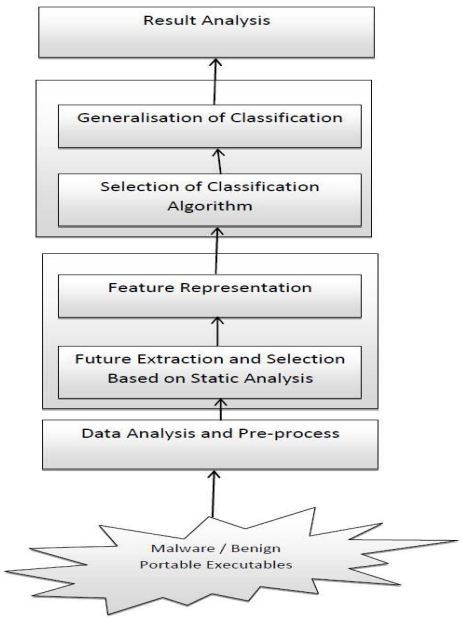

Fig. 4: Framework of proposed system

The product of data pre-processing is the final training set. Analyzing data that has not been carefully screened for such problems can produce misleading results. Thus, the representation and quality of data is first before running an analysis.

3.5. Feature Extraction and Selection: N-Grams:

A number of research studies has shown that better results are gained through n-gram size In the light of previous research, I chose three different n-gram sizes, 4, 5, and 6.

3.5.1. Frequency:

In the frequency calculation of ngrams, there were n-grams like 0x0000000000, 0xFFFFFFFFFF, and 0x0000000001 which have more than 10,000 occurrence frequencies, there is huge jump to more than 10,000 in frequencies after 500, and these n-grams are few sometimes less than 10. After analysis of number of n-grams in each frequency range, it was decided to not perform experiments on 1-49 because of very high number of n-grams in reduced feature set. For example, in n-gram 5, total number of n-grams was 7,865,739. It was a clear indication that it contains a large amount of uninteresting features. Feature reduction is critical element in this system. It is necessary to reduce features of a binary in such a way that interesting features should not be left out from the experiment and uninterested features should as less as possible. The approach was adopted based on feature frequency.

IV. Conclusion and future work:

A statistical analysis of malware reflects the behaviour of a piece of code. In this project, the portable executable files were disassembled and the n-grams were extracted from each file and further refined using frequency measure to classify the executable as malicious or benign. The experimental results for different sizes of n-grams are promising as a benchmark for improvement of the framework with different set of features thereby increasing the accuracy.

V. Results &future enhancement:

In future work, we hope to remove obfuscation from our malicious executables and rerun the experiment on classifying payload function. Removing obfuscation and producing “clean” executables may prove challenging, but doing so would provide the best opportunity to evaluate whether obfuscation affected the performance of the classifiers. The newer process should re train the data mining process by utilizing newer and different benign and malicious executables to further validate positive results. Retraining the model could be a risk as the results may come out negatively. To ensure this model does not overwhelm a system with false positives, benign executables need to be identified and classified.

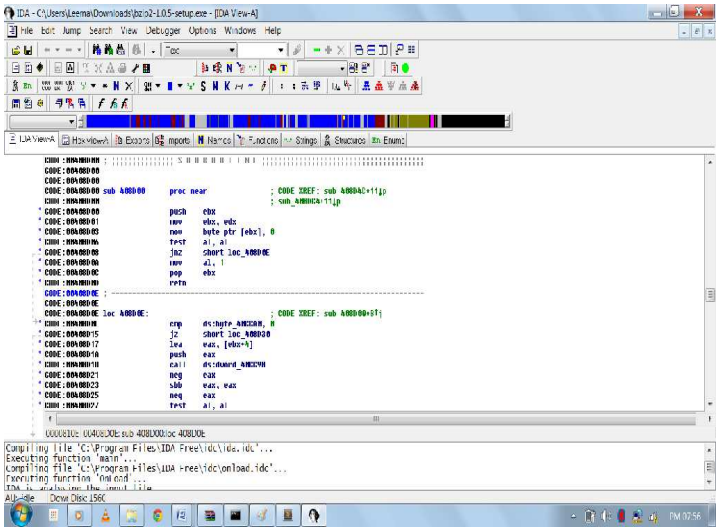

Fig. 5: Using IDA to disassemble bzip2-1.0.5 setup.exe

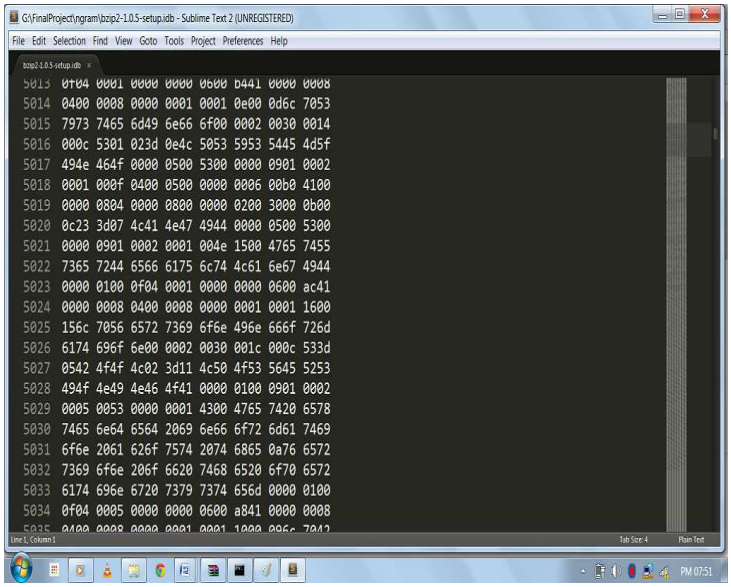

Fig. 6: Disassembled file of bzip2-1.0.5- setup.exe

REFERENCES

1. Cohen, F., 1989. Models Of Practical Defenses Against Computer Viruses.Computers & Security, 8(2): 149-160.
2. Stallings, W., 2006. Cryptography And Network Security, 4/E. Pearson Education India.
3. Bilge, L. and T. Dumitras, 2012. Before We Knew It: An Empirical Study Of Zero-Day Attacks In The Real World. In Proceedings Of The 2012 ACM Conference On Computer And Communications Security, pp: 833-844. ACM.
4. Ahmed, F., H. Hameed, M.Z. Shafiq and M. Farooq, 2009. Using Spatio-Temporal Information In API Calls With Machine Learning Algorithms For Malware Detection. In Proceedings Of The 2nd ACM Workshop On Security And Artificial Intelligence, pp: 55-62). ACM.
5. Witten, I.H. and E. Frank, 2005. Data Mining: Practical Machine Learning Tools And Techniques. Morgan Kaufmann.
6. Schultz, M.G., E. Eskin, E. Zadok and S.J. Stolfo, 2001. Data Mining Methods For Detection Of New Malicious Executables. In Security And Privacy, 2001. S&P 2001. Proceedings. 2001 IEEE Symposium On, pp: 38-49. IEEE.
7. Cohen, W.W., 1995. Fast Effective Rule Induction. In Proceedings Of The Twelfth International Conference On Machine Learning, pp: 115-123.
8. Kolter, J.Z. and M.A. Maloof, 2004. Learning To Detect Malicious Executables In The Wild. In Proceedings Of The Tenth ACM SIGKDD International Conference On Knowledge Discovery And Data Mining, pp: 470-478. ACM.
9. Fawcett, T., 2004. ROC Graphs: Notes And Practical Considerations For Researchers. Machine Learning, 31(1): 1-38.
10. Siddiqui, M., M.C. Wang and J. Lee, 2009. Detecting Internet Worms Using Data Mining Techniques. Journal Of Systemics, Cybernetics And Informatics, 6(6): 48-53.
11. Bilar, D., 2007. Opcodes As Predictor For Malware. International Journal Of Electronic Security And Digital Forensics, 1(2): 156-168.
12. Bilar, D., 2007. Callgraph Properties Of Executables. AI Communications, 20(4): 231-243.
13. Santos, I., Y.K. Penya, J. Devesa and P.G. Bringas, 2009. N-Grams-Based File Signatures For Malware Detection. ICEIS (2)9: 317-320.
14. Boser, B.E., I.M. Guyon and V.N. Vapnik, 1992. A Training Algorithm For Optimal Margin Classifiers. In Proceedings Of The Fifth Annual Workshop On Computational Learning Theory, pp: 144-152. ACM.
15. Gandotra, E., D. Bansal and S. Sofat, 2014. Malware Analysis And Classification: A Survey. Journal Of Information Security.
16. Elhadi, A.A.E., M.A. Maarof and A.H. Osman, 2012. Malware Detection Based On Hybrid Signature Behaviour Application Programming Interface Call Graph. American Journal Of Applied Sciences, 9(3): 283.

17. Tian, R., 2011. An Integrated Malware Detection And Classification System(No. Ph. D.). Deakin University.
18. Skoudis, E. and L. Zeltser, 2004. Malware: Fighting Malicious Code. Prentice Hall Professional.
19. Pramod, D. and R. Raman, 2014. A Study On The User Perception And Awareness Of Smartphone Security. International Journal Of Applied Engineering Research, ISSN, 0973-4562.
20. Ko, C., M. Ruschitzka and K. Levitt, 1997. Execution Monitoring Of Security-Critical Programs In Distributed Systems: A Specification-Based Approach. In Security And Privacy, 1997. Proceedings., 1997 IEEE Symposium On, pp: 175-187). IEEE.
21. Venu, P. and D.V.R. Prasad, N-Gram Analysis In SVM Training Phase Reduction Using Dataset Feature Filtering For Malware Detection.
22. Abou-Assaleh, T., N. Cercone, V. Kešelj and R. Sweidan, 2004. N-Gram-Based Detection Of New Malicious Code. In Computer Software And Applications Conference, 2004. COMPSAC 2004. Proceedings Of The 28th Annual International, 2: 41-42. IEEE.
23. Bergeron, J., M. Debbabi, J. Desharnais, M.M. Erhioui, Y. Lavoie and N. Tawbi, 2001. Static Detection Of Malicious Code In Executable Programs.Int. J. Of Req. Eng, 184-189: 79.
24. Bilar, D., 2006. Statistical Structures: Tolerant Fingerprinting For Classification And Analysis. Proc. Of The Black Hat USA.
25. Burrows, S. and S.M. Tahaghoghi, 2007. Source Code Authorship Attribution Using N-Grams. In Proceedings Of The Twelth Australasian Document Computing Symposium, Melbourne, Australia, RMIT University, pp: 32-39.
26. Prakash, M., U. Gowsika and S. Sathiyapriya, 2015. An Identification Of Abnormalities In Dental With Support Vector Machine Using Image Processing. In Emerging Research In Computing, Information, Communication And Applications, pp: 29-40. Springer India.