International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:7, No:11, 2013

Evaluating Performance of an Anomaly Detection Module with Artificial Neural Network Implementation

Edward Guillén, Jhordany Rodriguez, Rafael Páez

***Abstract*—**Anomaly detection techniques have been focused on two main components: data extraction and selection and the second one is the analysis performed over the obtained data. The goal of this paper is to analyze the influence that each of these components has over the system performance by evaluating detection over network scenarios with different setups. The independent variables are as follows: the number of system inputs, the way the inputs are codified and the complexity of the analysis techniques. For the analysis, some approaches of artificial neural networks are implemented with different number of layers. The obtained results show the influence that each of these variables has in the system performance.

***Keywords*—**Network Intrusion Detection, Machine learning, Artificial Neural Network.

I. Introduction

ANOMALY detection addresses the problem of detecting threats which have not been seen before. To achieve that, the most common approach is to define a profile as normal within a knowledge database and then, all the deviations from that profile are further analyzed looking for intrusions [1].

The implementation of an Anomaly Detection Module –ADM requires establishing some initial parameters such as: data extraction and selection, analysis technique, and how the information is represented for the analysis [1]. The purpose of this work is to analyze how different setups affect the detection's performance to finally show which components are more influential in order to obtain better anomaly detection modules.

An ADM can be seen as a classification system with the goal of classifying each event as normal or abnormal. For that reason, the same figures of merit of a classification system such as accuracy, true positives, and false positives, among others, can be applied to an ADM [2]. In addition to the figures of merit, the time is also a very important evaluation point for an ADM because the detection must be performed in near real time [3], [4]. Furthermore, the networks' capacity is continuously growing and the number of elements to be evaluated per second is increasingly large.

This work was supported in part by Military University Nueva Granada, the Colciencias' researchers program and the Doctoral Program granted to E.G. author in Pontifical Xaverian University.

E. Guillen and J. Rodriguez are with the Military University Nueva Granada, Cra 11 No. 101-80, Bogota, Colombia (e-mail: edward.guillen@unimilitar.edu.co, jhordany.rodriguez@ieee.org).

R. Paez is with the Pontifical Xaverian University, Bogota, Colombia (e-mail: paez-r@javeriana.edu.co).

This work analyzes and develops all the phases for non-content based attacks detection with the implementation of an anomaly detection module.

The first step in the implementation is the creation of the profile to be considered as normal. The profiles are obtained based on information about processes, events and traffic. The acquired information depends on the ADM's physical location [5]. In this case, the module is designed to work as a Network Intrusion Detection System –NIDS. The available data for a NIDS is related to the data extracted from network packets and network flows [5]. The collected information is analyzed and organized in a dataset in order to define different profiles.

ADM performance is affected by the quality of the obtained dataset. These datasets should describe the network behavior as good as possible. For example, a very tight profile will cause high false positives rate because any slight deviation will be considered as abnormal behavior. Furthermore, some detection approaches have been implemented with defined datasets, but the use of a general dataset to evaluate specific environments may not be the best option because the definition of normal behavior differs from network to network [6]-[8]. On the other hand, some of existing datasets have not been updated for a long time. The intrusions are changing continuously and their tracks must be obtained from updated sources [6]-[8]. The tool *Spleen* was used to get information from real and specific environments as well as to obtain additional traffic characteristics [7], [8].

Among the analysis techniques, Artificial Neural Networks –ANN have been widely used due to their capacity for pattern recognition [10]. In this work, some implementations are developed with different complexity levels in order to establish how different analysis levels affect the detection performance. However, the complexity of the analysis does not only depend on the ANN configuration but also in the number of used inputs and their representation, for that reason, different data selection processes and different data representations are used in this work to evaluate how different configurations affect the system performance.

The process developed in this work is the creation of anomaly detection modules with different configurations where the independent variables are the data used for the analysis, the encoding scheme and the complexity of the Analysis approach. With the obtained results, an analysis is performed in order to evaluate the importance of each component in the performance in terms of accuracy and processing time. The paper is organized as follows: The

second section gives information about previous related works. 3rd Section describes some methods and resources. The description and analysis of the experiments are given in Section IV. Finally, some conclusions can be read in Section V.

II. Related Work

In order to evaluate different approaches for anomaly detection some datasets have been created [5], [6], [9], [11]-[14]. These datasets allow performing different tests over the same information and evaluate the performance of analysis approaches.

Among the developed datasets, some of them have received special attention [6]. The most analyzed is the KDD99 developed by MIT [5]. It has been widely used in order to evaluate anomaly detection modules. This dataset collects information about network connections and extracts 41 traffic features; these features refer to individual network events as well as statistical information. Among all the features related with a connection, the status flag –SF of the connection plays a very important role since almost all the statistical measures are based on it. SF is obtained from the three hand shake process [1], [5], [7], [8], [15].

However, some authors have shown several weaknesses in the mentioned dataset [6]-[10]. The outdated of the database is one of the most important limitations. So, it makes the network description difficult because the current networks, services, applications, users as well as the current intruders are different than those 10 years ago.

Furthermore, new intrusion and evasion techniques have been developed by intruders in order to avoid being detected by the detection systems [16]-[18]. Attackers avoid controls by causing high variations on the commonly examined features to make its detection more difficult. For example, some new scan techniques performed by Nmap use some modifications in the TCP flags in order to get information about the victim systems by fingerprints recognition [17]. This technique takes advantage of the fact that not all the operating systems accomplish the RFC recommendations and they respond in different ways to the same situation. Such techniques allow the intruders to perform their scanning tasks without causing significant variations among the characteristics evaluated in the KDD99. However, the use of these techniques usually causes variations in other traffic features. In [7] and [9] new set of new features in order to take advantage of these changes for intrusion detection were introduced.

Other important point in the construction of a dataset is the way the traffic is labeled [6], [9]. Traffic labeling is a process in which each element is labeled according to a previous knowledge about its nature. It is important in supervised training because the analysis technique can query the labels in order to evaluate its own performance and modify its internal parameters if it is needed [6], [7]. There is another training method called unsupervised method [10]. This method tries to find relationships between the samples in order to create clusters. Therefore, to have labels for each single element is not required for unsupervised training. In [19] a method which clusters http requests for botnet detection is proposed.

Constructing datasets based on network level information has another critical point, the encrypted messages. These encrypted messages, beside the heterogeneous data about users and applications information makes the inclusion of content based features difficult [20]. For that reason the datasets created have shown low performance for content based attacks detection [20]. The present work focuses in non-content based attacks detection because is based on network level detection.

In addition to the dataset construction, some authors had developed feature selection processes in order to reduce redundancy and improve the processing time of the analysis techniques [21], [22]. Those processes are based on statistical analysis. One of the most used methods is the information gain based on Shannon theory [21]. Information gain uses a labeled dataset to calculate the contribution of each feature for the classification.

Once the data for the analysis is ready, the analysis technique must be selected and implemented. Many approaches have been proposed for this purpose including genetic algorithms [7], particle swarm [23], negative selection [24], artificial neural networks [10], and support vector machines [25], among others. However, since the anomaly detection problem can be seen as a classification process in which the system learns from experience, the ANN have been widely implemented with high-quality results [26].

However, most of the analysis techniques only work with numerical data, so, the discrete features, such as SF, must be represented as numerical data. The representation of discrete features can be addressed with different approaches. In [1] a comparative analysis is performed over different methods for discrete features representation: indicator variables, conditional probabilities and separability split value. Authors applied different techniques to an intrusion detection problem to finally conclude that the system performance is strongly influenced by the encoding technique.

III. Methodology and Resources

In this section, the tools and theories employed for the development of the experiments are explained.

A. Collecting Data

The first step is the creation of the profile which will be considered as normal. This profile is created based on information obtained from real scenarios with *Spleen* [7], [8]. Spleen obtains the information about non-content based features defined in the KDD99 and in order to improve the traffic description, new features were included as described in [7], [8].

The first 41 features are identified by the IDs from KDD99 [5] and the new features starts from number 42. In addition to the traditional types, a new type called “behavior change detector” –BCD [7] were introduced. The BCD features measure the difference between the number of events which match some condition in a time slot and the number of

events which satisfy the same condition in the previous time slot. The inclusion of BCD features helps in the detection of abrupt changes in the network traffic. However, since even the normal behavior in a network is not homogeneous over time, the conditions used in BCD features are based on events which require previous conditions, for example a connection rejection.

Datasets were created as follows: The traffic was captured in a mirror port of a LAN segment into a real environment and the attacks were generated using vulnerability and configuration assessment products as Nessus and Nmap [17], [18].

B. Features Selection

The analysis was done with a PCA method [27]. The goal of PCA is to obtain a set of not correlated factors which are linear combinations of the initial variables. In this way, the same information where each sample has p attributes can be represented using k attributes ($k < p$) while conserving the maximum amount of information about the original data. In other words, the goal is to obtain a basis Y which is a linear combination of the original basis (X) that re-expresses the dataset in a best way [27]. This is done by applying a linear transformation according to (1) where P corresponds to the principal components.

In order to make Y to re-expresses the data in a better way, it is necessary to reduce the redundancy in Y. To achieve this, the concept of covariance is employed. The covariance indicates the degree of linear relationship between two variables [27]. The covariance of a matrix Y can be expressed according to (2) where n is the total number of samples, each column corresponds to a sample -i.e. a connection- and each row represents all measurements of a particular type - i.e. a feature-. The matrix obtained with equation 2 is a square matrix of mxm where m is the number of features. The ideal covariance matrix of Y (C_y) is a diagonalized matrix, that is, all the off-diagonal elements must be close to zero. In other words, the correlation and redundancy between variables must be minimized. However, Y is unknown, and the intention is to find the best P (1). To address this issue, (3) is obtained by re expressing Y as PX [27]. Based on the fact that C_y must be a diagonalized matrix and C_x can be easily calculated by equation X, the new question is, what is the best choice of P?. According to [27], applying the rule expressed in equation W, a symmetric matrix is diagonalized by a matrix of its eigenvectors where D is a diagonal matrix and E is a matrix of eigenvectors of A arranged as columns, theorem 3 and 4 of [27]. A good choice of P is a matrix where each row corresponds to an eigenvector of C_x. An eigenvector of a matrix X is a vector which satisfies the equation 4 where λ corresponds to an eigenvalue of X and I_n is the identity matrix. An eigenvalue is a scalar which satisfies the condition expressed in (5), that is, the determinant of that matrix is equal to 0.

$$Y = PX \tag{1}$$

$$Cy = \frac{1}{n} YY^T \tag{2}$$

$$Cy = PCxP^T \tag{3}$$

$$(\lambda I_n - X)\vec{V} = \vec{0} \tag{4}$$

$$|\lambda I_n - X| = 0 \tag{5}$$

Each principal component is a vector which indicates a direction into the variable space. This vector follows the direction of the bigger variations in the dataset. The direction of the vector depends on the features. For example, if in a bidimensional space (X,Y), a principal component vector matches the axis X, it means that the variable located at the axis X is the one which contribute the most in the construction of that principal component, then it is the most relevant one [27]. In order to select the most relevant features, the most representative variables of the 10 principal components with the highest variability were selected.

C. Analysis Technique

The Analysis was performed with some neural network approaches.

1) Neural Network Overview

Neural networks are bio-inspired methods where the architecture of the brain is emulated in a very basic way [28]. In the brain the neurons establish connections between them. These connections establish a kind of network where some stimuli work as inputs, the information from these inputs passes through some neurons and their connections generating an answer. When these answers are good, the connections between the involved neurons become stronger or weaker in order to generate similar answers when similar inputs are provided. In this way, the neural networks can be used for pattern recognition. This process, known as learning from experience process, is the behavior the artificial neural networks try to emulate [28].

The basic neuron model defines the neurons as simple entities which receive some data and apply an activation function to it, in order to generate an output. The basic neuronal network model defines the network as a set of neurons that are connected each other's with links. These links alter the value of the inputs before they are applied to the neurons by computing their values with a weight. These weights represent the strength of the connections between neurons. In artificial neural networks (ANN) the data travel over the network in a specific order even when the network uses parallel processing [28]. Some neurons work with the answers generated by others neurons. The groups of neurons work in a "parallel" way because their inputs do not depend on their partner outputs. An ANN should have at least two layers: an input layer and an output layer. Additionally, an ANN can have one or more hidden layers. Fig. 1 (a) provides a graphical description about layers; each layer is drawn with a different shade. To compute their outputs the neurons use their activation functions. An activation function can be understood

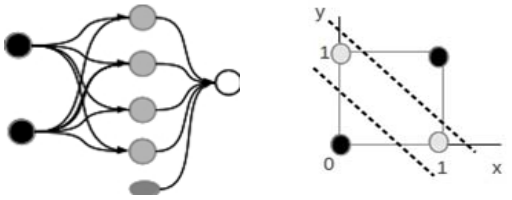
as an equation where the independent variable is the weighted sum of its inputs and the result is the output of the neuron.

The purpose of the learning process is to find the correct weights to determine the strength of the neurons connections such the proper answers are generated for the given inputs [28]. The most extended supervised training methods are based on back propagation techniques. These methods are composed by two basic phases [26]. In the first phase the outputs are calculated based on the current inputs. Then the error is obtained by comparing the result with the expected output. In the second phase the error is used to begin a back-propagation process. Depends on the training method, the weights are changed based on the direction or magnitude of the gradient [28]. To calculate this gradient the current output is applied to the derivative of the activation function, for that reason, the activation function must be derivable. Generally, the higher error the more change is applied to the current weights. This process is executed many times until some stop condition is satisfied, usually a fixed number of iterations or a desirable error value.

2) Neural Network Parameters

The main attributes of an ANN are: the number of layers, training method, the number of inputs and outputs, the activation function used in its neurons, and the way the inputs and outputs are represented. The choice of these parameters affects the performance of the network. For example, at the beginning of artificial intelligence, some researchers were discouraged in the use of ANN due to some limitations shown in the simple perceptron by Minsky and Papert [29]. This primitive ANN approach did not have hidden layers. Due to its nature, it was not able to properly separate data with complex distributions. A simple example of this limitation can be explained with the XOR problem shown in Fig. 1 (b) with X and Y as inputs.

The darker dots in Fig. 1 represent the answer 1 and the lighter points represent the answer 0. As it can be seen, it is not possible to separate the darker points and the lighter ones with just a single line. Two lines are required to separate them. The more layers the ANN has, the more lines can be used to separate the points. ANN recovered its attention around 1974 when Werbos added new features to the neural network model [30].



(a) (b)

Fig. 1 (a) ANN with 3 Layers (b) The XOR problem

In the same way, the training methods evolved from the basic Hebb rule [31] to relatively complex back propagation algorithms which are able to find the optimal weights in a faster way avoiding local minimums [28].

D. Encoding Scheme

The encoding scheme is referred in this paper as the way the discrete features are represented. The encoding schemes used in the experiments are equal distance intervals and indicator variables [1].

The only discrete input is the status flag of the connection which can assume the values S0, S1, S2, S3, REJ, RSTO, RSTR, SF among others [1], [7], [8]. In order to apply equal distance intervals a value was assigned to each possible state. These values were obtained by dividing the range 0 to 1.0 into the number of common states, and then each value was mapped to a status flag. The lower values were assigned to the first possible states according to the three hand shake process, in this way the values represent the progress in the connection establishment.

The method based on indicator variables [1] uses a variable to represent each one of the states. If the feature can assume 4 different values then 4 variables are required to represent that feature. Each indicator variable can only assume two values: 1 when the variable assumes its corresponding value and 0 otherwise.

IV. Experiment Setup

A. Experiment Description

Different configurations were implemented using different data, encoding scheme and analysis complexity levels.

Table I shows the different points which are considered for the experiments. The experiments were made using combinations of those configurable points. The total number of experiments is 48 (3x4x4).

The expected outputs are obtained from the nature of the events used for the dataset construction; in this case, the events were divided in 4 categories: normal, transport probe attacks, IP probes and DoS attempts.

The feature selection process was performed using the PCA technique. The features considered as the most relevant ones are: 34, 29, 48, 47, 33, 44, 31, 24, 46 and 26; the status flag was not considered by PCA method, but it is included due to its previously explained importance. As can be seen, the features 5 and 6 -source and destination bytes- did not get a very high score in the PCA method that is because the redundancy between them is relatively high.

TABLE I
EXPERIMENT ENCODING SCHEME

Selected data	Encoding scheme	Complexity level (ANN)
All features	Both inputs and outputs with intervals method.	3 layers
Selected features from old dataset	Inputs with intervals method and outputs with indicator variables.	4 layers
Selected features from both old and new dataset.	Inputs with indicator variables and outputs with intervals method	6 layers

Results include three of the new features in the 10 most relevant ones. It confirms that the inclusion of new features can help in classification processes. In order to perform the experiments which use only selected features from old dataset, the features 44, 46 and 48 were replaced for the most relevant features among the rest: 30, 35 and 37.

Each experiment was performed 1000 times in order to obtain the results: the experiment with the highest detection rate was used to obtain the accuracy measures; the processing time corresponds to the average time used by the experiments with each configuration.

In a real world implementation, the module and the data collector must work together. In that case, the analysis is performed each time a connection changes its status flag.

V. ANALYSIS AND RESULTS

The figures of merit evaluated were: Detection Rate (DR), True Positive Rate (TPR), False Positive Rate (FPR) and Processing Time.

Fig. 2 (a) shows the results obtained using different sets of features. As it can be seen, during the training phase the DR and TPR were similar. The lowest detection rate was obtained when all features were used during the testing phase. However, the main difference is observed in the processing time. This time was almost the double when all the features were used. Since the IDS should work in real time it must be as fast as possible to avoid packet dropping.

Fig. 2 (b) shows the results obtained when the discrete features where represented with different techniques (Equidistant intervals and indicator variables). In this case the best performance was achieved by represented the inputs and outputs with indicator variables. The most important discrete feature used in these experiments is the status flag of the connection with 16 different values. It causes the size of the inputs vector be 15 units bigger *(n-1)* than its size when equidistant intervals were used. As a disadvantage, using indicator variables increases the processing time as shown in Table II. Finally, Fig. 2 (c) shows the results obtained using different number of layers in the ANN configuration. In this case the detection rate does not improve as well as expected with the use of 8 layers, especially in testing phase.

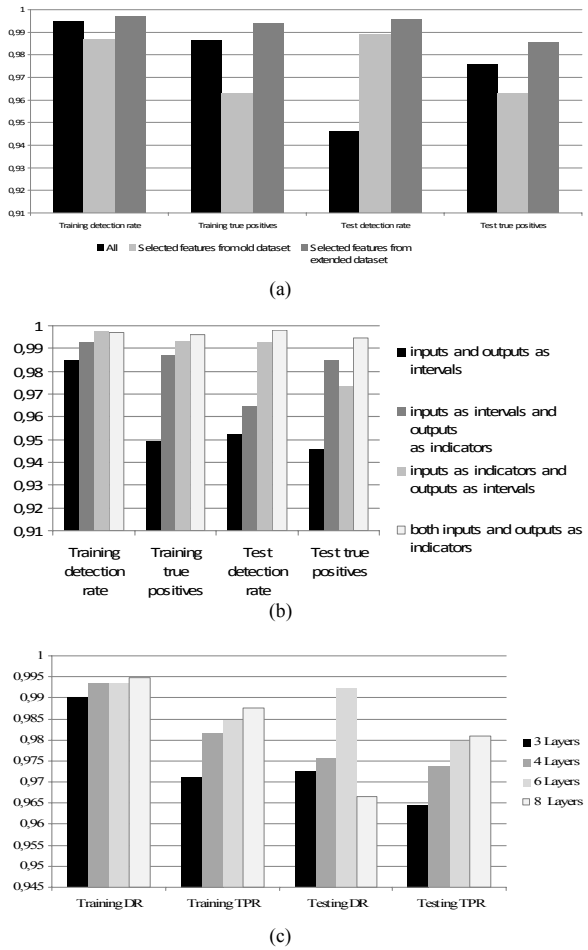


Fig. 2 (a) Figures of merit for different data selections; (b) Figures of merit using different representations for the discrete features; (c) Figures of merit using different complexity neural network approaches

The best detection rates were achieved when the 10 most relevant features were used and the inputs and outputs were represented by indicator variables with 6 layers. On the other hand, the lowest performance for detection rate was obtained when all the features were used. The training phase was not so bad, but when the testing phase was performed the false positive rate was very high compared with the other configurations. Because the FPR was high, the detection rate was down as it can be seen in Fig. 2 (b). It means that the obtained ANN does not have a good generalization capacity. It was difficult to decide the stop condition for the trainings because it would not be fare to train the ANN with 3 layers with the same iterations that the ANN with 8 layers. The

continuous output(s) of the network were applied to a comparative algorithm to determine the final discrete output. Although this algorithm was not considered by the training process the error training was used as the stop condition. Therefore the trainings were over when some convergence was achieved.

TABLE II
AVERAGE PROCESSING TIME

Settings	Time (ns)
All the features	*15687,67736*
Selected features from old dataset	*7460,03916*
Selected features from new dataset	*7543,58891*
Inputs and outputs as intervals	*6859,910875*
Inputs as intervals and outputs as indicators	*8042,89987*
Inputs as indicators and output as intervals	*12120,49942*
Both inputs and outputs as indicators	*13898,4304*
3 layers	*3913,67971*
4 layers	*5791,321724*
6 layers	*12188,63611*
8 layers	*19028,10302*

VI. CONCLUSION AND FUTURE WORK

More complex ML configurations does not necessary imply better results. It is recommended to evaluate the detection methods with different complexity levels. It is possible that the effectiveness does not improve after some complexity level. This critical point is very important because continuing adding complexity to the detection systems can increase the processing time causing non-desirable consequences as packets dropping.

The main differences in the performance were obtained using different attributes with different representations. It is a signal that selecting the correct attributes and their representation has a relevant role in the intrusion detection systems, probably, its relevance is bigger than the relevance of the ML technique employed. Furthermore, the way the attributes are extracted from the traffic can make the system faster and effective.

This work is focused on the detection of non-content based attacks and it is possible that the detection of content based attacks requires more complex ML unlike the results shown in this work. Because the correct extraction and selection of attributes are very important, the next step is the implementation of strategies for content based attacks analysis.

Both figures of merit, effectiveness and processing time, are important. The decision about how the anomaly detection module is implemented can depend on the device to perform the task. If a very robust machine is available, it is possible to use the approach with the higher effectiveness.

ACKNOWLEDGMENT

E. Guillen thanks to Military University for the labs used in getting the feature information with GISSIC investigation group. The work was possible in part with the support of Colciencias research program.

REFERENCES

[1] Hernández-Pereira, E.; Suárez-Romero, J.; Fontenla-Romero, O. & Alonso-Betanzos, A. Conversion methods for symbolic features: A comparison applied to an intrusion detection problem Expert Systems with Applications, 2009, 36, 10612 - 10617
[2] Edward Guillen, Yudy Alexandra Colorado, Daniel Padilla. Weaknesses and Strenghts Analysis over Network-based Intrusion Detection and Prevention Systems. 2009. LATINCOM ’09. IEEE Latin-American conference on. 1-5
[3] Artan, N.S. and Ghosh, R. and Yanchuan Guo and Chao, H.J. A 10-Gbps High-Speed Single-Chip Network Intrusion Detection and Prevention System. Global Telecommunications Conference, 2007. GLOBECOM '07. IEEE pages 343-348,
[4] Cheng Xu and Fei Yu and Zhenghui Dai and Guangxue Yue and Renfa Li. Data Distribution Algorithm of High-Speed Intrusion Detection system Based on Network Processor. Semantics, Knowledge and Grid, 2006. SKG '06. Second International Conference on.
[5] Haines J, Lippmann R, Fried D.J, Zissman M, Tran E, Boswell S. 1999 DARPA intrusion detection evaluation: Design and procedures. Technical report 1062. Massachusetts institute of Technology. Lincoln Laboratory.
[6] Ali Shiravi Hadi Shiravi, M. T. A. A. G. towards Developing a Systematic Approach To Generate Benchmark Datasets for Intrusion Detection 2011, 357-374.
[7] Guillen Edward, Rodríguez Jhordany and Paez Rafael. Improving Network Intrusion Detection with Extended KDD. IAENG Transactions on Engineering Technologies. 2013
[8] Edward Guillen, Jhordany Rodríguez, R. P. A. R. Detection of Non-Content Based Attacks Using GA with Extended KDD Features World Congress in Engineering and Computer Science, 2012.
[9] Vasudevan, A.; Harshini, E. & Selvakumar, S. SSENet-2011: A Network Intrusion Detection System dataset and its comparison with KDD CUP 99 dataset Internet (AH-ICI), 2011 Second Asian Himalayas International Conference on, 2011, 1 -5
[10] Shun, J. & Malki, H. Network Intrusion Detection System Using Neural Networks Natural Computation, 2008. ICNC '08. Fourth International Conference on, 2008, 5, 242 -246
[11] Lawrence Berkeley National Laboratory and ICSI, . LBNL/ICSI Enterprise Tracing Project. www.icir.org/enterprise-tracing/
[12] CAIDA, 2011. The Cooperative Association for Internet Data Analysis. http://www.caida.org/
[13] RTI International, 2011. PREDICT:Protected Repository for the Defense of Infrastructure Against Cyber Threats. http://www.predict.org/
[14] The Shmoo Group, 2011. Defcon. http://cctf.shmoo.com/
[15] Information science institute, University of southern California. RFC 793: Transmission control protocol. September 1981.
[16] Marpaung, J.; Sain, M. & Lee, H.-J. Survey on malware evasion techniques: State of the art and challenges Advanced Communication Technology (ICACT), 2012 14th International Conference on, 2012, 744 -749
[17] Nmap Port Scanning Techniques:Nmap Reference Guide 2012. http://nmap.org/book/man-port-scanning-techniques.html
[18] Nessus Vulnerability Scanner documentation
[19] Gu Guofei; Perdisci Roberto; Zhangh Junjie; Lee Wenke; BotMiner: Clustering Analysis of Network Traffic for Protocol- and Structure-Independent Botnet Detection. Georgia Institute of Technology, Damballa Inc Atlanta. May 2008.
[20] Sabhnani, M. & Serpen, G. Why Machine Learning Algorithms Fail in Misuse Detection on KDD Intrusion Detection Data Set The University of Toledo, October 2003
[21] H. Güneş Kayacık, A. Nur Zincir-Heywood, M. I. H. Selecting Features for Intrusion Detection: A Feature Relevance Analysis on KDD 99 Intrusion Detection Datasets Dalhousie University, Faculty of Computer Science.
[22] El-Khatib, K. Impact of Feature Reduction on the Efficiency of Wireless Intrusion Detection Systems Parallel and Distributed Systems, IEEE Transactions on, 2010, 21, 1143 -1149
[23] Dozier, G.; Brown, D.; Hurley, J. & Cain, K. Vulnerability analysis of AIS-based intrusion detection systems via genetic and particle swarm red teams. Evolutionary Computation, 2004. CEC2004. Congress on, 2004, 1, 111 - 116 Vol.1
[24] Forrest, S.; Perelson, A.S.; Allen, L.; Cherukuri, R. Self-nonself discrimination in a computer Research in Security and Privacy, 1994. Proceedings. 1994 doi: 10.1109/RISP.1994.296580. IEEE Computer

Society Symposium on , vol.no., pp.202-212, 16-18 May 1994, 1994, pp.202-212, 202-212
[25] Kim, D. S.; Nguyen, H.-N. & Park, J. S. Genetic algorithm to improve SVM based network intrusion detection system Advanced Information Networking and Applications, 2005. AINA 2005. 19th International Conference on, 2005, 2, 155 - 158 vol.2
[26] Devaraju, S.; Ramakrishnan, S., "Performance analysis of intrusion detection system using various neural network classifiers," Recent Trends in Information Technology (ICRTIT), 2011 International Conference on , vol., no., pp.1033,1038, 3-5 June 2011.
[27] Shlens Jonathon. A tutorial on Principal Component Analysis. Center for neural Science, New York University. April 2009.
[28] Jeff Heaton. Introduction to Neural Networks for C#, 2 edition. Heaton Research, October 2008. ISBN: 1604390093
[29] Marvin Minsky and Seymour Papert, 1972 (2nd edition with corrections, first edition 1969) Perceptrons: An Introduction to Computational Geometry, The MIT Press, Cambridge MA, ISBN 0-262-63022-2.
[30] Werbos, P.J. (1975). Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences. Ph.D Thesis Cambridge MA Hardvard University
[31] D.O Hebb. The organization of behavior. New York. Wiley, 1949. Introduction and chapter 4 “The first stage of perception; grow of the assembly”.