Volume 4, Issue 8, August 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Malicious Detection Using Multiple Classification Algorithms & Their Comparison Using Different Clustering Techniques

Ms. Milan Jain
Research Scholar
Department of CSE
Chandigarh Engineering College
Mohali, Punjab, India

Dr. Bikram Pal
Professor
Department of CSE
Chandigarh Engineering College
Mohali, Punjab, India

Abstract: -Modern infrastructures of computer and communication are more prone to variety of attacks. The method used for launching these attacks is by means of malicious software (malware) such as viruses, worms sand Trojan horses, which, when propagate, can cause great damage to commercial companies, governments and private users. The current growth in high-speed Internet connections help malware to spread and infect hosts very rapidly, therefore it is very important to detect and delete new (benign) malware in an effective manner. In this work, we propose three data mining algorithms to produce new classifiers with separate features: RIPPER, Naïve – Bayes and a Multi Classifier system and the comparison between three methods. It comprises of root kit data collection, data pre-processing, and classification and performance evaluation phases.

Keyword: - Data Mining; Virus; Classification; Security; Phases; Prediction

I. INTRODUCTION

Data mining is the process of handle data from different summarizing it into useful information. Data mining is also known as Knowledge Discovery in Data (KDD). Data mining has many applications in security including in national security (e.g., surveillance) as well as in cyber security (e.g., virus detection).Installing a root kit is usually the first thing that an attacker will do after gaining access to a system, as this will ensure that the attack will remain undetected [2].

II. MALICIOUS CODE

Malware includes computer viruses, ransom ware, worms, Trojan horses, root kits, key loggers, dialers, spyware, adware, malicious BHOs, rogue security software, and other malicious programs; the majority of active malware threats are usually worms or Trojans rather than viruses shown in fig 1. Anti-virus vendors are facing huge quantities (thousands) of suspicious files every day. These files are gathered from variety of sources that includes third party providers, files reported by customers and dedicated honey pots either automatically or explicitly.

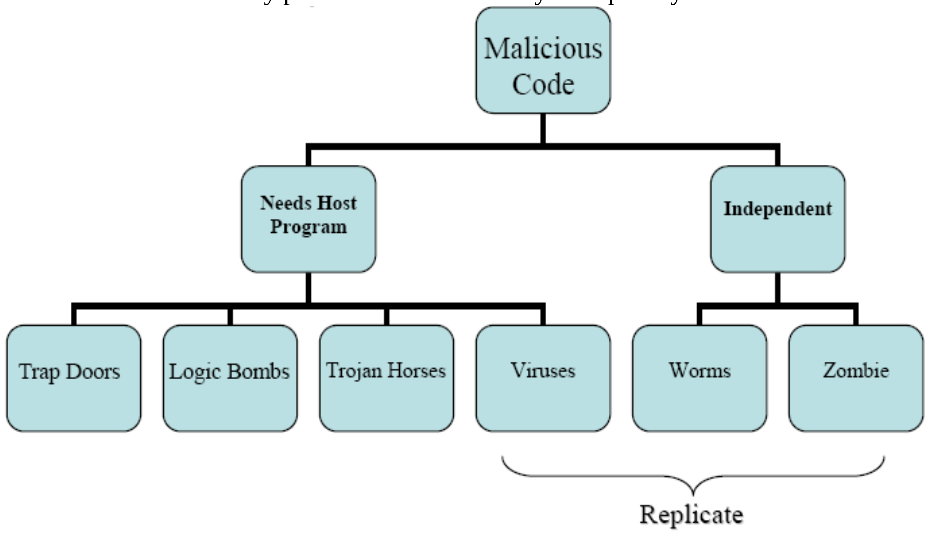

Fig 1:- Malicious Code infrastructure

A. *Polymorphic Malware*

If a virus is in running order so that it look different every times it replicated, but keeping the original code intact. This type of virus is referred as polymorphic virus. It consists of malicious code which is encrypted along with decrypted module. Polymorphic code is a method now commonly implemented in malware that uses a polymorphic generator to mutate the code while keeping the original algorithm intact. A typical implementation of a polymorphic code is to encrypt malware and include the encryptor/descryptor within the code. Polymorphic malwares have specially designed mutation engines. These Polymorphic viruses have a constant payload which is being encrypted with a different descriptor at each instance. Just by switching the order of instructions, a new decryption routine generated by the Mutation Engine. Polymorphic viruses typically encrypt the body of the virus and front-end it with a variable decryption routine. Thus, the body cannot be scanned because it is encrypted, and mutation engine is capable of generating too many different decryption routines [1] [3].

B. Metamorphic Malware

These are a kind of body-polymorphic, where body of virus itself changes from one instance to another. Metamorphic malwares use different types of obfuscation techniques to reprogram themselves into a new transformed code, i.e. similar to original code. The metamorphic nature of the malware enables malicious code to mutate while spreading across the network and making signature based detection completely ineffective [1]. First metamorphic malware recovers its base version and then it incorporates a different metamorphic payload within the base version to evade detection.

How computer viruses and computer worms spread

Have you ever wondered why the Melissa worm spread so quickly, but other computer viruses spread slowly or not at all? Symantec Antivirus Research Center (SARC) has created VBSim(Visual Basic Simulator), a tool to help our customers understand how viruses and worms spread. This program was first demonstrated at the Virus Bulletin 1999 conference in Vancouver, Canada.

III. TECHNIQUES

A. Signature-Based Techniques

Most of the antivirus tools are based on detection with signature based techniques. These signatures are created by examining the disassembled code of malware binary. Various dissemblers and debuggers are available which help in disassembling the portable executables. Disassembled code is analyzed and features are extracted. These features are used in constructing the signature of particular malware family.

B. Behavior-Based Techniques

Here mainly the goal is to analyze the behavior of known or unknown malwares. Behavioral parameters include various factors such as source/ destination address of malwares, types of attachments and other countable statistical features. Consequently each of the above technique can be further applied using static analysis, dynamic analysis or hybrid analysis.

IV. RELATED WORK

Schultz ET. al [1] in the paper" **Data Mining Methods for Detection of New Malicious Executables"** explained various methods for detecting a malicious executables. These new malicious executables are created at the rate of thousands every year and pose a serious security threat. Current anti-virus systems attempt to detect these new malicious programs with heuristics generated by hand.

Kinder et. al [3] presented a paper" **Detecting Malicious Code by Model Checking**". They explained a model checking method for detecting malicious code. In this paper, they present a flexible method to detect malicious code patterns in executables by model checking. While model checking was originally developed to verify the correctness of systems against specifications, they argue that it lends itself equally well to the specification of malicious code patterns.

V. ALGORITHMS DISCUSSION

The various algorithms are used to explore the application of data mining techniques for detection of malevolent code.

A. RIPPER

The first algorithm RIPPER is an inductive rule learner. This approach developed a detection model Consists of resource rules that was developed to detect examples of malicious executables. This algorithm is using a lib BFD data as characteristics. RIPPER is a rule-based learning approach that is building a set of rules that is able to determine the classes while reducing the ambiguities. The ambiguity is given by the training examples of unclassified by the rules.

$$W(R) = (p-n)/(p+n)$$

p: number of positive examples covered by the rule in the validation set

n: number of negative examples covered by the rule in the validation set

B. Naive Bayes

The next classifiers we are using is a Naive Bayes classifier. It calculate the likelihood that a program is having malevolent code given the features that are present in the program. This approach used both string and byte sequences data for computing a probability of a binary's malicious code having some features. The main assumption in this Algorithm is that the binaries contain same features such as signatures and machine instructions.

$$\mathrm{P}(c/x) = \frac{P(x/c)\,P(c)}{P(x)}$$

$$\mathrm{P}(c/X) = \mathrm{P}(x_1/c) \times \mathrm{P}(x_2/c) \times \ldots\ldots \times \mathrm{P}(x_n/c) \times \mathrm{P(c)}$$

- $\mathrm{P}(c/x)$ is the posterior probability of class (target) given predictor (atribute).
- P(c) is the prior probabiltiy of class.
- $\mathrm{P}(x/c)$ is the likelihood which is the probabilty of predictor given class.
- P(x) is the prior probabilty of predictor.

C. Multi-Naive Bayes

The next data mining algorithm is Multi-Naïve Bayes. This algorithm was importantly a collection of Naïve Bayes algorithms that supported on whole categorization for an example. In Naive Bayes algorithm, it is able to classify the examples in the test set of malicious executables program and this counted as a choice. This method was needed as it is using a machine with 1GB of RAM; the size of the binary data was very big to get into memory. Thus to solve this problem we divided it into smaller parts that could easily get into memory and hence training the naïve bayes algorithm. The Naive Bayes algorithm required a table chart of all strings or bytes to evaluate its possibilities. In every classifier, there is a rule set. The divination of the Multi-Naive Bayes algorithm is the multiplication of all the predictions of the Naive Bayes classifiers. In short it is used to calculate a collection of cluster for ambiguity or malicious code.

$$\mathrm{p}\,(F/C) \;\; = \frac{(\sum_i F_i)!}{\prod_i F_i\,!} \prod_i {p_i}^{F_i}$$

Hence concluding it will generate set of rules and multiplication value for prediction of classifiers.

D. K- nearest neighbor

The goal of the algorithm is to lead an attribute weight vector which improves KNN classification.

• Chromosomes are vectors of real-valued. Each chromosome is a vector of decimal numbers between 0 and 1 inclusive. A vector value is associated with each classification attribute and one is associated with each of the k neighbors. Thus the length of the vector is the number of attributes plus k. The initial population of chromosomes in each run of the KNN algorithm was randomly generated.

VI. PROPOESD WORK

Basically the phase of classification tells that which algorithm has shown the best performance i.e. which yields the highest accuracy after the execution of all the preceding algorithms.

Proposed Algorithm

Step1. Data acquisition is done.

Step2. Applying clustering algorithm based on KNN.

$$\mathrm{D}\,(\mathrm{x},\,\mathrm{y}) = \sqrt{\sum_{j=i}^{d} w_{j\,2}(x_i - y_i)^2}$$

Step3. Find ambiguity based on graphic technique.

Step4. Apply multi naïve bayes algorithm for malicious detection.

Step5. Multi naïve bayes

$$\mathrm{p}\,(F/C) \;\; = \frac{(\sum_i F_i)!}{\prod_i F_i\,!} \prod_i {p_i}^{F_i}$$

Step6. Training is done for 50 percent of data.

Step7. Target value will be varied according to ambiguity and permission axis.

Step8. Test the remaining data according to target variation.

Step9. According to estimated testing malicious code has been detected and shown from the dataset or can be mitigate from dataset.

Thus it can be said that best classifiers are used to detect the ambiguity in the data and gene rates Root kit value for better prediction. Thereby comparing all the classifier prediction valued analyses. According to the comparison of the parameters used for the performance evolution of the system it can be predicted which approach provides better results.

VII. RESULT AND DISCUSSION

The performance evaluation of system and their experimental results are shown in the following section using the tool MATLAB (MATrix LABoratory) for numerical computation and visualization.

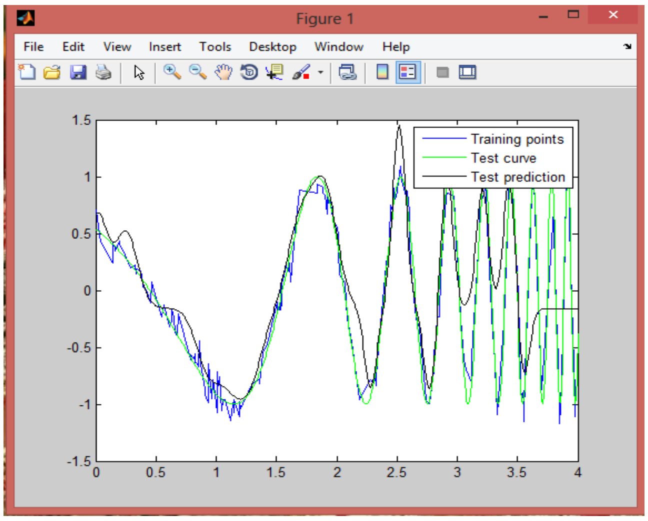

Fig 2:- Performance analyses of malicious code detection.

In fig 2 estimated results are shown on the basis of training and testing of dataset and then their analyses is given.

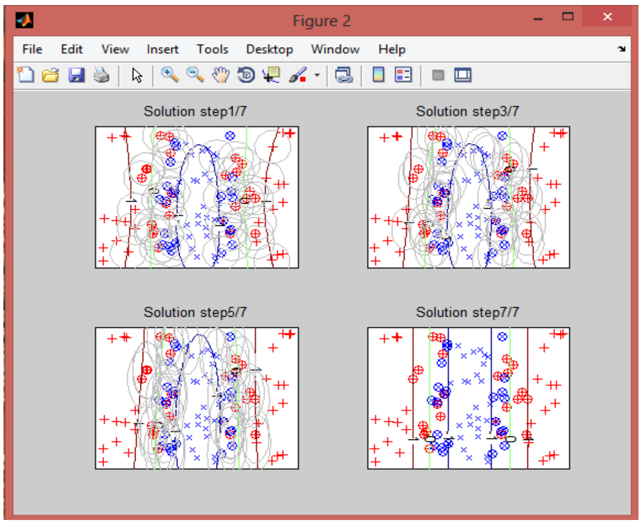

Fig 3:- Represent detection of malicious data

Data is distinguished according to clusters presented in fig 3. Here four sections are divided and these are for seven iterations identified by loop in black. In first section one Malicious is found, in second another Malicious are found, in third classification is done and in last malicious data is shown. Blue color is showing the malicious data and then the output is given after we refine it.

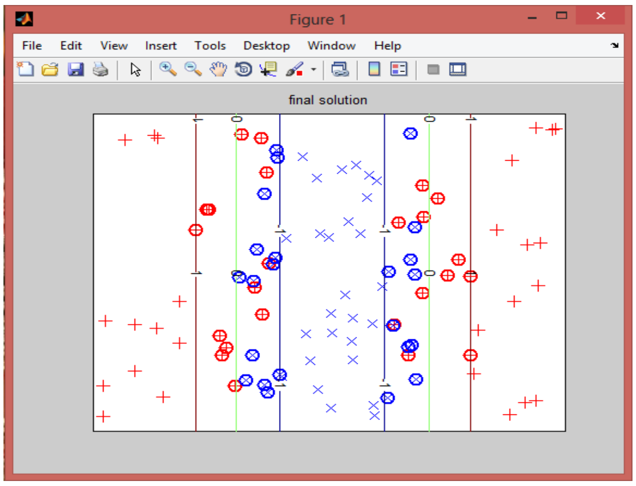

Fig 4:-Represent malicious data

Among four segments the segment which showing malicious data is shown in fig 4. Here circles are executable which are already executable that are in red and which we assume that are in blue**.**

VIII. COMPARISON TABLE

The following measures were utilized for the rootkit prediction using classification algorithms along with the clubbing of clustering algorithms in data mining and then their further comparison is shown to evaluate the system performance as under.

Algorithms used	System performance evaluation					
	Precision	Recall	F-measure	Time in minute	Accuracy %	MCC
Decision tree with ripper	0.7	.4	.023	0.10	81.2	0.51
Decision tree with naïve bias	0.6	0.5	.004	0.9	83.1	0.55
Decision tree with multi naïve bias	.06	0.02	.016	0.6	84.2	0.61
KNN with ripper	0.7	0.3	0.5	0.11	87.1	0.59
KNN with naive bias	0.4	0.2	0.6	0.12	81.9	0.65
KNN with multi naïve bias	0.8	0.5	0.7	0.5	90	0.72
SVM with ripper	0.04	0.1	0.4	0.7	87	0.67
SVM with naive bias	0.2	0.4	0.03	0.8	84	0.69
SVM with multi naïve bias	0.5	.3	0.2	0.10	81	0.63

This table includes results of all algorithms for some parameter like accuracy, precision, recall and f-measure etc.From the results it is concluded that results of KNN with multi naïve bias is better than all others.

IX. CONCLUSION AND FUTURE WORK

Prepositions are often among the most frequent words in a language. The main challenging issue in preposition phrase is a malicious problem. In resolving the malicious for preposition phrase various techniques are used like fuzzy set, possibility theory, classifier, nearest neighbor method, genetic algorithm etc. This research describes different types of techniques to resolve malicious problems in preposition phrase and also different techniques used in literature review. The proposed approach has been implemented and tested using a set of test case. One of the obtained results has been presented and discussed in this paper showing that KNN with multi naïve bias is better than all others. Here best results represent system precession and system recall is good.

Thus the proposed approach has been implemented and better results have been obtained in order to make the running algorithms better I n both time and space. The overall obtained results indicate the approach is viable as genetic algorithm is used to optimize the mining. We can also extend our algorithm for the utilization of byte sequence in future. It is also expected that this process will lead to enhancement of good algorithms like J48 so that the computer systems can be protected from malwares and consume less time.

REFERENCE

[1] Matthew G. Schultz *"Data Mining Methods for Detection of New Malicious Executables",a*cademiccommons.columbia.edu/download/.../binaryeval-ieeesp01.pdf

[2] Dr. R. Geetha Ramani, Suresh Kumar. S,Shomona Gracia Jacob "Root kit (Malicious Code) Prediction through Data Mining Methods and Techniques", 978-1-4799-1597-2/13/$31.00 ©2013 IEEE.

[3] Johannes Kinder, *"Detecting Malicious Code by Model Checking*"pure.rhul.ac.uk/portal/files//17566588/mcodedimva05.pdf.

[4] Bhavani Thuraisingham, *"Data Mining for Security Applications"*, IEEE/IFIP International Conference on Embedded and Ubiquitous Computing, 2008.

[5] Kirti Mathur, *" A Survey on Techniques in Detection and Analyzing Malware Executables",* International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 4, April 2013.

[6] Guillermo Suarez-Tangle, *"Evolution, Detection and Analysis of Malware for Smart Devices"* IEEE Communications Surveys & Tutorials, Accepted for Publication, pp.1-27, 2013

[7] Parisa Bahraminikoo, *"Utilization Data Mining to Detect Spyware"* IOSR Journal of Computer Engineering (IOSRJCE) Volume 4, Issue 3, pp.01-04, 2012