Journal of Theoretical and Applied Information Technology
31st January 2015. Vol.71 No.3
© 2005 - 2015 JATIT & LLS. All rights reserved·

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

IMPLEMENTATION OF AN INTRUSION DETECTION SYSTEM BASED ON SELF ORGANIZING MAP

[1]EMIRO DE LA HOZ FRANCO, [2]ANDRES ORTIZ GARCIA
[3]JULIO ORTEGA LOPERA, [4]EDUARDO DE LA HOZ CORREA
[5]FABIO MENDOZA PALECHOR
[1]Full Time Professor, Systems Engineering Program, Universidad de la Costa, CUC, Colombia
[2]Professor Hired Doctor, Communications Engineering Department, University of Malaga, Spain
[3]Full Professor, Computer Architecture and Technology Department, CITIC, University of Granada, Spain
[4]Full Time Professor, Systems Engineering Program, Universidad de la Costa, CUC, Colombia
[5]Full Time Professor, Systems Engineering Program, Universidad de la Costa, CUC, Colombia
E-mail: [1]edelahoz@cuc.edu.co, [2]aortiz@ic.uma.es, [3]jortega@ugr.es, [4]edelahoz6@cuc.edu.co, [5]fmendoza1@cuc.edu.co

ABSTRACT

The main purpose of this study is to identify a methodology to validate the effectiveness of an Intrusion Detection Systems proposed in three phases (selection, training and classification) using FDR to feature selection and Self Organizing Maps to training-classification. Therefore, initially are covered basics introductory in the first four items, related to the input dataset, the intrusion detection system and the metrics that are necessary to evaluate the IDS, the feature extraction technique FDR and the funcionality about the self-organizing map (SOM). Later in the methodology Item, in the body of the paper, a functional model proposed to described the intrusion detection, such model is validated from the comparation of metrics in simulation develops enviroments. Finally concluded that the detection rates obtained by the proposed functional model are: sensitivity of 97.39% (fits correctly identified as attacks) and a specificity of 62.73% (normal traffic correctly identified as normal traffic) using only 17 features of the dataset input. These results are compared with other simulating scenarios different, consulted from the documentary sources, from which it is suggested to integrate at the proposed model other techniques for training and classification processes to optimize the intrusion detection model.

Keywords: *Intrusion Detection System – IDS, Self-Organizing Map – SOM, Fisher's Discriminant Rate – FDR, Gaussian Mixture Model (GMM), dataset NSL-KDD*

1. INTRODUCTION

The Techniques detect malicious traffic, used in the prevention of attacks from various data sources, such as firewalls, data encryption algorithms, virtual private networks (VPN) and Intrusion Detection Systems (IDS) are not fully effective. The Firewalls restrict traffic from unknown services by blocking ports; however the attackers sometimes encapsulated attacks within data packets corresponding to the firewall enabled services. Moreover firewalls, encryption algorithms and virtual private networks are techniques commonly used to prevent intrusions from remote connections and do not affect the classification of the network local traffic. Moreover, existing commercial IDS, despite intrusive prevent traffic from both inside and outside of corporate computer networks, are based on signatures, which implies the non-detection of unknown attacks and the need to be constantly updating the database. A feasible solution is the implementation of an IDS with a methodology based on anomaly detection, using a machine learning algorithm, capable of detecting attacks which possess no prior knowledge. Ask a solution of this kind involves generating simulation contexts that take a dataset of network connections to subject it to scrutiny.

In this research was used the NSL-KDD dataset, which can be downloaded from [1]. It contains network traffic with a diverse collection of compiled links. The dataset contains 8 files, 4 in “txt” format and the other in “arff” format (Attribute Relation File Format).

The files "KDDTrain+_20Percent.txt" and "KDDTest+.txt", which contain a distinct collection of connections, from the NSL-KDD dataset, were used respectively for training of SOM and the classification of connections phases. The files in question are constituted by network connections records occupying 100 bytes each, and each record in turn consists of 41 characteristics describing the connection. Use all the features to classify connections in normal traffic and intrusion generates an unnecessary computational burden, since all are not relevant for this task. Therefore by feature extraction technique FDR, are prioritized and extract the optimal number of features to evaluate. This number is used to train a neural network using the machine learning algorithm SOM content. Network traffic file "KDDTest+.txt" is then classified using the map already trained and finally the level of performance of the implemented classifier is determined, assessing the sensitivity and specificity metrics.

The initial purpose of this study is to lay the foundation for future commercial implementation of an intrusion detection system in network, to classify the normal and abnormal traffic, unsupervised way, eliminating the need for regular updating of database attacks. To date no commercial IDSs have implemented that make use of unsupervised intrusion detection techniques (or based on anomalies). This research has taken as reference some background on commercial deployment of IDSs [2-5] and in relation to the use of artificial intelligence techniques in intrusion detection [6-7].

2. INTRUSION DETECTION SYSTEMS

IDSs, standardized based on [8-11], are data protection tools that greatly complement the use of other security techniques. IDSs are classified according to the criteria of approach or analysis type, data source or sources of information, according to its structure and its response or behavior [12-13]. Regarding the approach, IDSs are subclassified into Misuse-based Intrusion Detection and Anomaly-based Intrusion Detection, in Figure 1, the functional architecture shown, see [14-15].

Misuse-based Intrusion Detection monitors the activities that occur in a system and compares them with a database of predefined attack signatures, generating an alert in case the activity is identified as an attack. According to [6] this technique is widely used in commercial products because of its predictability and precision, however for the method to be effective it is necessary to keep updated the database of signatures, the deficiency of this method that it´s not identifies new nonexistent attacks in the database.

Anomaly-based Intrusion Detection works assuming that the attacks are different to normal activity, you can reach this inference after a training process, which will be identified, "what is considered normal activity?", analyzing unusual behavior in both host and network traffic. This generated profiles are constructed from the analysis of association patterns, these profiles represent normal behavior of users, hosts or network connections. Furthermore, an attack is represented by one or more connections, considering that each connection consists of 41 attributes. They can identify if the connection is normal or if some kind of attack. Attributes can be of two types: continuous or numeric (take real or integer values) and discrete or symbolic (take values from a specified list).

To perform the intrusion detection requires evaluating different attributes that make up the respective connections that refer to attacks. To determine the efficiency of the classifier must consider the metric sensitivity and specificity, which are statistical measures of the performance of a binary classification test. The first, also called “recall rate” measures the proportion of "true positives", a "sensitivity" of 100% means that the test recognizes that all attacks were actually detected as attacks, see equation (1). The second measures the proportion of "true negatives" 100% "specificity" indicates that all the normal traffic is properly identified as such, see equation (2). To determine the sensitivity and specificity metrics is necessary to evaluate the following:

- True Positive (TP): attack correctly identified as attack.
- False Positive (FP): normal traffic incorrectly identified as attack.
- True Negative (TN): normal traffic correctly identified as normal traffic.
- False Negative (FN): attack incorrectly identified as normal traffic.

From the foregoing,

$$\text{sensitivity} = \mathrm{TP}/(\mathrm{TP} + \mathrm{FN}) \qquad (1)$$

$$\text{specificity} = \mathrm{TN}/(\mathrm{TN} + \mathrm{FP}) \qquad (2)$$

The feature extraction technique FDR was adopted as a mechanism for identifying the optimal number of features to be extracted from the dataset.

3. FISHER DISCRIMINANT RATE - FDR

The feature extraction process simplifies the amount of resources needed to accurately describe a large set of data. Fisher Discriminant Ratio defined in [16-18], find the optimal transformation matrix preserving most of the information that can be used to discriminate between the different categories. Therefore, the analysis requires that data are category labels, to mathematically formulate the optimization or reduction process. This method requires calculated the mean vector and the covariance matrix for each class and for the complete data set (with all cluster classes). From this, we can formulate the optimization criterion. In equation (3) FDR, we see that the numerator represents the covariance of the training data grouped in the transformed features space and the denominator represents the average covariance within each class in the transformed features space.

$$\mathrm{FDR} = \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2} \qquad (3)$$

Where u_1, u_2 are the mean and σ_1^2, σ_2^2 are the variances of “y” in the classes w_1, w_2 respectively, after the screening along “w”.

After performing the feature extraction process applies the machine learning algorithm from the Self- Organizing Map (SOM).

4. SELF-ORGANIZING MAP (SOM)

SOM defined in [19-20], enable the representation of multidimensional data in much smaller spaces, typically 1, 2 or 3 dimensions. The most prominent feature is that the SOM learns to classify data using an algorithm for unsupervised learning. SOM consists of a network of nodes, each of which has a specific topological position with coordinates (x, y) in the lattice and contains a weight vector of the same dimension as the input vectors.

SOM training occurs in several steps and many iterations:

1. Each node is initialized with its own vector of weights, they will be set to small random values standardized.
2. A vector is chosen randomly among the set of training data and presented to the network (input vector).
3. Each node is examined to calculate the weight which is more "close" to the input vector. The winning node is commonly known as the Best Matching Unit (BMU).
4. The radius of the neighborhood of the BMU is calculated, which is a large value initially associated with the size of the network, and decreases with the passage of time. Nodes that are located within this radius are considered neighbors of the BMU.
5. Each neighbor node to the BMU adjusts its weights to coincide more with the input vector. The closer is the node of the BMU, more its weight will be altered.
6. Step 2 is repeated for "n" iterations.

To determine the best matching unit (BMU), an iterative process is performed by each node in the network and the Euclidean distance between the weight vector of each node and the current input vector is calculated. The node with the weight vector closest to the input vector is labeled as BMU. The Euclidean distance is calculated in (4).

$$\text{Distancia Euclidiana} = \sqrt{\sum_{i=0}^{n} (V_i - W_i)^2} \qquad (4)$$

Where "V" is the current input vector and "W" is the weight vector of each node in the network.

After determining what the BMU, the next step is to calculate which of the other nodes in the network, which are not the BMU, are in the neighborhood. Once identified these nodes, proceed to alter their weight vectors. Therefore, the radius of the neighborhood is calculated, using Pythagoras to determine whether each node it is within radial distance or not. The neighborhood area shrinks over time, since it is directly proportional to the radius of the neighborhood, which is calculated from the decreasing exponential function, defined in (5).

$$\sigma(t) = \sigma_0 \exp(\frac{t}{\lambda}), \; t = 1,2,3 \ldots \qquad (5)$$

Where σ_0 denotes the width of the frame in time “t_0” and λ denotes a constant time “t” which is the current time step. After several iterations the neighborhood will be adjusted to the size of a single node, the BMU, which will determine the value of the radius, which is required to identify whether a node is not within the neighborhood. If a node is located within the neighborhood, then the weight vector should be adjusted. Both the BMU node and each of the nodes located in the neighborhood, have a vector of weights adjusted according to (6).

$$W(t+1) = W(t) + \Theta(t)L(t)(V(t) - W(t)) \qquad (6)$$

Where "t" is time and "L" is a small variable called the learning rate, which decreases over time, equation (6) indicates that the weight at time "t +1" is set to neighborhood nodes from the current instant weight "W(t)", plus a fraction "L(t)", the difference between the current weight of "W(t)" node and the weight vector entry in the present moment "V(t)".

The learning rate as the radius of the neighborhood, using an exponential decay function to determine its value in the time variation. As shown in (7).

$$L(t) = L_0 \exp(-\frac{t}{\lambda}),\ \ t = 1,2,3 \ldots \qquad (7)$$

In equation (6), represents the amount of influence that the distance from one node to the BMU has on their learning, in the current time instant is calculated using (8).

$$\Theta(t) = \exp(-\frac{dist^2}{2\sigma^2(t)}),\ \ t = 1,2,3 \ldots \qquad (8)$$

Where “dist” is the distance from one node to the BMU and “σ” is the radius of the neighborhood, enunciated above. The function also decreases over time. Some specific examples of application of the SOM, can be consulted in [21-30].

5. METHODOLOGY

In the simulation experiments are analyzed of network connection records, from the NSL-KDD dataset DARPA. From a functional experiences was proposed an intrusion detection model, see Figure 2. The model comprises three phases: training, classification and calculation of performance metrics. For application several simulation scenarios were implemented by varying the amount of features to evaluate in the first two phases, thus the choice of features was prioritized by the Fisher discriminant ratio (FDR).

The **evaluation criteria used** to test the performance of the proposed methodology has been **cross-validation**, using two datasets with different data connections, metrics Sensitivity and Specificity were evaluated. The first dataset was used for the training phase and the second dataset for phase of classification and calculation of metrics.

In the training phase loads the dataset KDDTrain+_20Percent DARPA, balanced by connection type (normal or attack), this dataset represents the input data stream, then the feature reduction algorithm of FDR is applied and selecting features in order of relevance (see Table 1) and, finally, the SOM training is performed, implying a normalization, creating the data structure, initialization and training of the map, and a tagged data.

In the qualifying stage is loaded dataset KDDTest+ DARPA, which represents the flow of data to classify, different set of training data, the features are reduced using FDR generated from the new dataset, taking into has the same number of selected features in the training phase and finally proceeds to sort the data, generating a data structure containing both labeling of the new data as the predictive labeling from calculation based BMUs map created in the training phase.

In the final stage were calculated false positives, true positives, false negatives and true negatives, which allowed to determine the sensitivity and specificity which will indicate the efficiency of the proposed model. Several scripts were developed in matlab to operationalize the functional model of intrusion detection.

Prior to the training phase carry out the pre-processing that parsing and normalize the data from the NSL-KDD dataset DARPA. The parsing is performed for the purpose of handling the dataset in matlab. The parsing involved generating a matrix containing in each column the features of each connection in each row connections that corresponding to network traffic. Balancing data is also made by connection type in the preprocessing phase (the connection types are normal or attack traffic). The purpose normalize by type is a balance between the number of connections that refer to normal and attacks traffic, as if in the training phase, a machine learning algorithm will supply many more links of a given type is possible that during the qualifying generating a bias algorithm, by classifying more connections of a certain type, since this type of learned more during the training phase. The dataset used for the classification process is not balanced as it is intended to simulate as close to reality situation.

Moreover, in the training phase of the SOM has been necessary to implement the "somtoolbox" having matlab, which have been used the functions "som_normalize", "som_data_struct", "som_lininit", "som_batchtrain" and "som_autolabel" respectively for: standardization, creation of the data to be processed by the SOM, map initialization, training and labeling thereof.

In standardizing the "var" method was used in order to normalize to "1" all components of data through a linear process. Then the map is initialized with the data structure created in the previous step with a size of 4x4 (the map will consist of 16 nodes) and a hexagonal lattice type to represent such nodes. Map for training an initial neighborhood radius of 4 and a final neighborhood radius of 0.00001 was defined and Gaussian neighborhood function is set, moreover, the length of the training set is defined to 2000 and a trace level 'tracking' is set to 1. Have identified the values of these parameters as the most optimal, product of the analysis of iterative executions.

At the end of the initialization and training processes by implementing functions "som_lininit" and "som_batchtrain" respectively, get the trained map. The training phase culminates with the labeling map, using the "som_autolabel" function with the tagging mode 'vote' for the purpose of assigning tags as those that have been instantiated.

In qualifying was identified the best match units (BMU) of SOM, has been used to do the "som_bmus" function, of the "somtoolbox" matlab.

This function searches the BMU from the SOM to the arrangement of data supplied. The function returns an array containing the BMU for each vector dataset. By "smap.labels" function, are identified the tags projected by the SOM, which are then compared with the actual dataset labels. It is likely that the project labeling by implementing functions "smap.labels" and "som_bmus" some connections lack labels, why use a probabilistic method to assign labels to those units that have not done. To do this, using a Gaussian Mixture Model (GMM) the probability of occurrence of each vector data in terms of each core (node) of the SOM is estimated using the functions "som_estimate_gmm" and "som_probability_gmm". The latter returns the array containing the above probability.

The final process involves the creation of a nx2 array, where "n" represents the total number of connections and the two columns represent: the first, the labels projected for each connection record (indicating whether it is a normal connection = 0 or attack connection = 1) and the second, the actual labels of the data from the dataset KDDTest +.

In the final phase was calculated the metrics thrown by the implementation of intrusion detection model. The intention is to classify each connection according to its identification as: True Positive, False Positive, True Negative or False Negative, for it iterates through each data connection from dataset is classified, once done, the sensitivity is calculated (true positive rate) and specificity (true negative rate), which ultimately allow us to measure the efficiency of the proposed functional model.

Building on the priority level identified by FDR, 40 simulation scenarios using the proposed model were generated. Considering that in the training phase: was normalized using the linear operation of variance, the map with a size of 4x4 is initialized, is set for training an initial radius of 4 and a final radius of 0.00001, was used training length 2000 and a trace level of 1, and training map an algorithm SOM of batch was used. The optimal number of features, with a sensitivity of 97.39% and a specificity of 62.73% was identified as 17 (see Table 2).

The SOM was trained with 17 features selected with FDR. Given that the structure of the map is 4x4 and a hexagonal lattice type to display the label of each node (0 = normal, 1 = Attack). As can be seen in Figure 3. Each node in the topological map occupies a position that can be represented in 3D space, according to the application of the learning algorithm and following the Gaussian neighborhood function, using the "som_batchtrain" function from "somtoolbox" matlab. In Figure 4 you can see a spatial distribution of the nodes that make up the SOM after training.

6. DISCUSSION AND RESULTS

The present study was based on some benchmarks in relation to intrusion detection systems based on anomalies, in which different feature selection techniques are used in the preprocessing phase and other techniques for network traffic classification.

In [31] was presented an implementation of algorithms from FEAST, using a Matlab Toolbox with the purpose of selecting the method with better precision results for different attacks detection using the least number of features.

In [32] was presented a network anomaly detection technique based on Probabilistic Self-Organizing Maps (PSOM) to differentiate between normal and anomalous traffic. The detection capabilities of the proposed system could be modified without retraining the map, only modifying the activation probabilities of the units.

Is presented in [33] an intrusion detection technique using an ensemble of support vector classifiers and dimensionality reduction techniques (FDR+Kernel PCA, FDR+PCA, FDR+Isomap) to generate a set of discriminant features.

The main contribution of [34] is a multi-objective approach for feature selection and its application to an unsupervised clustering procedure based on Growing Hierarchical Self-Organising Maps (GHSOMs).

In [35] were used Chi-square (CS) and Info.Gain (IG) selection techniques, additionally in the training and classification phases the ANN, KNN, SVM and Liblinear algorithms were used. The performance of the methodology proposed in this paper, was measured in terms of the detection rate, precision, F-score and accuracy.

[36] presents preliminary results on proposed technique of using Membrane Computing (MC) to enhance the performance of a Bee Algorithm (BA) based feature selection of anomaly IDS. For SVM classification process was used for the experiments were randomly taken from Knowledge Discovery and Data mining KDD-Cup 99 dataset. The experiments, produced a high Attack Detection Rate (ADR) and significantly reduced False Alarm Rate (FAR).

In [37] is used for the classification of data the EDADT algorithm, for the preprocessing the Hybrid IDS model, a semi-supervised approach and a variation of HOPERAA Algorithm for the mitigating DDoS attacks. The data were analyzed with the help of the snort rules that are predefined within it and anomaly score for each packet. Under anomaly based approach, we have four types of statistical methods like Packet Header Anomaly Detection (PHAD), Network Traffic Anomaly Detector (NETAD), Application Layer Anomaly Detector (ALAD), Learning Rules for Anomaly Detection (LERAD).

The table 3 shows the comparison of intrusion detection methods referenced in the papers consulted on the proposed method, assessing metrics Sensitivity and Specificity.

The proposed method has a sensitivity of 97.39%, outperforming all other methods. However their specificity is less than the other methods. Importantly, the proposed method uses only 17 of the 41 features, no information on the number of features used by the other methods possess.

7. CONCLUSIONS

Whenever using the Fisher discriminant rate as feature selection method and SOM as training and classification technique, such a combination is more susceptible to detecting attacks (True Positive) that detecting normal traffic (true negative). This most likely because the probability calculated from the Gaussian Mixture Model (GMM) should be optimized to enhance the level of occurrence of each vector data in terms of each core (node) of the SOM.

The results obtained with the proposed method (SOM + FDR) have low rates of detection of normal traffic. Therefore, to increase the detection rates of normal traffic, in light of the literature, it is desirable to optimize such a probability of occurrence, using a fitness function from the implementation of a genetic algorithm. Other tests should also be performed seeking to optimize the initial size of the map, the initial radius and the end of the training set, and the training length.

8. FUTURE WORK

Recreate comparative scenarios to evaluate different features reduction techniques, supported by the NSL-KDD dataset and implementing the SOM learning algorithm and GHSOM.

Recreate two similar scenarios, as the previous, one that implements a genetic algorithm for classification of normal and abnormal traffic, and another that implements the genetic algorithm in the process of categorical classification of traffic in computer networks (DoS, R2L, U2R, Probe and Normal).

REFRENCES:

[1] The NSL-KDD Dataset. http://nsl.cs.unb.ca/NSL-KDD/.

[2] SourceFire - Snort. http://www.snort.org/.

[3] CheckPoint® Software Technologies Ltd. NFR (Network Flight Recorder). http://www.checkpoint.com/corporate/nfr/index.html.

[4] CISCO System. Cisco Intrusion Detection (NetRanger). http://www.cisco.com/warp/public/cc/pd/sqsw/sqidsz/index.shtml.

[5] IBM. RealSecure Network Sensor. http://www-947.ibm.com/support/entry/portal/Overview/Software/Tivoli/RealSecure_Network_Sensor.

[6] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani. “A Detailed Analysis of the KDD CUP 99 Data Set”, IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009. CISDA 2009, pp. 1 – 6, july 2009.

[7] M. Shyu, S. Chen, K. Sarinnapakorn, and L. Chang. “A novel anomaly detection scheme based on principal component classifier,” Proceedings of the IEEE Foundations and New Directions of Data Mining Workshop, in conjunction with the Third IEEE International Conference on Data Mining (ICDM03), pp. 172–179, 2003.

[8] USC Information Sciences Intitute. “Common Intrusion Detection Framework”, http://gost.isi.edu/cidf/. Enero-2014.

[9] CIDF Working Group (Clifford Kahn, Don Bolinger and Dan Schnackenberg). DRAFT Specification. Communication in the Common Intrusion Detection Framework v 0.7. 8 June 1998. http://gost.isi.edu/cidf/drafts/communication.txt.

[10] R. Feiertag, C. Kahn, P. Porras, D. Schnackenberg. A Common Intrusion Specification Language (CISL). 11 June 1999. http://gost.isi.edu/cidf/drafts/language.txt.

[11] Common Vulnerabilities and Exposures – CVE. http://cve.mitre.org/about/index.html.

[12] Prelude Technologies. http://www.prelude-technologies.com/.

[13] SRI - International a real-time Intrusion-Detection Expert System (IDES). http://www.csl.sri.com/papers/9sri/9sri.pdf.

[14] S. Noel, D. Wijesekera, C. Youman. Modern Intrusion Detection, Data Mining, and Degrees of Attack Guilt. In Applications of Data Mining in Computer Security, D. Barbarà and S. Jajodia (eds.), Kluwer Academic Publisher, 2002.

[15] A. Lazarevic, J. Srivastava, V. Kumar. A Survey of Intrusion Detection techniques. book "Managing Cyber Threats: Issues, Approaches and Challenges", to be published by Kluwer in spring 2004.

[16] A. Balakrishnama. Linear Discriminant Analysis - A Brief Tutorial, Institute for Signal and Information Processing, Department of Electrical and Computer Engineering, Mississippi State University. 1998.

[17] R. Fisher. The Use of Multiple Measurements in Taxonomic Problems In: Annals of Eugenics, 7, p. 179—188. 1936.

[18] V. Venkatachalam, S. Selvan. Performance comparison of intrusion detection system classifiers using various feature reduction techniques. International journal of simulation, 2008 - Citeseer.

[19] T. Kohonen. “Self-organizing Maps”. Springer Series in Information Sciences. Volume 30, 1997. 2nd edition.

[20] Kohonen’s Self Organizing Feature Maps. http://www.ai-junkie.com/ann/som/som1.html.

[21] D. Phuc, M. Xuan. Using SOM based Graph Clustering for Extracting Main Ideas from Documents. Research, Innovation and Vision for the Future, 2008. RIVF 2008. IEEE International Conference on. p, 209 – 214. July 2008.

[22] I. Manolakos, E. Logaras. High throughput systolic SOM IP core for FPGAs. Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on. P, II-61 - II-64. April 2007.G

[23] K. Yin, L. Gang. Fault Pattern Recognition of Thermodynamic System Based on SOM. Electrical and Control Engineering (ICECE), 2010. International Conference on. P, 3742 – 3745. June 2010.

[24] L. Min, W. Dongliang. Anormaly Intrusion Detection Based on SOM. Information Engineering, 2009. ICIE '09. WASE International Conference on. P, 40 – 43. July 2009.

[25] J.C. Patra, J. Abraham, P.K. Meher, G. Chakraborty. An Improved SOM-based Visualization Technique for DNA Microarray Data Analysis. Neural Networks (IJCNN), The 2010 International Joint Conference on. P, 1 – 7. July 2010.

[26] B. Fritzke. (1995). A growing neural gas network learns topologies. In Tesauro, G., Touretzky, D. S., and Leen, T. K., editors, Advances in Neural Information Processing Systems 7, pages 625–632. MIT Press, Cambridge MA.

[27] T. Martinez, K. Schulten, (1994). Topology representing networks. Neural Networks, 7(3):507–522.

[28] A. Ocsa, C. Bedregal, E. Cuadros-Vargas, (2007). DB-GNG: A constructive self-organizing map based on density. In Proceedings of the International Joint

Conference on Neural Networks (IJCNN07). IEEE.

[29] Y. Prudent, A. Ennaji. (2005). A k nearest classifier design. ELCVIA, 5(2):58–71.

[30] R. H. White. (1992). Competitive hebbian learning: algorithm and demonstrations. Neural Networks, 5(2):261–2.

[31] F. Mendoza, E. de la hoz, A. de la hoz, Application of feast (Feature Selection Toolbox) in IDS (Intrusion Detection Systems), vol.70, No.3, Journal of Theoretical and Applied Information Technology – JATIT, 2014, pp. 579-585.

[32] E. de la Hoz, E. de la Hoz, A. Ortiz, J. Ortega, Network anomaly detection with Bayesian self-organizing maps, in: Proceedings of the International Work-Conference on Artificial Neural Networks (IWANN), LNCS, vol. 7092, Springer-Verlag, 2013, pp. 532–537.

[33] E. de la Hoz, A. Ortiz, J. Ortega, E. de la Hoz. Network Anomaly Classification by Support Vector Classifiers Ensemble and Non-linear Projection Techniques, in: Proceedings of the International Conference on Hybrid Artificial Intelligence Systems (HAIS), LNAI, vol. 8073, Springer-Verlag, 2013, pp. 103–111.

[34] E. de la Hoz, E. de la Hoz, A. Ortiz, J. Ortega, A. Martínez-Álvarez, Feature selection by multi-objective optimisation: Application to network anomaly detection by hierarchical self-organising maps, vol. 71, Knowledge-Based Systems, 2014, pp. 322-338.

[35] C. Guo et al. Efficient intrusion detection using representative instances, vol. 39, Computers & Security, 2013, pp. 255-267.

[36] R. K. Idowu et al. An Application of Membrane Computing to Anomaly-Based Intrusion Detection System, vol. 11, Procedia Technology, 2013, pp. 585–592.

[37] G.V. Nadiammai, M. Hemalatha. Effective approach toward Intrusion Detection System using data mining techniques, vol. 15, Egyptian Informatics Journal - Cairo University, 2014, pp. 37–50.

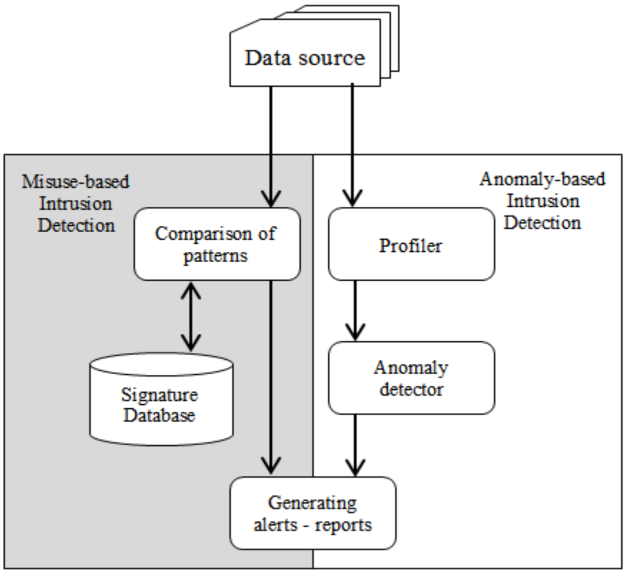
Data source
Misuse-based Intrusion Detection
Comparison of patterns
Signature Database
Generating alerts - reports
Anomaly-based Intrusion Detection
Profiler
Anomaly detector

Figure 1: Architecture of IDSs by type of analysis

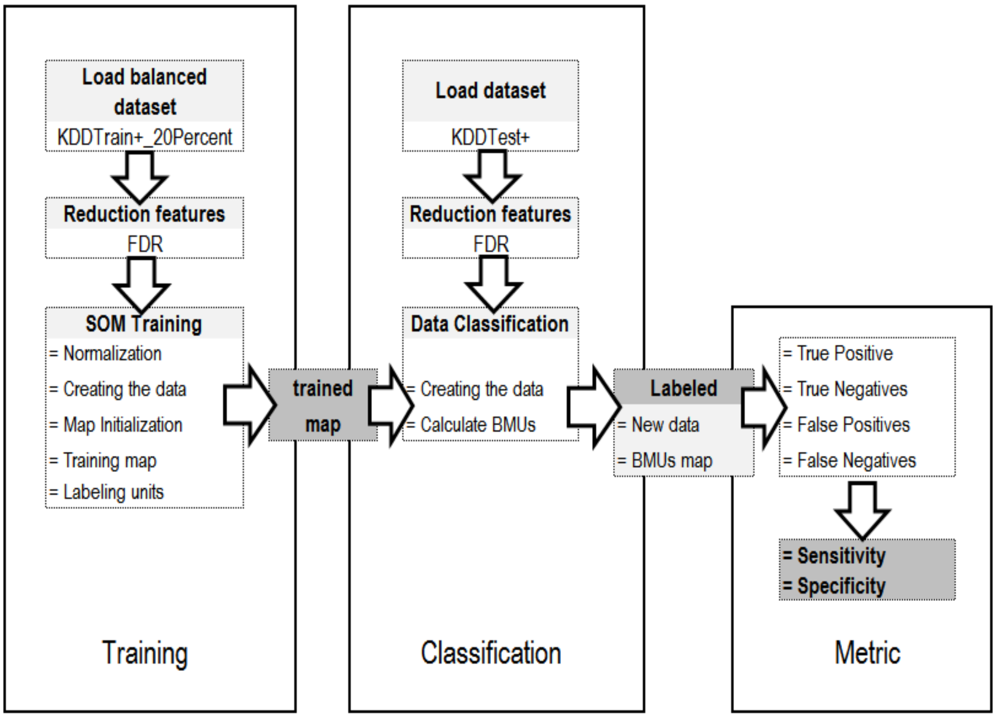

Figure 2: Functional model proposed intrusion detection

Table 1: Priority level assigned by FDR at 41 features dataset

Priority	FDR	FDR_IND	Feature	Type
1	17,5819	34	dst_host_same_srv_rate	continuous
2	17,3008	29	same_srv_rate	continuous
3	10,4640	12	logged_in	symbolic
4	6,1303	38	dst_host_serror_rate	continuous
5	6,0634	25	serror_rate	continuous
6	5,9486	39	dst_host_srv_serror_rate	continuous
7	5,8595	26	srv_serror_rate	continuous
8	2,6535	35	dst_host_diff_srv_rate	continuous
9	2,3114	30	diff_srv_rate	continuous
10	1,0948	40	dst_host_rerror_rate	continuous
11	1,0025	27	rerror_rate	continuous
12	1,0002	9	urgent	continuous
13	0,9725	41	dst_host_srv_rerror_rate	continuous
14	0,9643	28	srv_rerror_rate	continuous
15	0,7549	18	num_shells	continuous
16	0,4517	31	srv_diff_host_rate	continuous
17	0,4220	37	dst_host_srv_diff_host_rate	continuous
18	0,3833	14	root_shell	continuous
19	0,3242	15	su_attempted	continuous
20	0,2985	4	flag	symbolic
21	0,2952	22	is_guest_login	symbolic

Priority	FDR	FDR_IND	Feature	Type
22	0,1854	19	num_access_files	continuous
23	0,1527	36	dst_host_same_src_port_rate	continuous
24	0,1253	8	wrong_fragment	continuous
25	0,0131	2	protocol_type	symbolic
26	0,0018	17	num_file_creations	continuous
27	0,0006	3	service	symbolic
28	0,0003	33	dst_host_srv_count	continuous
29	0,0001	10	hot	continuous
30	0,0001	23	count	continuous
31	0,0000	32	dst_host_count	continuous
32	0,0000	13	num_compromised	continuous
33	0,0000	16	num_root	continuous
34	0,0000	24	srv_count	continuous
35	0,0000	1	duration	continuous
36	0,0000	6	dst_bytes	continuous
37	0,0000	5	src_bytes	continuous
38	0,0000	7	land	symbolic
39	0,0000	11	num_failed_logins	continuous
40	0,0000	20	num_outbound_cmds	continuous
41	0,0000	21	is_host_login	symbolic

Table 2: Comparison Metrics simulation environments

Number of features	Metrics		Eligibility Criteria
	Sensitivity	Specificity	
2	91,822698	61,723270	76,77
3	94,262295	43,278922	68,77
4	99,486125	51,956895	75,72
5	99,363492	53,364563	76,36
6	99,536354	54,435642	76,99
7	99,122056	54,100929	76,61
8	99,233455	54,252022	76,74
9	98,151214	53,352526	75,75
10	97,064691	50,475440	73,77
11	98,482111	56,596120	77,54
12	98,436857	54,490442	76,46
13	99,689730	50,209616	74,95
14	99,180514	53,302848	76,24
15	98,479792	56,145773	77,31
16	98,473851	56,077107	77,28
17	**97,394844**	**62,732305**	**80,06**
18	98,484568	56,201483	77,34
19	98,484568	56,201483	77,34
20	**99,694699**	51,211657	75,45

Number of features	Metrics		Eligibility Criteria
	Sensitivity	Specificity	
21	98,769988	52,280968	75,53
22	**98,476374**	55,487067	76,98
23	98,497110	55,508816	77,00
24	56,834245	0,000000	28,42
25	44,092888	1,522843	22,81
26	44,096134	1,523129	22,81
27	55,501514	1,733333	28,62
28	55,369884	1,948843	28,66
29	38,834520	1,263220	20,05
30	39,339510	1,288004	20,31
31	35,496441	11,990434	23,74
32	28,182383	14,328808	21,26
33	28,181012	14,342735	21,26
34	58,855688	45,881841	52,37
35	34,494319	36,064047	35,28
36	29,761717	8,196555	18,98
37	26,171108	37,299120	31,74
38	28,266162	37,454237	32,86
39	81,993205	44,097687	63,05
40	81,993205	44,097687	63,05
41	81,993205	44,097687	63,05

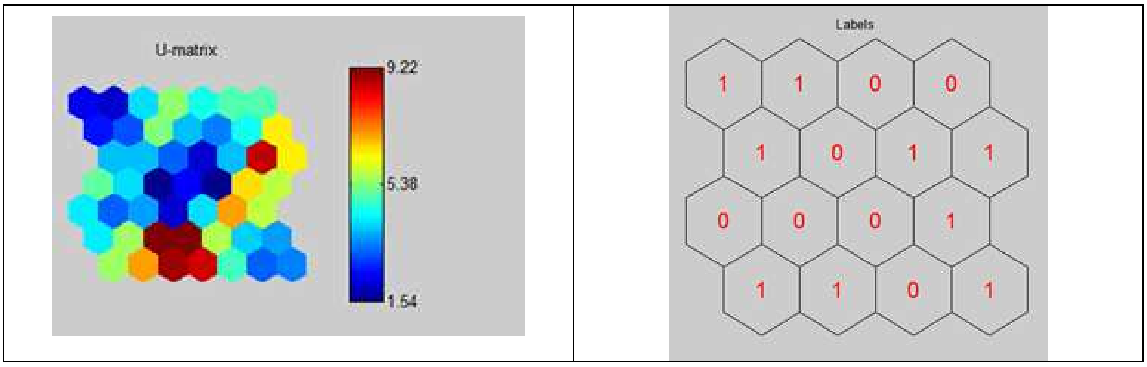

Figure 3: SOM structure (hexagonal lattice labeling)

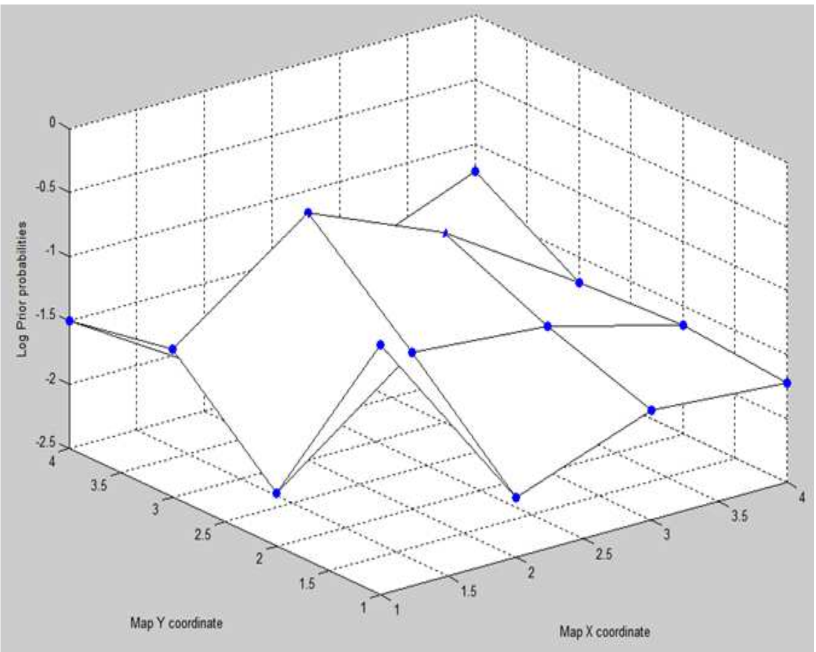

Figure 4: Spatial representation of SOM nodes

Table 3: Performance of technique proposed vs. existing algorithms

Algorithms	Sensitivity (%)	Specificity (%)
C4.5	86,57	82,00
SVM	83,82	64,29
C4.5+ACO	89,26	85,42
SVM+ACO	87,42	67,95
C4.5+PSO	92,51	88,39
SVM+PSO	90,06	70,8
EDADT	96,86	92,36
Proposed SOM+FDR	**97,39**	62,73

View publication stats