DATA MINING FOR MALICIOUS CODE DETECTION AND SECURITY SYSTEM APPLICATION

Mr. Mahesh N Gunjal, Ms. Ketaki R Takawale,

Mr. Akshay B Raut, Ms. Sonam P Jadhav,

Prof.Vinod Wadne

Department of Computer Engineering,

JSPM's Imperial College of Engineering & Research,

Wagholi,Pune, India.

ketakitakawale@gmail.com,

Abstract— Over the past few years, a new computer security problem has arisen, malwares and spywares. Anti-Virus techniques cannot deal with such applications or code, due to their silent infection techniques and their differences from a regular virus. Various Anti-Spyware programs have been implemented as a counter-measure but most of these programs work using the signature method, which is weak against new spyware. We has presented a data-mining framework that detects new, previously unseen malicious executables by checking their source code. This paper takes our work as a candidate and applies its techniques against a new spyware dataset collected, to see whether their techniques can be used against this new threat.

Keywords— Malicious Code Detection, Data Mining, Computer Security, Prediction, Machine learning, wamp server, Mining algorithms.

Introduction:

Datamining is the actual process of handling data from different sources summarizing it into useful information. Data mining can also be termed as Knowledge Discovery in Data (KDD). Data mining has many applications in security including in national security (e.g., surveillance,etc) as well as in cyber security (e.g., virus detection,etc).Installing a rootkit 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.We have used Bayesian classification, in Bayesian classification we have a hypothesis that the given data belongs to particular class. We then calculate the probability for the hypothesis to be true. This is among the most practical approaches for certain types of problems. The approach requires only one scan of the whole data. Also, if at some stage there are additional training data, then each training example can incrementally increase/decrease the probability that a hypothesis is corrected.

Methodology:

Following are the algorithms used in proposed system:

1.Ripper algorithm:

The first algorithm ripple is an inductive rule trainee. This approach generated a detection model consists of resource rules that was developed to detect examples of malicious files. This algorithm is using a Lib BFD data as characteristics. RIPPLE is rule -based trainee approach of building set of rules i.e able to appoint classes while rising the ambiguity is given by the training examples of unclassified by the rules.[8]

2.Multi-navie byes:

The Naïve Bayes model is a very simplified Bayesian probability model used here. In this system, consider the probability of an end result of given several related evidence variables in data. The max probability of end result is encoded in the model along with the probability of the evidence variables occurring given that the end result occurs. The probability of an existing evidence variable given that the end result occurs is assumed to be independent of the probability of other evidence variables given that end results occur.

We have proposed a framework of Intrusion Detection System using multi Naïve Bayes algorithm. The framework classifies the input dataset with KDD cup dataset.[10] The Framework detects attacks in the datasets using the multi naive Bays algorithm. Compared to the neural based approach, our approach achieve higher detection rate, less time consuming and has low cost factor. However, it generates somewhat more false positive. **Hence we used multi-navie byes algorithm.**

The next data mining algorithm is Multi-Naïve Bayes. This algorithm was a collection of Naïve Bayes algorithms that supported overall concept for an example. In multi Naive Bayes algorithm, it can classify the examples in the test set of malicious executables program and this counted as a probability of occurrences. 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 in to memory. Thus to solve this problem we divided it into smaller parts that could easily get into memory and hence training the multi-naïve bayes algorithm. The Naive Bayes algorithm required able chart of all strings or bytes to evaluate its possibilities. In every classifier, there is a rule set. The classification of the Multi-Naive Bayes algorithm is the multiplication of all the predictions of the Naive Bayes classifiers. Shortly it is used to calculate a collection of data intruders for ambiguity or malicious code. Hence it will generate set of rules and multiplication value for prediction of classifiers.

Algorithm with its calculations:

Using Bayes' theorem, the conditional probability can be decomposed as

$$p(C_k|\mathbf{x}) = \frac{p(C_k)\ p(\mathbf{x}|C_k)}{p(\mathbf{x})}.$$

In plain English, using Bayesian probability terminology, the above equation can be written as

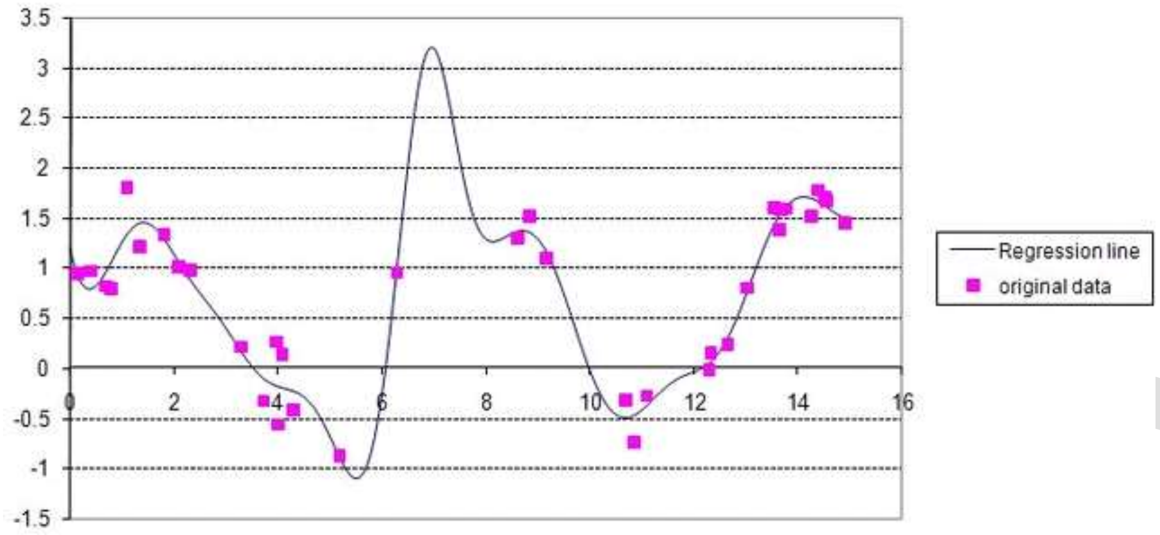
$$\text{posterior} = \frac{\text{prior} \times \text{likelihood}}{\text{evidence}}.$$

The discussion so far has derived the independent feature model, that is, the naive Bayes probability model. The naive Bayes classifier combines this model with a decision rule. One common rule is to pick the hypothesis that is most probable; this is known as the maximum a posteriori or MAP decision rule. The corresponding classifier, a Bayes classifier, is the function that assigns a class label $\hat{y} = C_k$ for some k as follows:

$$\hat{y} = \underset{k \in \{1,\dots,K\}}{\operatorname{argmax}}\ p(C_k) \prod_{i=1}^{n} p(x_i|C_k).$$

3.K-nearest neighbor -

The actual goal of the algorithm is to check an attribute weight vector which improves overall KNN classification. Likewise Chromosomes are vectors of real-valued. Each chromosome here is a vector of decimal numbers between 0 and 1.A vector value is linked with each classification attribute and one is linked with each of the k neighbors. Thus the length of the vector i.e. chromosomes is the number of attributes plus k.[9] The initial population of chromosomes(vector) in each run of the KNN algorithm was randomly generated. The simplest way of doing this is to use K-nearest Neighbor's-nearest neighbor algorithm (KNN) [7] is part of supervised learning that has been used in many applications in the field of data mining, statistical pattern recognition and many others.KNN is a method for classifying objects based on closest training examples in the feature space. An object is classified by a majority vote of its neighbors. K is always a definite integer. The neighbors are taken from a set of objectives for which the correct classification is known. It is usual to use the Euclidean distance, though other distance measures such as the Manhattan distance could in principle be used instead.



The algorithm on how to compute the K-nearest neighbors is as follows:

Determine the parameter K = number of nearest neighbors beforehand. This value is all up to you.

Calculate the distance between the query-instance and all the training samples. You can use any distance algorithm. Sort the distances for all the training samples and determine the nearest neighbor based on the K-th minimum distance. Since this is supervised learning, get all the Categories of your training data for the sorted value which fall under K. Use the majority of nearest neighbors as the prediction value.

For a tutorial and implementation of the different distances

4.TEST RESULTS OF ALOGRITHM IMPLEMENTATION

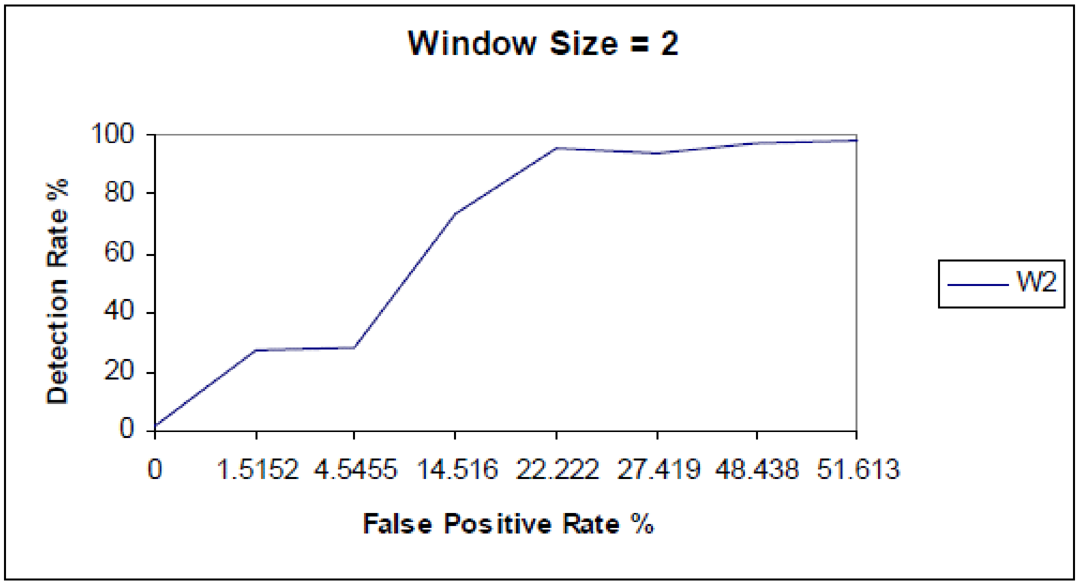
- **TEST RESULT**

To evaluate our system we are interested in several
Quantities, just like:

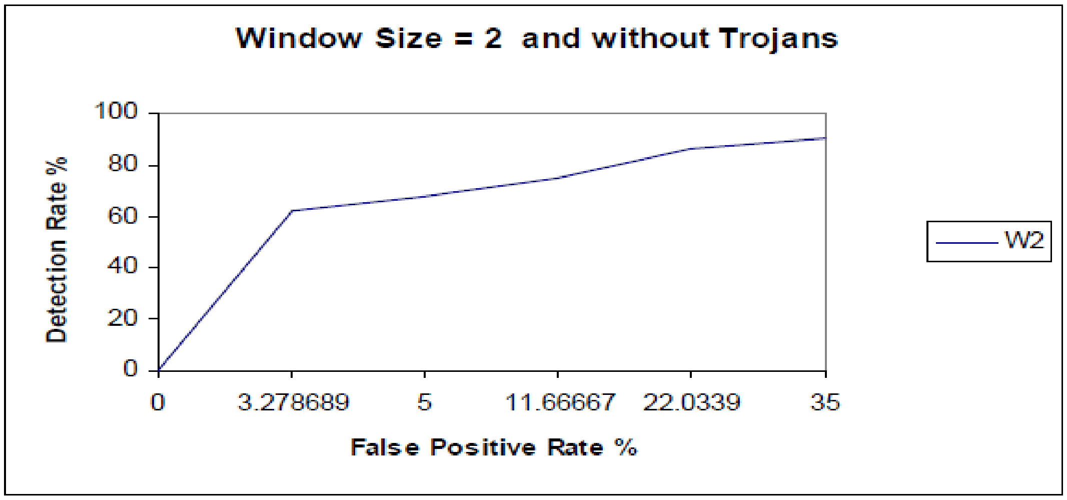
1. True Positives (TP), the number of malicious executable examples classified as malicious executables
2. True Negatives (TN), the number of benign programs classified as benign.
3. False Positives (FP), the number of benign programs classified as malicious executables
4. False Negatives (FN), the number of malicious executables classified as benign binaries.

The "detection rate" of the classifier is the percentage of the total malicious programs labeled malicious. The "false positive rate" is the percentage of benign programs which were labeled as malicious. The "overall accuracy" is the percentage of true classifications over all data.

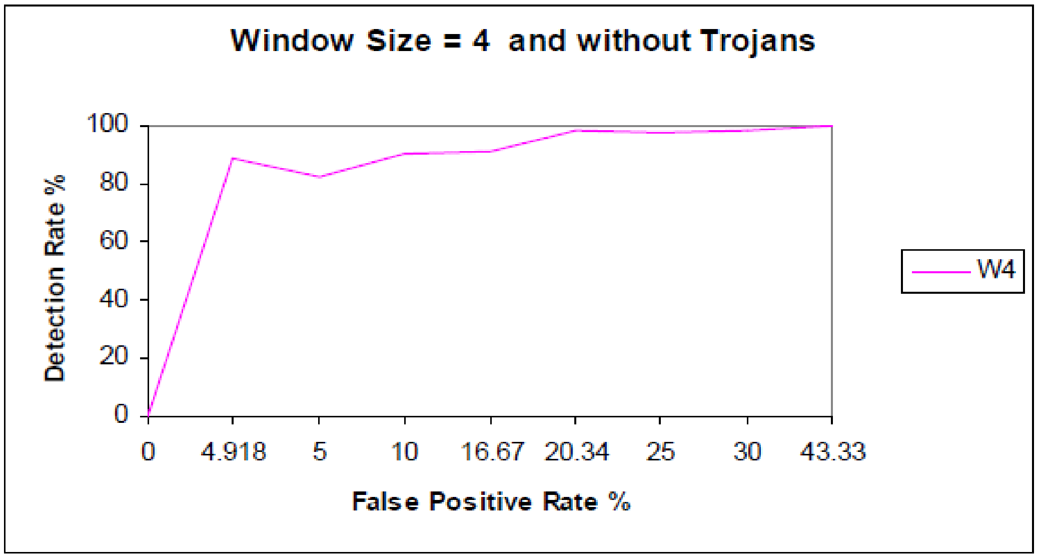
The first test is done using a window size of 2 and its results curve is shown in **Table 1.**



Results were better for this case, but still the overall precision was rather low, so I decided to enquitry the reasons by looking at the classification test results. I consummation that a class of spyware called Trojans had a quite low detection rate. These are different from or uninspired spyware, since they are quite complex programs and their binary size were rather big compared to other files in the set of data. I think that these programs may be the reason for the high false positive rate and low detection rate, so I run another test for a window size of 2, after excluding the 59 Trojans from the dataset.



As the **Table 3** shows, the overall accuracy has improved for the window size of 2 and we reach 80% detection rate before a false positive rate of 15%. Also we score better for low false positive rates. These results encouraged for a window size of 4 test without the Trojans in the dataset and the results of this test are shown in **Table 4.**



We had 80% detection rate even before the false positive rate of 4% and overall accuracy has been improved. Below you can see best overall accuracy results for each run in **Table 5.**

	TP	TN	FP	FN	Detection Rate	False Positive Rate	Overall Accuracy
Window Size=2	118	49	14	5	95.93	22.22	89.78
Window Size=4	119	46	17	4	96.75	26.98	88.71
Window Size=2 w/o Trojan	96	46	13	15	86.49	22.03	83.53
Window Size=4 w/o Trojan	99	58	3	12	89.19	**4.92**	**91.28**

ACKNOWLEDGMENT:

We would like to sincerely thank Prof. Vinod Wadne sir our guide for her support and encouragement

CONCLUSION:

Data mining-malicious code detectors have been very resultantly in detecting malicious code such as viruses and worms. Henceforth we successfully implemented Data mining code detection to provide solutions such as intrusion detection and auditing, etc. [2] for all above we have successfully implemented detection. Other applications are also successfully implemented data mining for malicious code detection such as worm detection, managing firewall policies. Secondly concluded the various types of algorithms to detect the intrusions n set probability or choice to check and remove up threats successfully from mined data. Algorithm detects and removes all threats include non real-time threats and real-time threats. Also implemented resulted Data mining applied for credit card fraud detection and biometrics related applications.[2]Progress has been made on topics such as stream data mining; there is still a lot of work to be done here and concluding we have discussed the consequences to privacy for Data mining. It is expected that this procedure will lead to the development of better algorithms for identifying the root kit that has infected a system.

REFERENCES:

[1].Dataminingfor malicious codedetection and security application http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=52487&queryTxt%3DData+Mining+for+Malicious+Code+Detection+And+Security+Applications
[2].http://www.mjret.in/V2I3/M2-2-3-7-2015.pdf
[3].I.H. Witten, E. Frank, Data Mining: Practical Machine Learning Tools and Techniques, 2nd Ed.Morgan Kaufmann, 2005.
[4].Data mining for malicious code detection and security application
http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=606118012-14 Sept. 2011 Intelligence and

Security Informatics Conference (EISIC), 2011 European

[5].J. Z. Kolter and m. A. Maloof, "learning to detect malicious executables in the wild," in Proceedings of

Theacm symp. On knowledge discovery and data mining (kdd), pp. 470-478, August 2004.

[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].http://www.bing.com/search?q=knn+algorithm&qs=SC&pq=k+n+al&sc=8-6&sp=1&cvid=0c8919067b414e1bae338526ee0f6b60&FORM=QBRE

[8].M .G Schultz,E.Eskin.E Zadok and S .JStolfo,"Datamining methods for detection of new malicious executables",Proccedings of the 2001IEEESymposioum on Security and privacy,IEEE Computer Society.

[9] K-nearest neighbor

https://www.google.co.in/?fe_rd=cr&ei=rK5RVZ_ILafW8gfzn4GYDg&gws_rd=ssl#q=+Chromosomes+are+vectors+of+real-valued.+Each+chromosome+here+is+a+vector+of+decimal+numbers+between+0+and+1.A+vector+value+islinked+with+each+classification+attribute+and+one+is+linked+with+each+of+the+k+neighbors.+Thus+the+length+of+the+vector+i.e+chromosomes+is+the+number+of+attributes+plus+k.+The+initial+population+of+chromosomes%28vector%29+in+each+run+of+the+KNN+algorithm+was+randomly+generated.The+simplest+way+of+doing+this+is+to+use+K-nearest+Neighbor.K-nearest+neighbor+algorithm+%28KNN%29+is+part+of+supervised+learning+that+has+been+used+in+many+applications+in+the+field+of+data+mining%2C+statistical+pattern+recognition+and+many+others.

[10] kddcup dataset -

https://www.google.co.in/?gfe_rd=cr&ei=rK5RVZ_ILafW8gfzn4GYDg&gws_rd=ssl#q=We+have+proposed+a+framework+of+Intrusion+Detection+System+using+multi+Na%C3%AFve+Bayes+algorithm.+The+framework+classifies+the+input+dataset+with+KDD+cup+dataset.+The+Framework+detects+attacks+in+the+datasets+using+the+multi+naive+Bayes+algorithm.+Compared+to+the+neural++based+approach%2C+our+approach+achieve+higher+detection+rate%2C+less+time+consuming+and+has+low+cost+factor.+However%2C+it+generates+somewhat+more+false+positive