International Journal of Cyber-Security and Digital Forensics (IJCSDF) 2(1): 25-35
The Society of Digital Information and Wireless Communications, 2013 (ISSN: 2305-0012)

AUTOMATED SIGNATURE CREATOR FOR A SIGNATURE BASED INTRUSION DETECTION SYSTEM WITH NETWORK ATTACK DETECTION CAPABILITIES (PANCAKES)

De Ocampo, Frances Bernadette C.
College of Computer Studies
De La Salle University – Manila
1004 Manila, Philippines (632) 524-4611 loc. 130
tetet_1992@yahoo.com

Del Castillo, Trisha Mari L.
College of Computer Studies
De La Salle University – Manila
1004 Manila, Philippines (632) 524-4611 loc. 130
trishamaridelcastillo@gmail.com

Gomez, Miguel Alberto N.
College of Computer Studies
De La Salle University – Manila
1004 Manila, Philippines(632) 524-4611 loc. 130
miguel.gomez.n@gmail.com

Abstract---A Signature-based Intrusion Detection System (IDS) helps in maintaining the integrity of data in a network controlled environment. Unfortunately, this type of IDS depends on predetermined intrusion patterns that are manually created. If the signature database of the Signature-based IDS is not updated, network attacks just pass through this type of IDS without being noticed. To avoid this, an Anomaly-based IDS is used in order to countercheck if a network traffic that is not detected by Signature-based IDS is a true malicious traffic or not. In doing so, the Anomaly-based IDS might come up with several numbers of logs containing numerous network attacks which could possibly be a false positive. This is the reason why the Anomaly-based IDS is not perfect, it would readily alarm the system that a network traffic is an attack just because it is not on its baseline. In order to resolve the problem between these two IDSs, the goal is to correlate data between the logs of the Anomaly-based IDS and the packet that has been captured in order to determine if a network traffic is really malicious or not. With the supervision of a security expert, the malicious network traffic would be verified as malicious. Using machine learning, the researchers can identify which algorithm is better than the other algorithms in classifying if a certain network traffic is really malicious. Upon doing so, the creation of signatures would follow by basing the automated creation of signatures from the detected malicious traffic.

Keywords---Anomaly-based Intrusion Detection System, Network-based Attacks, Feature Extraction, Network Attributes, and Machine Learning Algorithm

I. INTRODUCTION

Attacks in a system can greatly affect networks. Systems are needed today in monitoring and checking networks regularly[16].

An intrusion detection system (IDS) monitors network traffic, by alerting the system when a malicious traffic is detected. Anomaly-based IDS is a type of IDS. This type of IDS centers on the concept of a baseline for a network behavior.

Current Anomaly-based IDS that are available today face one main problem. At present, this type of IDS could highly produce false positives. False positive means any normal behavior that is identified as anomalous or malicious. People cannot fully depend on one IDS in identifying malicious traffic.

The solution to address the problem above is to use Machine Learning in order to filter more the network traffic. A Signature-based IDS is also used for more filtration process. It involves searching network traffic for a series of packet sequences known to be malicious. But Signature-based IDS depends on predetermined intrusion patterns that are manually created. If the signature database of the Signature-based IDS is not updated, network attacks just pass through this type of IDS without being noticed.

Having an Automated Signature Creator called Pancakes is the solution for the manual signature creation. Through this system, signatures will be created automatically, along with this the generated signatures would be fed onto a Signature- Based IDS named Snort. Before Pancakes can generate signatures a module called Log Attribute Selected Module is implemented and this paper focuses on this module only.

The paper is organized as follows: prevalence of network intrusion, intrusion detection system, system overview, system modules, machine

learning, methodology, preliminary results, conclusion and future work.

II. PREVALENCE OF NETWORK INTRUSION

Network-based attacks are continuously increasing nowadays. There are number of reasons why network attackers would want to attack networks. Some of which are: to use user accounts and escalate user privileges, to perform actions to deplete network resources and bandwidth or to simply enjoy the challenge of trying to compromise highly secured networks' security systems. Network-based attack consists of types such as self-propagating programs, spoofing, session hijacking and buffer overflow [2]. These types of attack can be some of the root causes of a damaged network. Nowadays, network hackers, methods and tools has improved tremendously, hackers no longer requires the full knowledge about hacking. In short, normal people can now hack small things with the help of hacking tools which can be easily installed in a computer. Because of these, a need for network security is very high in order to protect important data today.

III. INTRUSION DETECTION SYSTEM

As discussed above, an intrusion detection system (IDS) monitors network traffic by watching out for suspicious activities, and once suspicious activities has been detected it should now alert the system. An IDS has two detection techniques: Anomaly-based and Signature-based [7]. Both systems are similar with regards to their goal but they are different with regards to the detection technique it uses.

Signature-Based IDS run off a database of known malicious code and uses algorithms to compare packets to the database and highlight malicious packets. The packets captured by the IDS are compared against the signature database. The database needs to be constantly maintained and updated (just like an anti-virus program). A key advantage to this type of detection method is that signatures are easy to develop and to understand if you know what network behavior you are trying to identify. The main disadvantage of using a Signature-Based IDS is that the system is only as good as the database, meaning that if the database is not consistently maintained and updated it will not be able to detect the latest types of malicious packets coming into and from the network [4].

Anomaly-Based system detects an attack by creating a history database over time to create a baseline. It then compares incoming traffic to this baseline and looks at the behavior of the network to detect any anomalies and malicious attacks. The main advantage of an anomaly-based system is its ability to detect previously unknown (or variants of known) attacks when they appear and reappear [8].

IV. AUTOMATED SIGNATURE CREATOR FOR A SIGNATURE BASED INTRUSION DETECTION SYSTEM

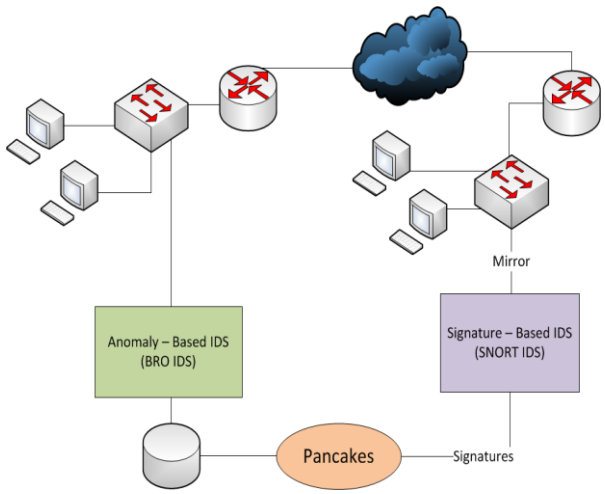

Figure 1 - Overview of the System

The Automated Signature Creator for a Signature-based Intrusion Detection System utilizes an Anomaly-based IDS (Bro IDS) in order to pre-identify malicious traffic coming from a network of virtual machines and/or from a data that is to be processed by the IDS. Logs would be the output of Bro IDS, this would be used by Pancakes in order to correlate the logs to the actual network capture which would help determine if a certain traffic that has pass through the network is an actual malicious traffic or not. All data coming from Bro IDS is stored in a database of log files.

Upon doing so, Pancakes would now know which network capture it is supposed to create signatures for because it has now identified if a certain network capture is malicious. Logs and the packet captures would be the basis of the signature generation because it contains each and every detail that a signature is supposed to have. Even if normal network traffic does not have an impact on

the study, it would still be used in order to create a baseline of malicious network traffic.

After extracting features from the network captures, Pancakes will now generate signatures that is to be passed on and fed to the Signature-based IDS (Snort IDS) that is used in the study. Snort bases its signature creation on non-payload and payload based rule options. As of the moment, the signatures that this study aims to generate are non-payload based detection rules. This would give an impact on the study not only thus it have a concrete number of rule formats and options to follow, thus it could also greatly affect the accuracy of the signatures generated because it is specific to form rules for a certain detection based.

A. Log Attribute Selection Module

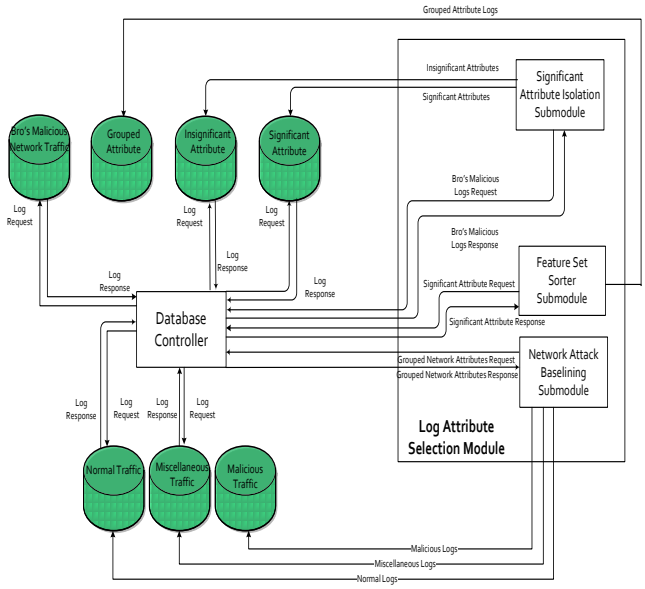

Figure 2 - Block Diagram of Log Attribute Selection Module

The Log Attribute Selection Module is composed of three Sub Modules such as: the Significant Attribute Isolation Sub Module, the Feature Set Sorter Sub Module, and the Network Attack Baselining Sub Module. The Signature Creation Module analyzes and tests the outputs of this module.

The Significant Attribute Isolation Submodule starts isolating packets coming from the database containing Bro's malicious logs. Upon doing so, it will identify significant network attributes that is needed in order to proceed to the Feature Set Sorter Submodule. The Feature Set Sorter Submodule is responsible for determining the compilation and grouping of the network attributes. All outputs rendered is to be received by the Network Attack Baselining Submodule, this submodule is apt to identify where each grouped network attribute is place whether it should be in the Normal Traffic, or Malicious Traffic using a machine learning algorithm.

B. Signature Creation Module

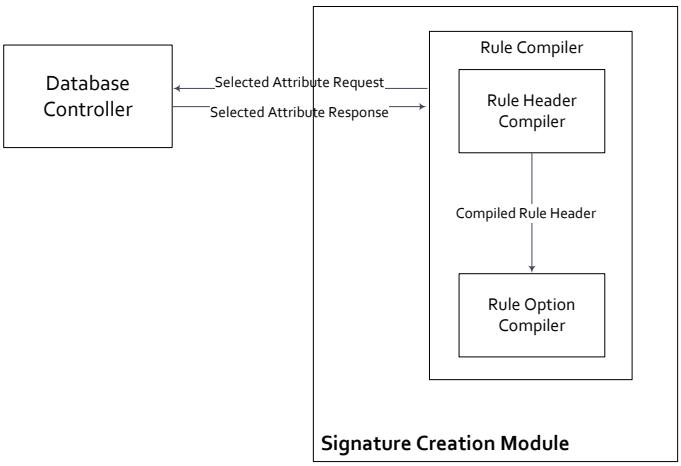

Figure 3 - Block Diagram of Signature Creation Module

The Signature Creation Module generates the rules that would be passed on to the Signature-based IDS, Snort. It will receive its data from the Log Attribute Selection Module. The selected network attributes originating from the previous module serves as the basis for the signature creation. A set of signatures will be generated in each distinct threat coming from a specific source that was identified.

V. MACHINE LEARNING

Machine learning refers to a system capable of acquiring and integrating knowledge from a known data, automatically. The capability of the systems to learn from experience, training, analytical observation, and other means, results in a system that can continuously self-improve and thereby exhibit efficiency and effectiveness [9]. Machine learning can either be supervised or unsupervised. A supervised learning system needs to be trained using already classified training data as opposed to an unsupervised system where such training is not done. The list below gives an overview of machine learning algorithms the researchers have used in the study along with their technical and mathematical definitions. In the later part of the paper the preliminary machine learning results of each algorithm is exhibited and explained.

A. Naïve-Bayes Classification Algorithm [11]

This algorithm is named after Thomas Bayes, who proposed the Bayes Theorem. It is a supervised machine learning algorithm, and it also gives practical learning algorithms and prior knowledge, and observed data can be combined. It formulates the probabilities for its hypothesis and it is also less affected by noise in the data. An advantage of the Naive Bayes algorithm is that it only requires a small amount of training data to estimate the parameters, means and variances of the variables, necessary for classification. Because independent variables are assumed, only the variances of the variables for each class need to be determined and not the entire covariance matrix.

Naïve Bayes algorithm was based on Bayes Theorem. Figure 4 shows the formula of the theorem.

$$P(T|E) = (P(E|T)XP(T))/(P(E|T)XP(T)+P(E|T)XP(T))$$

Figure 4 - Bayes Theorem

T stands for a theory or hypothesis that we are interested in testing. E represents a new piece of evidence that seems to confirm or disconfirm the theory. Bayes' Theorem is a theorem of probability theory. It is how the probability that a theory is true when affected by a new piece of evidence.

B. C4.5 Algorithm[3]

This algorithm is an extension of ID3 algorithm. It is used to generate a decision tree. ID3 algorithm uses the concept of entropy to build decision trees which C4.5 also uses. The training data is a set S=s1,s2,... of already classified samples. Each sample si=x1,x2,... is a vector where x1,x2,... represent attributes or features of the sample. The training data is augmented with a vector C=c1,c2,... where c1,c2,... represent the class to which each sample belongs. C4.5 selects one attribute of the data that most efficiently and successfully splits its set of samples into subsets enriched in one class or the other. The normalized information gain which is the difference in entropy is the standard that results from selecting an attribute for splitting the data.

C. Random Tree[15]

Random tree is a gathering of tree predictors. Random trees can learn more than one class at a time simply by collecting the class "votes" at the leaves of each of many trees and selecting the class receiving the maximum votes as the winner. In the case of regression, the classifier response is the average of the responses of all trees.

The algorithm has different training sets for each tree but contains same parameters. These sets are generated from the original training set using the bootstrap procedure: for each training set, you randomly select the same number of vectors as in the original set (=N). The vectors are chosen with replacement. Meaning, some vectors can occur more than once and some will be missing. At each node of each trained tree, not all the variables are used to find the best split, but a random subset of them. With each node a new subset is generated. However, its size is fixed for all the nodes and all the trees. It is a training parameter set to $\sqrt{number\_of\_variables}$.

D. Random Forest[14]

Random Forest is consists of many individual trees. In order to classify a new object from an input vector, put the input vector down each of the trees in the forest. Each tree gives a classification, and then the tree votes for that class. The forest chooses the classification having the most votes among all the trees in the forest.

This algorithm is sometimes called as a black box, a system who's internal process are invisible and defined only by its inputs and outputs. A series of predictor variables are input and the system delivers the output. Inside the black box system, the variables are associated and the outputs are generated.

A random forest must have a strong connection among all trees. The generalization error depends on the strength of each tree and the correlation between any trees in the forest.

E. Logistic Regression[13]

Logistic Regression is really a technique for classification. It is a type of predictive model that can be used when the target variable is a categorical variable with two categories. It doesn't involve decision trees like any other algorithms.

The logistic model formula, shows in Figure 5, computes the probability of the selected response as a function of the values of the predictor variables. The logistic formula has each continuous predictor variable, each dichotomous predictor variable with a value of 0 or 1, and a dummy variable for every category of predictor

variables with more than two categories less one category.

$$P = 1/(1+\exp(-(B0 + B1*X1 + B2*X2 + ... + Bk*Xk)))$$

Figure 5 - Logistic model formula

VI. METHODOLOGY

The process for gathering the feature set is listed below, this list gives a more detailed explanation on how the Log Attribute Selection Module of the system works.

A. Network Attack Generation

Network Attack Generation is the preliminary process that will issue the beginning of the automatic creation of signatures for the Signature-based IDS. These attacks are to be captured using a Packet Capture library.

The generated network attacks would then be saved in a packet capture file which will be the basis of the signature creation before it undergoes the different processes that the data will eventually go through. This process is responsible for creating the initial network traffic that the Anomaly-based IDS is going to examine.

B. Data Collection

Along with the generation of network attacks, the collection of data concerning network attacks can be collected through different sources over the Internet. These data are all saved as packet capture files similar as the Network Attack Generation Process.

C. Bro IDS Log Collection

Once data has been collected and generated, it will now be fed to the Anomaly-based IDS (Bro IDS). Feeding the packet captures files in the IDS will identify if the packet capture files that has been fed to it are legitimate attacks and are malicious network traffic.

In order to determine if the data are considered by Bro IDS as malicious, all of its reviewed data are stored in different log files. The most significant log files are the two: (a) weird.log, and (b) notice.log. These two files are significantly different in terms of the level of maliciousness of the network traffic that has been reviewed.

D. Packet Parser

After generating and collecting the packet capture files containing malicious network traffic, the said data is now deployed in the parser module of the system. This process is responsible for categorizing and organizing the attributes that are inside the packet capture files.

Depending on the packet type (i.e. ICMP, UDP, and TCP), the corresponding packet fields are supplied with data that has been extracted from the packet itself. This is implemented using the NetResearch's JPCAP API (created by Keita Fujii).

E. Logs and Packet Capture Correlation

The correlation of the Logs coming from Bro IDS and the data coming from the Packet Capture Files are needed to be mapped out and linked to each other because the Logs of Bro IDS contains minimal details regarding network attack that it has already detected. The signature generation could not be done by fully depending on the logs of Bro IDS thus; the packet capture files are still relevant and needed for the study.

The correlations of these data are being done by matching strings of data from one to another and/or by mapping out the time stamp on each data. In order to match the time stamp the capture time would be converted to epoch time because the logs of Bro IDS uses it as a basis for which data in the pcap file it is analyzing and it is treating as either normal or malicious network traffic.

F. Feature Extraction

Feature Extraction is the process wherein each network packet that has been linked to a log as malicious network traffic would be deconstructed and to be saved in a database and a CSV (Comma Separated Value) file which contains the data that would eventually be fed to the machine learning phase of the system. The table below shows the preliminary features that are to be extracted from the packet capture. The features will still be revised in order to fully satisfy the needs of the machine learning algorithm to be used.

Table 1 - Feature Set

Feature Name	Feature	Feature Description
cap_no	Packet Number	The packet's captured identification number.
epc_time	Epoch Time	The packet's epoch time
cap_length	Captured Length	The packet's captured length.
eth_frame_type	Frame Type	The Ethernet frame type of the packet.

eth_src_mac	Source MAC	The packet's source MAC address.
eth_dst_mac	Destination MAC	The packet's destination MAC address.
ip_version	Version	Identifies the version of the IP (IPV4/IPV6).
ip_tos_priority	TOS:Priority	Identifies the priority of the type of service.
ip_tos_throughput	TOS:Throughput	Identifies the throughput of the type of service.
ip_tos_reliability	TOS:Reliability	Identifies the reliability of the type of service.
ip_length	Length	The size of the IP packet header.
ip_identification	Identification	Identifies the type identification being used.
ip_don't_fragment	Don't Fragment	Identifies if the IP should not be fragmented.
ip_more_fragment	More Fragment	Identifies if the IP should be fragmented.
ip_fragment_offset	Fragment Offset	Identifies the fragment offset of the IP.
ip_ttl	Time to Live	Determines if the packet has been too long inside the network.
ip_protocol	Protocol	Identifies the protocol number for the IP.
ip_src_ip	Source IP	The source IP address of the packet.
ip_dst_ip	Destination IP	The destination IP address of the packet.
ip_src_hostname	Source host name	The source hostname of the packet.
ip_dst_hostname	Destination Host Name	The destination hostname of the packet.
arp_hrdwr_type	Hardware Type	Identifies the hardware type.
arp_protocol_type	Protocol Type	Identifies the protocol type.
arp_hrdwr_add_length	Hardware Address Length	The hardware address length.
arp_protocol_add_length	Protocol Address Length	Identifies the protocol address length.
arp_operation	Operation	Identifies the ARP operation.
arp_sender_hrdwr_add	Sender Hardware Address	Identifies the sender hardware address.
arp_sender_protocol_add	Sender Protocol Address	Identifies the sender protocol address.
arp_target_hrdwr_add	Target Hardware Address	Identifies the target hardware address.
arp_target_protocol_add	Target Protocol Address	Identifies the target protocol address.
udp_src_port	Source Port	The source port of the incoming packet.
udp_dst_port	Destination Port	The destination port of the incoming packet.
udp_packet_length	Packet Length	The size of the UDP packet.
tcp_src_port	Source Port	The source port of the packet.
tcp_dest_port	Destination Port	The destination port of the packet.
tcp_seq_number	Sequence Number	The packet's TCP sequence number.
tcp_ack_number	Ack Number	The TCP ACK number.
tcp_urg_flag	URG Flag	The TCP Urgent flag that inform the receiver of the packet that a segment within the data should be given priority.
tcp_ack_flag	ACK Flag	The TCP Acknowledgement flas is used to acknowledge the successful receipts of the packets.
tcp_psh_flag	PSH Flag	The TCP Push flag determines that the packet has been prioritized.
tcp_rst_flag	RST Flag	The TCP Reset flag indicates that the connection has been reset by the host computer.
tcp_syn_flag	SYN Flag	The TCP Synchronization flag is the initial value sent in establishing a handshake.
tcp_fin_flag	FIN Flag	The TCP Finished flag is used to indicate that the connection has been ended.
tcp_window_size	Window Size	The TCP Window Size indicates the maximum number of octets.
icmp_type	Type	The type of ICMP packet.
icmp_code	Code	The code that identifies the type of the message.
icmp_id	ID	The identification number that aids in the assembly of the packet.
icmp_seq	Sequence	The sequence number that aids in matching the reply of the packet.
class	Class	Identifies where the packet belongs to (Malicious or Normal Traffic).

The class feature is used to identify if a certain data is malicious or not. Its value could be normal or malicious. The epoch time is used in order to match the logs of Bro IDS and the packet capture itself.

G. Network Attribute Baselining

The Network Attribute Baselining process is responsible for the creation of a standard which could pre-identify if a network traffic that is being processed is either normal or malicious network traffic.

Machine Learning would be used in the study. It is useful in order to check which one of the algorithms to be tested by the system is better than the other algorithms in checking the maliciousness of network attacks. The selected algorithm is used for creating a baseline which could specify and differentiate the normal traffic from malicious network traffic.

The study focuses on supervised machine learning and uses classification in order to effectively identify which category the data belongs to. Here are some of the algorithms that would be used in order to create the baseline: (a) NaiveBayes Algorithm, (b) C4.5 Algorithm, (c) Random Tree, (d) Random Forest and (e) Logistic Regression.

H. Signature Generation

Like what was written earlier, the system generates the rules that would be passed on to the Signature-based IDS, Snort. It will obtain its data coming from the machine learning process. The data that will be received was already filtered as malicious instances, basically separated from the normal and miscellaneous ones.

VII. PREVIOUS MACHINE LEARNING RESULTS

Table 2 - Amount of Initial Data

Class	Number of Instances
Malicious	764
Normal	736

The table above shows the initial amount of data collected for the machine learning phase of the system. The data will still increase and it will also be revised in order to satisfy the needs of the machine learning algorithm. Satisfying the suitable features for each algorithm would result to having a more accurate, more precise, and an unbiased result set. Listed below are the initial results of various machine learning algorithms that Weka has. The data is classified as either Malicious or Normal.

In order to identify which algorithm performs well using the given data set, it is a must to remember that when the TP Rate and Kappa Statistic are closer to 1 this means that the algorithm is promising. For Mean Absolute Error and FP Rate if the data is closer to 0 then the algorithm is better.

The letters in the table stands for each algorithm that has been tested in the study: (A)NaiveBayes, (B) C4.5, (C) Random Tree, (D) Random Forest, and (E) Logistic Regression.

Table 3 - Machine Learning Initial Results

	A	B	C	D	E
Correctly Classified Instances	95.9333 %	99.9333 %	88.1333 %	99.0667 %	98.1333 %
Incorrectly Classified Instances	4.0667 %	0.0667 %	11.8667 %	0.9333 %	1.8667 %
Kappa Statistic	0.9187	0.9987	0.7621	0.9813	0.9627
Mean Absolute Error	0.0428	0.0007	0.1228	0.1234	0.0191
TP Rate	0.959	0.999	0.881	0.991	0.981
FP Rate	0.04	0.001	0.121	0.009	0.018

The Kappa Statistic represents the measurement which determines the percentage of the chance agreement. If the Kappa Statistic is close to 1 then it indicates that the agreement is perfect, if the Kappa Statistic is close to 0 then the agreement is just by chance. In Table 3, the chances of all the algorithms being perfect are high because all the values are higher than 0.9 making it close to 1, meaning for each algorithm that was used the algorithm was not guessing on the classifications of the data.

Based on table 3, Algorithm (A) NaiveBayes was able to correctly classify 95.9333 % out of the initial data which is a total of 1500 instances. (C) Random Tree's output has the lowest rate of it being suited for the study because the results of the other algorithms are way higher. But, still the result of (B) C4.5 and (D) Random Forest has the closest relationship, having only a difference of .8666% basing on the results of the Correctly Classified Instances and a difference of -.8666% basing on the Incorrectly Classified Instances. Algorithm (B) C4.5 Algorithm exhibited the most favorable results amongst the other algorithms used in the study. It appears to have the most strategic way of classifying the data compared to all the algorithms that is presented here in the study (A), (C), (D), and (E).

The True Positive (TP) Rate indicates the rate that the predicted label of the data is highly correct proving that the data has been labeled correctly. The False Positive (FP) Rate indicates the rate that a predicted label of the data is wrong because the data has been labeled incorrectly. In this case, a normal traffic could be labeled as malicious and that could represent the FP Rate of the Machine Learning results. This means that the data that was used was able to correctly classify most of the malicious class from the normal class. The correctly classified instances of the data could be used in the succeeding processes of the system. The FP Rate outcome is relatively low meaning that the some of the data were incorrectly classified.

Algorithm (C), as mentioned earlier, it has the lowest rate in the study. Next to it is the algorithm (A), followed by (E), and lastly, (B) and (D) which have the closest relationship.

Upon determining the highest and the lowest scores of grades inside the table above, the most effective machine learning algorithm could be deduced using the data above. As seen in Table 3, (B)'s Kappa Statistic and TP Rate is closer to 1, this means that the algorithm favors the data given. The most ineffective machine learning algorithm for the study is (A) NaïveBayes.

VIII. NEW MACHING LEARNING RESULTS

The machine learning process uses training set and testing set. A training set is a set of data or instances in order to discover potentially predictive relationships. A training set should be as close as possible to the actual data that a user expects to see. On the other hand, the testing set is a separated data that also contains instances with the same attributes as the training data. The classification only works if the training set has the same number and name of attributes as the testing set. It is crucial to check the training set and the testing set every time. The only difference between these two sets is that the class labels are randomly assigned in each instance in the testing set.

The instances are classified by two main classes namely, malicious and normal. The training set is consists of a total of 1692 number of instances, 846 classified as malicious and 846 as normal. While the testing set is consists of a total of 535 number of instances, 292 classified as malicious and 243 as normal. The labeling for the testing data is done randomly.

The initial classes that are going to be considered in this study are the following:

Table 4 – Initial Classes with Description

Class	*Description*
Normal	A network traffic that does not recognized as malicious by Bro IDS.
Malicious	A network traffic that is recognized as malicious by Bro IDS.

A. Machine learning without feature selection

The system has to test various machine learning algorithms in order to identify which one of the algorithms to be tested by the system is better than the other algorithms in identifying the maliciousness of network attacks. Shown in Table 5 are the correctly classified instances rates in each algorithm. All of them were used in order to classify instances.

In order to identify which algorithm performs well using the given data set, it is a must to remember that when the TP Rate and Kappa Statistic are closer to 1 this means that the algorithm is promising. For Mean Absolute Error and FP Rate if the data is closer to 0 then the algorithm is better.

The letters in the table stands for each algorithm that has been tested in the study: (A)NaiveBayes, (B) C4.5, (C) Random Tree, and (D) Random Forest.

Table 5 – Classification Results

	A	***B***	***C***	***D***
Correctly Classified Instances	77.0093 %	72.7103 %	93.4579 %	84.8598 %
Incorrectly Classified Instances	22.9907 %	27.2897 %	6.5421 %	15.1402 %
Kappa Statistic	0.5204	0.417	0.8676	0.6856
Mean Absolute Error	0.2257	0.2729	0.0943	0.1651
TP Rate	0.77	0.727	0.935	0.849
FP Rate	0.266	0.333	0.069	0.18

The Kappa Statistic represents the measurement which determines the percentage of the chance agreement. If the Kappa Statistic is close to 1 then it indicates that the agreement is perfect, if the Kappa Statistic is close to 0 then the agreement is just by chance. In Table 5, the chances of (C) being close to perfect are high because the Kappa value of 0.8676 and the Correctly Classified Instances of 93.4579 are higher than the others.

The True Positive (TP) Rate indicates the rate that the predicted label of the data is highly correct proving that the data has been labeled correctly. The False Positive (FP) Rate indicates the rate that a predicted label of the data is wrong because the data has been labeled incorrectly. In this case, a normal traffic could be labeled as malicious and that could represent the FP Rate of the Machine Learning results. This means that the data that was used was able to correctly classify most of the malicious class from the normal class. The

correctly classified instances of the data could be used in the succeeding processes of the system. The FP Rate outcome is relatively low meaning that the some of the data were incorrectly classified.

The table shows that algorithm (C) still gives the highest result when it comes to TP. Others ranges from 0.70 – 0.89. Upon determining the highest and the lowest scores of grades inside the table above, the most effective machine learning algorithm could be deduced using the data above. As seen in Table 5, algorithm (C) is the best algorithm to use in classifying the maliciousness of every instance. Therefore, the system used Random Tree Algorithm.

B. Machine learning with feature selection

The system had to test different types of Feature Selection methods and evaluators available in WEKA GUI. During testing, the proponents decided to choose the "Ranker" search method in order for them to easily identify the most beneficial attributes based on their ranking of individual evaluations. The Ranker search method consists of different types of attribute evaluators such as ReliefF, GainRatio, InfoGrain and etc.. The proponents use four attribute evaluators for the feature selection process namely, Information Gain, Gain Ratio, Symmetrical Uncertainty and Relief Attribute.

In order to choose the desirable algorithm among the four, the proponents are required to test and classify the training data and testing data with the reduced features. Like what was written earlier, the attributes that were selected as important attributes are based on the ranking of each evaluator tested.

At this point, each feature selection evaluator is needed to test with each machine learning algorithm used in the study (C4.5, NaiveBayes, Random Tree and Random Forest). In result, the Relief Attribute Evaluator gives the best accuracy result with a percentage of 81.8692% for the Random Tree algorithm. The table below shows the classification results using the reduced features with the machine learning algorithm.

The letters in the table stands for each algorithm that has been tested in the study: (A)NaiveBayes, (B) C4.5, (C) Random Tree, and (D) Random Forest.

Table 6 – Classification Results with Reduced Features

	A	***B***	***C***	***D***
Correctly Classified Instances	77.0093 %	72.7103 %	81.8692 %	78.6916 %
Incorrectly Classified Instances	22.9907 %	27.2897 %	18.1308 %	21.3084 %
Kappa Statistic	0.5204	0.417	0.6377	0.5518
Mean Absolute Error	0.2257	0.2729	0.172	0.2143
TP Rate	0.77	0.727	0.819	0.787
FP Rate	0.266	0.333	0.174	0.257

If the Kappa Statistic is close to 1 then it indicates that the agreement is perfect, if the Kappa Statistic is close to 0 then the agreement is just by chance. In Table 6, the chances of (C) to be the best algorithm are very high because its Kappa Statistic is the highest among all the other algorithms tested.

The table also shows that algorithm (C) still gives the highest result when it comes to accuracy for the reason that the Correctly Classified Instances of (C) is the highest. To conclude, algorithm (C) is the best algorithm to use in classifying the maliciousness of every instance using reduced features.

To compare with the test earlier using the complete and untrimmed attributes, they both resulted to have Random Tree as their most desirable algorithm. The proponents used the machine learning without feature selection because Random Tree gives a higher accuracy there compared with the algorithm with feature selection.

C. Identifying three classes: malicious, normal, and miscellaneous

As mentioned earlier, the initial data only contains two classes: malicious and normal. Because of this, the proponents need to know first how they can identify miscellaneous instances. In order to

achieve this, the study uses a function called distributionForInstance(Instance instance).

The final classes that are going to be considered in this study are the following:

Table 7 – Final Classes with Description

Class	***Description***
Normal	A network traffic that does not recognized as malicious by Bro IDS.
Malicious	A network traffic that is recognized as malicious by Bro IDS.
Miscellaneous Activity	A network traffic that can hardly compromise networks or computers.

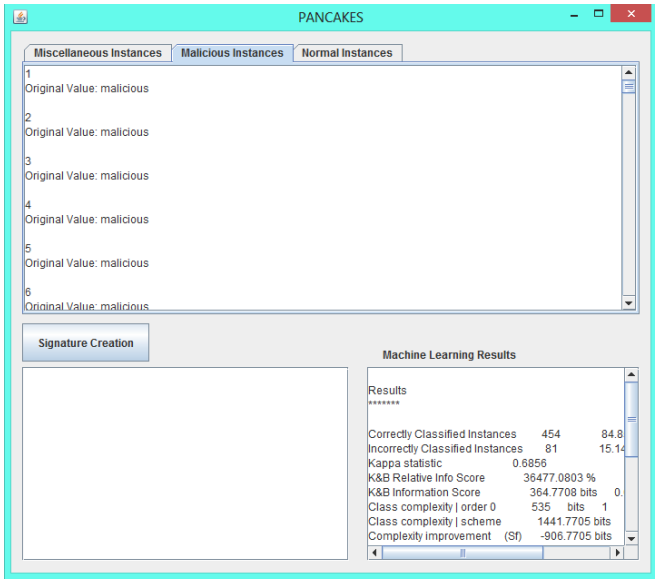

Figure 6 – Malicious Instances

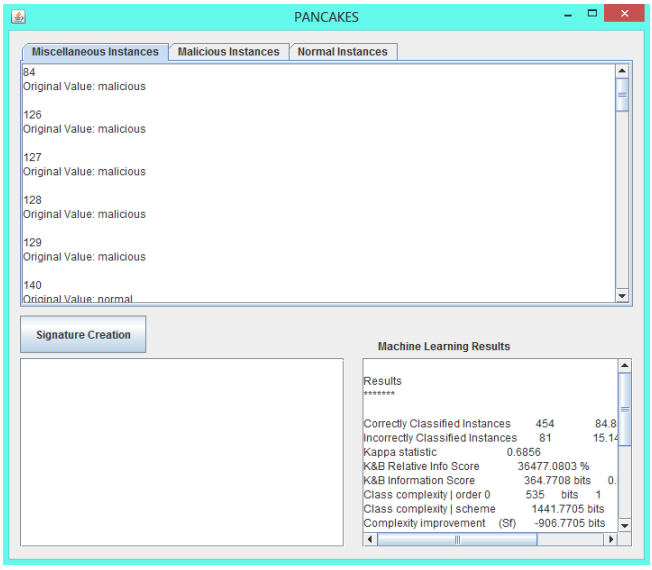

Figure 7 – Miscellaneous Instances

In this test, the proponents used Random Tree Algorithm because we found out earlier that it gives the best results among all algorithms tested.

Figure 6 shows the list of malicious instances outputted based on the distribution of classes a while ago. Basically, the numbers on the screen indicate the instance number of the malicious traffic. The one labeled as "Original Value" is the original class of a specific instance. As the user scrolls down the results, he/she can see some results of "normal" being the original value of an instance. On the other hand, Figure 7 shows the results for the miscellaneous instances. It shows how the proponents filter out the malicious ones from the miscellaneous and normal instances. With this, the malicious instances will now be passed on to the signature generation process.

IX. SIGNATURE GENERATION

In this test, the system is able to create network based attack signatures.

Figure 8 – Signatures Generated by PANCAKES

Once the user pressed the "Signature Creation" button, the signatures generated are outputted in the text area of Pancake's graphical user interface. The proponents also saved the data produced in a file named pancakes.rules located in /etc/snort/rules (for Ubuntu) and C:\Snort\rules (in Windows) file path. Figure 9 shown below is the screenshot of the file. The data shows the actual data being used by the Snort IDS. These data are called the signatures or rules. These rules are simple to write, yet powerful enough to detect a wide variety of hostile or merely suspicious network traffic.

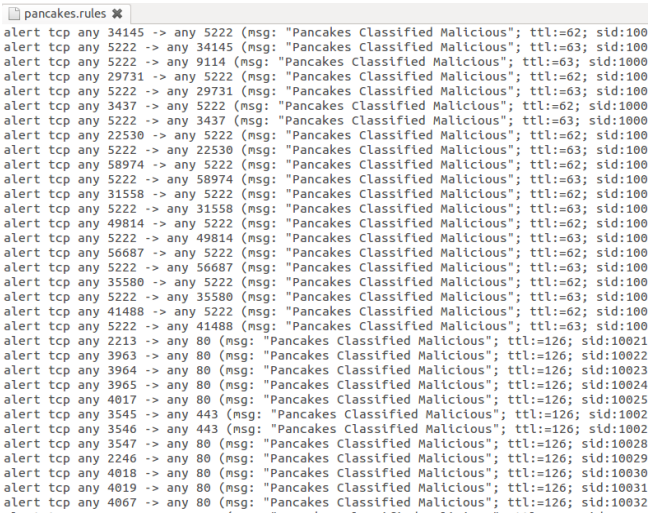

Figure 9 – Generated Signatures Outputted in pancakes.rules

X. CONCLUSION

Based on the machine learning results, the analysis shows that Random Tree Algorithm is the most efficient algorithm because it has exhibited the highest percentage for Correctly Classified Instances. The study proved that automation of signatures is possible with the use of machine learning. Through machine learning, the system was able to further determine the malicious instances within a capture file that is already pre-analyzed by an Anomaly Based IDS. In addition to this, the system was able to predetermine instances that are classified to be network attacks.

XI. RECOMMENDATION

To further improve the system, thorough research with regards to newly developed attacks is needed in order to fully implement an efficient system which could greatly affect the signature-based IDS. At the end of the day, any type of signature-based IDS is dependent on the signatures that are created by its users, founders, and researchers. Aside from this, categorization of network attacks should be in mind, this will help in further determining proper ways to mitigate threats are emerging and further developing.

REFERENCES

[1] B. Karp, J. Newsome and D. Song, "Polygraph: automatically generating signatures for polymorphic worms," Internet: http://www.ece.cmu.edu/~dawnsong/papers/polygraph.pdf, n.d. [Feb,2012].

[2] G. Berg and S. Goel, "Security threats: network-based attacks," Internet: http://www.cs.albany.edu/~berg/risk_analysis/Lectures/Security_Threats.pdf, n.d. [Feb,2012].

[3] G. Melli, "C4.5 algorithm," Internet: http://www.gabormelli.com/RKB/C4.5_Algorithm, n.d. [July,2012].

[4] H. Tom, "Differences between signature and anomaly based ids," Internet: http://www.stmxupa.com/hon/index.php/component/content/article/115?format=pdf, Feb. 16, 2011 [Nov,2012].

[5] J. Crowcrof and C. Kreibich, "Honeycomb – creating intrusion detection signatures using honeypots," Internet: http://nms.lcs.mit.edu/HotNets-II/papers/honeycomb.pdf, n.d. [Feb,2012].

[6] M. Tanase, "The great IDS debate : signature analysis versus protocol analysis," Internet: http://www.symantec.com/connect/articles/great-ids-debate-signature-analysis-versus-protocol-analysis, Nov. 2, 2010 [Feb,2012].

[7] M. W. Tzeyoung "Intrusion detection systems," Internet: http://iac.dtic.mil/iatac/download/intrusion_detection.pdf, Sept. 25, 2009 [Feb,2012].

[8] N. Gustavo and C. Miguel, "Anomaly-based intrusion detection in software services," Internet: http://wraits11.di.fc.ul.pt/papers/nascimento-webintdetect.pdf, n.d. [Nov,2012].

[9] Robin, "Machine learning overview," Internet: http://intelligence.worldofcomputing.net/machine-learning/machine-learning-overview.html, Mar 1, 2010 [Nov,2012].

[10] "A tutorial on clustering algorithms," Internet: http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/kmeans.html, n.d. [July,2012].

[11] "Bayes' Theorem," Internet: http://www.trinity.edu/cbrown/bayesweb/, n.d., [Nov,2012].

[12] "K-Means clustering algorithm," Internet: http://mathworld.wolfram.com/K-MeansClusteringAlgorithm.html, n.d. [July,2012].

[13] "Logistic regression," Internet: http://www.dtreg.com/logistic.htm, n.d. [Nov,2012].

[14] "Random forests leo breiman and adele cutler," Internet: http://www.stat.berkeley.edu/users/breiman/RandomForests/cc_home.htm, n.d. [Nov,2012].

[15] "Random trees," Internet: http://docs.opencv.org/modules/ml/doc/random_trees.html, n.d. [Nov,2012].

[16] G. Meera and S. K. Srivatsa. "Detecting and preventing attacks using network intrusion detection systems" *International Journal of Computer Science and Security*,vol. 2, pp. 49-60, n.d.