Villanova Journal of Science, Technology and Management
Vol. 1. No. 1. 2019. ISSN: 2672-4995 (Online) 2672-4987 (Print)

POST ANALYSIS OF SNORT INTRUSION FILES USING DATA MINING TECHNIQUES: DECISION TREE AND BAYESIAN NETWORK

JEGEDE T.J
Department of Computer Science
University of Ibadan
Oyo State, Nigeria
top4real@gmail.com

ASANBE M.O
Department of Computer Science
Villanova Polytechnic, Imesi-Ile,
Osun State, Nigeria
moasanbe@gmail.com

ABSTRACT

Network security is a crucial information technology activity today. Intrusion Detection Systems (IDS) are among the fastest growing technologies in computer security domain. These systems are designed to identify/ prevent any hostile intrusion into a network. Most conventional intrusion detection systems have limitations in the way they log their alerts which snort exhibit is known as the infidelity issue, that is to say snort IDS does not infer the behavior of the network traffic generated, which can result in misinterpretations. Therefore in this project data mining techniques was applied to the logged alert in order to extract hidden knowledge of the traffic pattern. This research investigates the network domain of data mining using the network alerts generated from snort intrusion detection system in order to mine the alerts for re-classification. The data comprised of nine sixty (960) records of alerts. Classification task is used to evaluate the alerts making use of Bayesian Network and Decision Tree methods. The output of the two classification methods – Bayesian Network and Decision Tree are compared to determine the one that gives the best classification results. At the modeling stage, open source software called WEKA 3.6.13 was used. The data set was divided into two sets – - Training and Testing. Sixty eight percent (68%) was used for training while thirty two percent (32%) was used for testing. From the output generated from the experiment, Decision tree outperformed Bayesian network in most aspects and the existing snort with data mining is more reliable and efficient over snort alone. The results obtained from the analysis clearly demonstrated that Decision tree outperformed Bayesian network. Decision tree demonstrated a superior performance than Bayesian network in term of the number of correctly classified instances and also in terms of Root Mean Squared Error, Root Relative Squared Error, Mean Absolute Error, Relative Absolute Error. Bayesian

Network outfitted Decision Tree in time taken to build the model but performed poorly at the classification. The time taken for naïve bayes and decision tree classifiers are 0.12 and 0.32 seconds respectively.
Keywords: Intrusion detection, Network, Network threats, Protocols, Data mining, Classification

1. INTRODUCTION

Over the past ten years, the number of threats on information systems has significantly increased. Consequently, classical information security technologies such as authentication and cryptography have gained more and more attention. Meanwhile, intrusion detection systems (IDSs) have emerged as a new approach to detect and protect information systems [1]. An IDS monitors an information system for evidence of attacks. Once attacks have been detected, the IDS raises alerts to report them. The alerts are presented to experts or a knowledge system that evaluates them and initiates an adequate response. It is a challenging task to evaluate intrusion detection alerts and generate an appropriate response [2].

IDSs provide an extra layer of defense to computer networks by gathering and analyzing information in a network in order to identify possible security breaches. If an intrusion is detected, IDS generates a warning called an alert or alarm. Generally, there are two broad of classes of IDSs: signature based and anomaly based. The signature based IDSs generally recognize patterns of attack. This IDS essentially contains attack descriptions or signatures and match them against the audit data stream, looking for evidence of known attacks. A signature -based IDS works similar to anti-virus software. It employs a signature database of well-known attacks, and a successful match with current input raises an alert. Signatures generally target widely used applications or systems for which security vulnerabilities are widely advertised. Anomaly based IDS usually look for deviations from normal usage behavior in order to identify abnormal behavior. Generally, the anomaly detection techniques rely on models of the normal behavior of a computer system. The anomaly based IDSs may focus on the users, the applications, or the network.

Despite several successes linked to IDSs, they are plagued by several issues making the art of accurately detecting intrusions far from perfect [3]. Among the issues that contribute to poor performance of IDSs are: production of overwhelming number of alerts and high number of false positive alerts. It is estimated that an IDS may generate tens of thousands alerts per day. The vast imbalance between interesting alerts and non-interesting alerts has undoubtedly

undermined the performance of IDSs [4]. As a result, the important alerts might be misclassified, misinterpreted, delayed or ignored [5].

According to Georgios et al. [6], over the last few years, the research in intrusion detection has focused on the post processing of alerts in order to identify and separate interesting alerts. Identifying and separating interesting alerts has always been challenged by several issues such as IDSs use general signatures that hardly capture all variations of known attacks hence difficult to differentiate legitimate activities from the illegitimates ones. The amount and types of data in corporate are more, so that the vulnerability is increases on data. To manage, analyze and identify the problems on data we need data mining techniques. Data mining technique plays a vital role in intrusion detection. Some of the application of data mining in intrusion detection are classification, clustering and frequent pattern matching. Intrusion means the vulnerability of the network such as denial of services (DOS), spoofing, spamming, etc. The process of intrusion detection is finding the malicious activities which are in the network to prevent the data integrity, security, confidentiality, worms, viruses and availability. [7].

Snort intrusion detection system alerts will be re-categorized using data mining approach in order to process the network log to help detect network traffic pattern on the campus network. Snort is a fast, signature-based and open-source intrusion detection system. Snort has received great tolerance in the Intrusion Detection System (IDS) market and has been widely recognized as the reliable open source tool [8]. Snort is capable of performing real-time traffic analysis and packet logging on the network. It performs protocol analysis and can detect variety of network attacks by using signature matching algorithms. Snort can be configured as a packet sniffer, packet logger and Network Intrusion Detection System (NIDS). As packet sniffer, it reads the packets off the network. In a packet logger mode, it logs packets to the storage device. NIDS mode enables the Snort to analyze the network traffic against set of defined rules in order to detect intrusion threats.

The current snort intrusion detection detects and profiles malware based on signature profile of the threats which are predetermined. However, malware exhibits different behaviors which snort cannot show. This has caused a major performance bottleneck in the detection system. Therefore, this study proposes to use data mining techniques to process the network input data to help expose

malware and non-malware traffic on the network based on alerts generated by snort intrusion detection system.

This study proposed is to re-categorize alerts from snort intrusion detection system using two data mining approaches: Decision Tree and Bayesian Network. Efforts will be directed as developing a database of malware behavior profiles based on snort intrusion detection system alerts and analyzing the alerts so as to re-classify traffic pattern.

2. REVIEW OF RELATED WORKS

Recently, researchers have being exploring the field of network security as there are ample number of research papers discussing various problems within the network and providing examples for successful solutions reached by using data mining. Various machine learning approaches like Association Rule, Support Vector Machine, Decision Tree, Random Forest, Naive Bayes and Clustering have been proposed for detecting and classifying unknown samples into either known malware families or underline those samples that exhibit unseen behavior.

Schultz *et al*, [9] (2001) were the first researchers to introduce the concept of data mining for detecting malwares. They applied three different static features for malware classification: Portable Executable (PE), strings and byte sequences. In the PE approach, the features (like list of DLLs used by the binary, the list of DLL function calls, and number of different system calls used within each DLL) are extracted from DLL information inside PE files. Strings are extracted from the executables based on the text strings that are encoded in program files. The byte sequence approach uses sequences of n bytes extracted from an executable file. They applied a dataset consisted of 4266 files including 3265 malicious and 1001 benign programs. A rule induction algorithm called Ripper was utilized to find patterns in the DLL data. A learning algorithm Naive Bayes was used to find patterns in the string data and n-grams of byte sequences were used as input data for the Multinomial Naive Bayes algorithm. The Naive Bayes algorithm, taking strings as input data, gives the highest classification accuracy of 97.11%. The authors exacted that the rate of detection of malwares using data mining method is twice as compared to signature based method. Later on their results were amended by Kolter, et al. [10] (2004). They used n-gram (instead of non-overlapping byte sequence) and data mining method to detect malicious executables. They used different classifiers including Naive-Bayes, Support

Vector Machine, Decision Tree and their boosted versions. They concluded that boosted decision tree generates the best classification results.

Kong *et al*, [11] presented a model for automated malware classification based on structural information (function call graph) of malwares. After extracting the fine grained attributes based on function call graph for each malware sample, the similarity is evaluated for two malware programs by applying discriminate distance metric learning which clusters the malware samples belonging to same family while keeping the different clusters separate by a marginal distance. The authors then applied an ensemble of classifiers that learn from pair wise malware distances to classify malwares into their respective families.

Osunade *et al*, [12] presented a threat characterization framework for attacks from the victim and the aggressor perspective of intrusion using data mining technique. The data mining technique integrates both Frequent Temporal Sequence Association Mining and Fuzzy Logic. Apriori Association Mining algorithm was used to mine temporal rule patterns from alert sequences while Fuzzy Control System was used to rate exploits. The experiment shows that accurate threat characterization in multiple intrusion perspectives could be actualized using Fuzzy Association Mining. Also, the results proved that sequence of exploits could be used to rate threat and are motivated by victim properties and attacker objectives.

Siddiqui *et al*, [13] applied variable length instruction sequence along with machine learning for detecting worms in the wild. Before disassembling the files, they detect compilers, packers. Sequence reduction was carried out and decision tree and random forest machine learning models were applied for classification. They tested their method on a data set of 2774 including 1444 worms and 1330 benign files.

Anderson *et al*, [14] presented a malware detection algorithm based on the analysis of graphs constructed from dynamically collected instruction traces. A modified version of Ether malware analysis framework was used to collect data. The method uses 2-grams to condition the transition probabilities of a markov chain (treated as a graph). Machinery of graph kernels is used to construct a similarity matrix between instances in the training set. Kernel matrix is constructed by using two distinct measures of similarity: a Gaussian kernel, which measures local similarity between the graph edges and a spectral kernel which measures global similarity between the graphs. From the kernel matrix, a support vector machine is trained to classify the test data. The performance of multiple kernel learning method used in this work is demonstrated by

discriminating different instances of malware and benign software. Limitation of this approach is that the computation complexity is very high, thus limiting its use in real world setting.

Tian *et al*, [15] applied an automated tool for extracting API call sequences from executables while these are running in a virtual environment. They used the classifiers available in WEKA library to discriminate malware files from clean files as well as for classifying malwares into their families. They used a data set of 1368 malwares and 456 cleawares to demonstrate their work and achieved an accuracy of over 97%.

Lee *et al*, [16] proposed a method that clusters the malicious programs by using machine learning method. All the samples of data set are executed in a virtual environment and system calls along with their arguments are monitored. A behavioral profile is created on the basis of information recorded regarding sample's interaction with system resources like registry keys, writing files and network activities. The similarity between two profiles is calculated and then by applying k-medoids, different samples are grouped into different clusters. After completing the training process, the new and unknown samples are assigned to the cluster having medoid closer to the sample i.e. nearest neighbor.

Santos *et al*, [17] proposed a hybrid unknown malware detector called OPEM, which utilizes a set of features obtained from both static and dynamic analysis of malicious code. The static features are obtained by modeling an executable as a sequence of operational codes and dynamic features are obtained by monitoring system calls, operations and raised exceptions. The approach is then validated over two different data sets by considering different learning algorithms for classifiers Decision Tree, K-nearest neighbor, Bayesian network, and Support Vector Machine and it has been found that this hybrid approach enhances the performance of both approaches when run separately.

Raftopoulos *et al*, [18] conducted a sophisticated experiment to assess the security of suspected infected systems in a production environment using data from several independent sources, including intrusion alerts, blacklists, host scanning logs, vulnerability reports, and search engine queries. They found that the false positive rate of their heuristic was 15% and analyze in-depth the root causes of the false positives. Having validated there heuristic, they applied it to their entire trace, and characterize various important properties of 9 thousand infected hosts in total. For example, they found that among the infected hosts, a small number of heavy hitters originate most outbound attacks and that future infections are more likely to occur close to already infected hosts.

Subbulakshmi *et al.,* [19] described a two-phase automatic alert classification system to assist the human analyst in identifying the false positives. In the first phase, the alerts collected from one or more sensors are normalized and similar alerts are grouped to form a meta-alert. These meta-alerts are passively verified with an asset database to find out irrelevant alerts. In addition, an optional alert generalization is also performed for root cause analysis and thereby reduces false positives with human interaction. In the second phase, the reduced alerts are labeled and passed to an alert classifier which uses machine learning techniques for building the classification rules. This helps the analyst in automatic classification of the alerts. The system is tested in real environments and found to be effective in reducing the number of alerts as well as false positives dramatically, and thereby reducing the workload of human analyst.

Srinivas *et al.,* [20] (2007) presented the state-of-the-art of the evolution of intrusion detection technology and address a few intrusion detection techniques and IDS implementations. An overview of computer attack taxonomy and computer attack demystification along with a few detection signatures was presented. Special emphasis is also given to the current IDS limitations. Further they described few obfuscation techniques applied to recent viruses that were used to thwart commercial grade antivirus tools.

3. EXPERIMENTAL DESIGN AND PROCEDURES

3.1 Data Acquisition

The data used in this research work was captured from the Information Technology and Media Services (ITeMS), University of Ibadan, Ibadan. A live Snort Intrusion Detection System was deployed on the wireless server called APNearU. Some malware traffic were downloaded from malware-traffic-analysis.net which is a website used by academic security researchers. The data was in packet capture (pcap) format. The generated alerts from Snort have a lot of insignificant information, which needs to be eliminated. The essential details in each alert includes IP Address of source and destination host, alert identification, time stamp, alert length and source with destination port number. The data include some categories of variables, the records of the packet data variables are shown in table.

Table 1: Packet Data Variables

S/N	VARIABLE NAME	VARIABLE FORMAT	VARIABLE TYPE
1.	Alert Identification	302, 416, 531,...	Numeric
1.	Alert Time Stamp	07/17- 10:59:06,…	Numeric
2.	Source IP Address	192.168.101.66,...	Numeric
3.	Destination IP Address	100.168.101.53,...	Numeric
4.	Protocol	tcp, http, dns,…	Nominal
5.	Alert Length	60, 1531,…	Numeric
6.	Source Port Number	21, 23, 25,…	Numeric
7.	Destination Port Number	18162, 18161, …	Numeric

3.2 Preprocessing Data Feature Extraction

This is the phase in which irrelevant data are eliminated from the collection, such as data errors, irrelevant fields (i.e header len, time-to live, fragment offset and soon), insignificant information etc. it helps to transform the input features to produce new relevant features. Data needs to be processed because it could be noisy and inconsistent. In the dataset, some classes of data such as fragment offset and sequence number were not selected to be part of the mining process. This is because they do not provide any knowledge for the dataset processing also duplicate data are eliminated.

3.3 System Design

In this research work a hybridized classifier system was designed to achieve the aim of the work as shown in figure 1. The Decision Trees and Bayesian Network classifiers have been selected because of their performances in various domains. They have both been successfully deployed to a variety of real-world classification tasks in industry, business, science and education with good performances.

A decision tree is a model that comprises of a root node, branches, and leaf nodes. Each internal node signifies a test on a feature, each branch explains the result of a test, and each leaf node contains a class label. The topmost node in the

tree is the root node. The Decision Tree classifiers are considered "white box" classification model as they can provide explanation for their models and can be used directly for decision making. The Bayesian Network classifier uses the Bayes theorem to predict the class as the one that maximizes the posterior probability. The main task is to estimate the joint probability density function for each class, which is modeled via a multivariate normal distribution

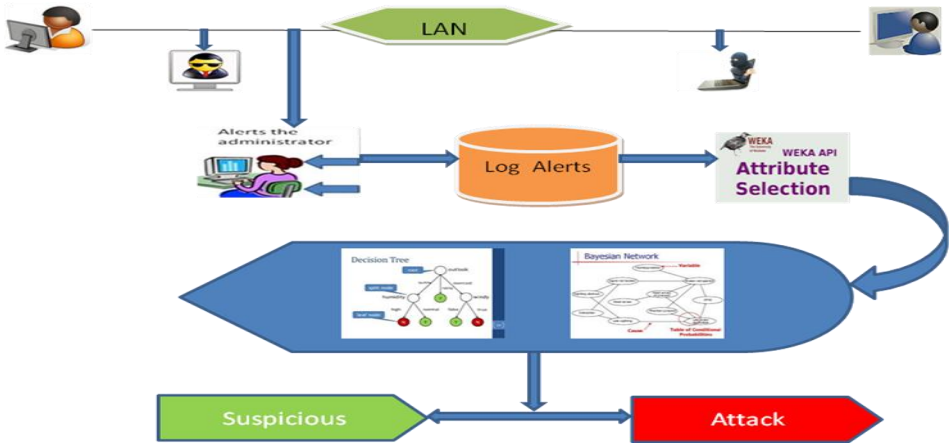

Figure 1: Methodology Framework

3.4 Snort Intrusion Detection System

In this research work, Snort IDS was deployed on the network in order to detect alerts which will be used in this research work. What snort does was that it logs the flow of packets in the alert log file in packet capture (pcap) format which was later organized into different attributes for further processing using data mining techniques.

Snort was put into operation under command prompt because snort cannot be put into operation as normal network software like wireshark.

The following codes were used to run Snort from command prompt interface:

1. >cd \snort.
2. >cd bin
3. >snort –V
4. >snort –W
5. >snort –i 5 –c c:\snort\etc\snort.conf –T
6. >snort –i 5 –c c:\snort\etc\snort.conf –A console

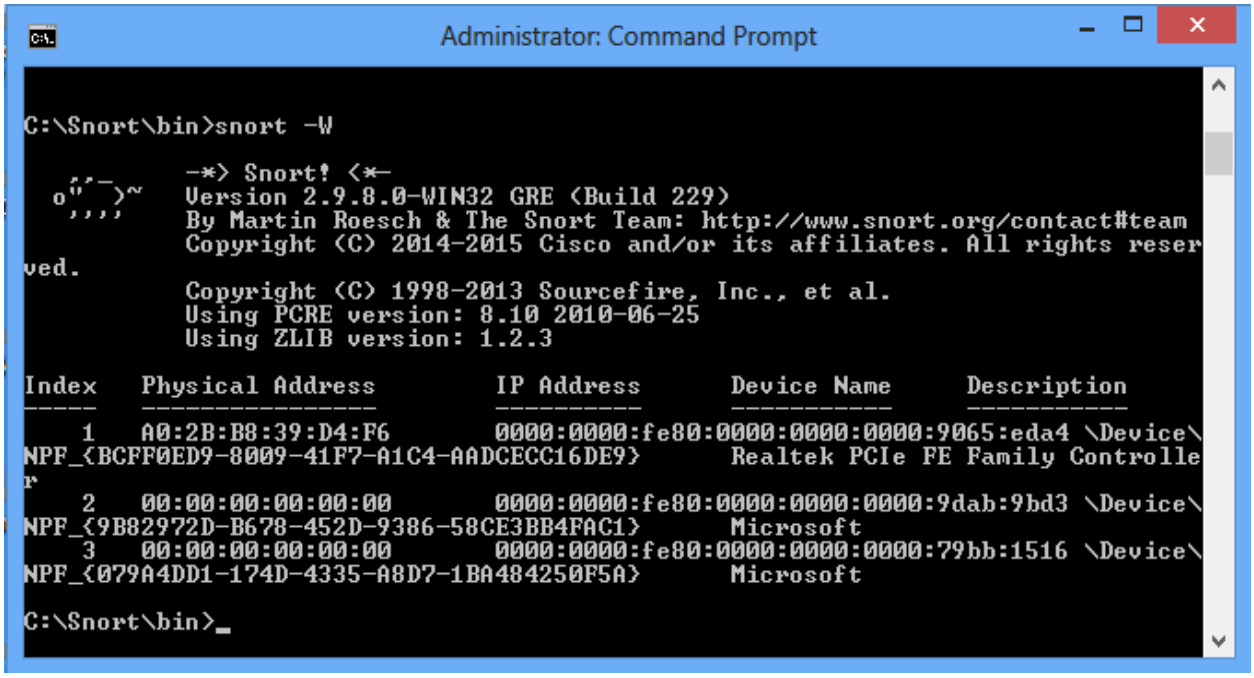

Figure 2: Snort Working Interface

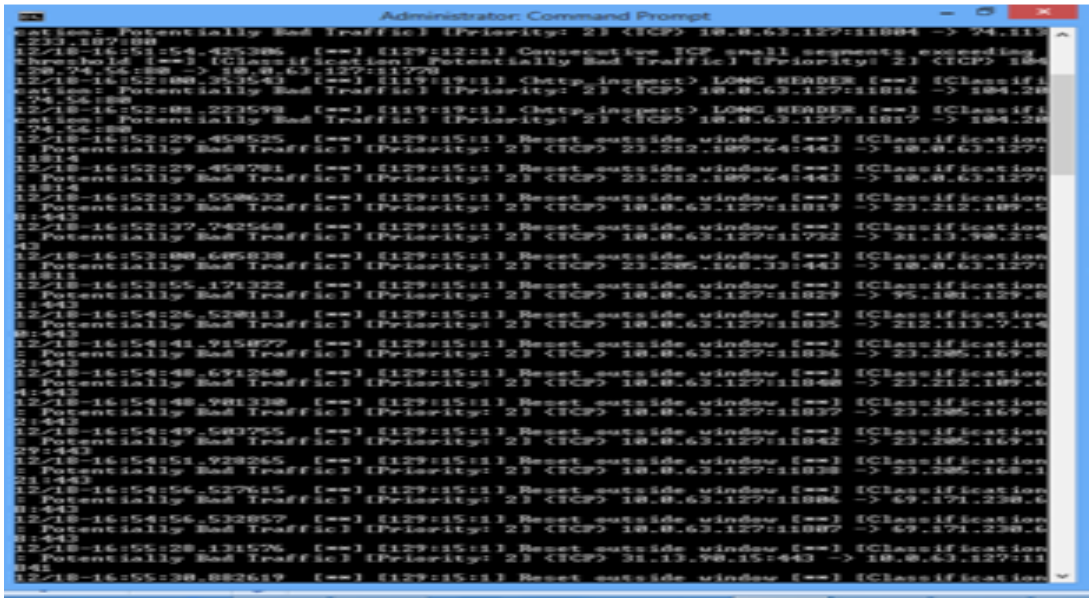

Figure 3: Network Packet Traffic

4. RESULTS AND DISCUSSION

The dataset consists of eight input variables and an output variable. The input variables are: Alert identification, Time stamp, Source IP Address, Destination IP Address, Protocols, Source port, destination port, Alert length. The output variable was "traffic pattern" which was assigned into the category of Attack traffic and Suspicious traffic

4.1 Training Dataset

The training dataset was used to enable the system to observe relationships between input data and predict the final outcomes. This allows the system to learn and develop a relationship between the input and the expected output. Sixty eight percent (68%) of the dataset was used for training making 956 instances.

4.2 Testing Dataset

The test dataset instances were then loaded into the system consisting of 320 instances; thirty two percent (32%) of records of malware datasets was used to predict their traffic pattern.

Table 2: Comparative Analysis on Testing Set

Metrics	**Value (NaiveBayes)**	**Value (Decision Tree)**
Time taken to build the model	0.02 seconds	0.04 seconds
Correctly Classified Instances	94.6875 %	96.875 %
Incorrectly Classified Instances	5.3125 %	3.125 %
Kappa statistic	0.8895	0.9336
Mean absolute error	0.0531	0.0501
Root mean squared error	0.2305	0.1583
Relative absolute error	11.3689 %	10.7253 %
Root relative squared error	47.69 %	32.7565 %
Total Number of Instances	320	320

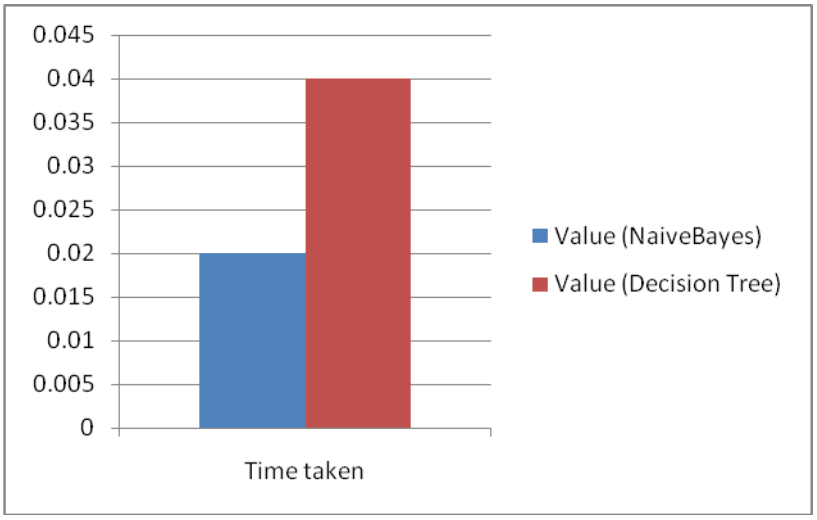

Figure 4: Time Graph Analysis: Comparing Naivebayes and Decision tree model

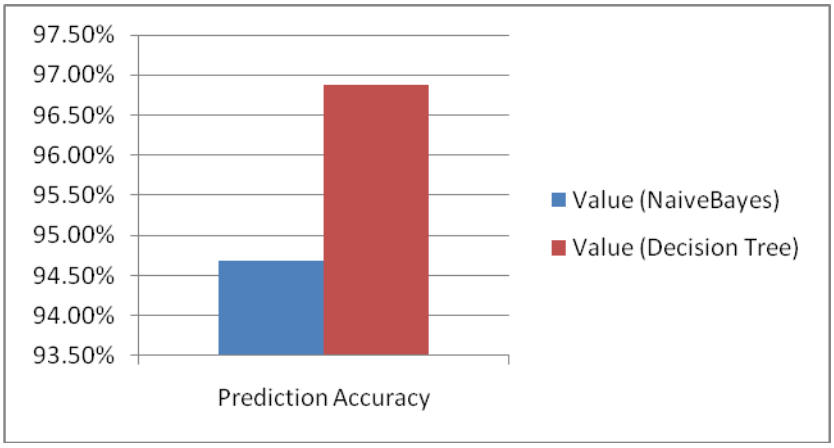

Figure 5: Prediction Accuracy Graph: Correctly Classified Instances

4.3 Discussion of Results

Results from the experiment in table 4.5a, naïve bayes is better in term of processing time to build its model than the decision tree because it took 0.12 seconds to process the dataset and it took 0.32 seconds for decision tree to complete its model; while decision tree does better in predicting the highest level of classifying accuracy. The main disadvantages of C4.5 classifier was that it took more CPU time and memory in execution. The disadvantage of naïve bayes model was that it has low classification accuracy.

5 CONCLUSION

Snort Intrusion Detection System Alerts was further re-categorized by decision tree and naïve bayes classifiers. From the result generated, decision tree classifier is better than naïve bayes classifier because decision tree classifier provides favorable characteristics such as high classification accuracy and low error metrics. It can be used in boosting classification performance and required in checking the network intrusion alerts. Snort Intrusion Detection System (SIDS) provides an abstract computing environment for data mining tasks, independent of the computer hardware and software on which it executes. Data mining techniques can incorporate the protocol directly and bring about an upgrade on how network traffics are logged. This work can be recommended for network analyst who will have great opportunity to check the logs, pattern of traffics and it will also reduce intruder from their deadly act on the network. In subsequent works, more records of network traffic can be worked on in order to obtain better generalization.

ENDNOTES

[1] Debar, H., Dacier, M. and Wespi, A. (2000): A revised taxonomy for intrusion detection systems, Annales des Telecommunications, 55, 7-8 (2000) 361-378.

[2] Axelsson, S. (2000): The Base-rate fallacy and the difficulty of intrusion detection, ACM Transactions on Information and System Security, vol 3, pp 186–205.

[3] W. Robertson and W. K. Robertson (2004): "Alert verification determining the success of intrusion attempts", The Proceedings of the Detection of Intrusions and Malware and Vulnerability Assessment, Dortmund, Germany, pp. 25-38.

[4] T. Chyssler, *et al* (2004): "Alarm Reduction and Correlation in Intrusion Detection Systems", Proceedings of the International Workshops on Enabling Technologies, Infrastructures for Collaborative Enterprises, pp. 229-234.

[5] T. Pietraszek (2004): "Using adaptive alert classification to reduce false positives in intrusion detection", The Proceedings of the symposium on Recent Advances in Intrusion Detection (RAID'04), Sophia Antipolis, France, pp. 102-124.

[6] Georgios P. Spathoulas and S. K. Katsikas, "Reducing False Positives in Intrusion Detection System", Computer and Security, vol. 29, no. 1, (2010), pp. 35-44.

[7] M.A.Shanti (2014): Application of Data Mining Using Snort rule for intrusion Detection.

[8] Faeiz Alserhani, *et al* (2009): "Evaluating Intrusion Detection Systems in High Speed Networks", In Press, Fifth International Conference of Information Assurance and Security (IAS 2009), IEEE Computer Society.

[9] Schultz, M., Eskin, E., Zadok, F. and Stolfo, S. (2001): Data Mining Methods for Detection of New Malicious Executables. Proceedings of 2001 IEEE Symposium on Security and Privacy, Oakland, 38-49.

[10] Kolter, J. and Maloof, M. (2004): Learning to Detect Malicious Executables in the Wild. Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 470-478.

[11] Kong, D. and Yan, G. (2013): Discriminant Malware Distance Learning on Structural Information for Automated Malware Classification. Proceedings of the ACM SIGMETRICS/International Conference on Measurement and Modeling of Computer Systems, 347-348.

[12] Osunade O., Adeyemo B. and Oriola, O. (2012): Network Threat Characterization in Multiple Intrusion Perspectives using Data Mining Technique, International Journal of Network Security and Its Applications (IJNSA), Vol.4, No.6.

[13] Siddiqui, M., Wang, M.C. and Lee, J. (2009): Detecting Internet Worms Using Data Mining Techniques. Journal of Systemics, Cybernetics and Informatics, 6, 48-53.

[14] Anderson, B. *et al* (2011): Graph Based Malware Detection Using Dynamic Analysis. Journal in Computer Virology, **7**, 247-258. http://dx.doi.org/10.1007/s11416-011-0152-x.

[15] Tian, R., Islam, M.R., Batten, L. and Versteeg, S. (2010): Differentiating Malware from Cleanwares Using Behavioral Analysis. Proceedings of 5th International Conference on Malicious and Unwanted Software (Malware), Nancy, 19-20.

[16] Lee, T. and Mody, J.J. (2006): Behavioral Classification. Proceedings of the European Institute for Computer Antivirus Research Conference (EICAR'06).

[17] Santos, I., *et al* (2013): A Static-Dynamic Approach for Machine Learning Based Malware Detection. Proceedings of International Conference CISIS'12-ICEUTE'12, Special Sessions Advances in Intelligent Systems and Computing, 189, 271-280.

[18] Raftopoulos E. and Dimitropoulos X (2010): Detecting, Validating and Characterizing Computer Infections in the Wild.

[19] Subbulakshmi T. *et al* (2010): Real Time Classification and Clustering of ids Alerts using Machine Learning Algorithms international journal of artificial intelligence and applications (ijaia), vol. 1, no.1.

[20] Romero C, Olmo JL, Ventura S (2013): A meta-learning approach for recommending a subset of white-box classification algorithms for Moodle datasets. Department of Computer Science, University of Cordoba, Spain.