International Journal of Research in Advent Technology (IJRAT)
Vol. 1, Issue 3, October 2013, ISSN: 2321–9637

RULE LEARNIG BASED SELF ORGANIZING INTRUSION DETECTION SYSTEM

Sandip R. Sonawane [1], Shailendra M. Pardeshi [2], Vipul D. Punjabi [3], Rajnikant B. Wagh [4]
[1 2] Department of Information Technology, [3 4] Department of Computer Engineering, RCPIT Shirpur
[1 2 3] Assistant Professor, [4] Associate Professor
sandipsonawane2006@rediffmail.com, pardeshishailendra1184@gmail.com

ABSTARCT:
Recently, security has become a key issue in information technology as the number of computer security breaches are exposed to an increasing number of security threats. To identify these malicious threats various data-mining and machine learning algorithms and techniques have been developed for intrusion detection systems (IDS) and are used for protecting computers and networks from the different malicious attacks and threats. In existing IDS system the manual tuning process depends on the human operators in working out the tuning solution and it integrates it into the detection model. This paper focuses on intrusion detection system which makes the tuning automatically. The key idea is to use the binary SLIPPER as a basic module, which is a general purpose rule learning algorithm based on confidence-rated boosting. This system is evaluated using the NSL-KDD intrusion detection dataset. An experimental result shows this system with SLIPPER algorithm gives better performance in terms of detection rate, false alarm rate, total misclassification cost and cost per example on NSL-KDD dataset rather than that of on KDD dataset.

Keywords*-Intrusion, attacks, confidence value, false positiv, false predictio, total misclasification cost, tuning.*

I. INTRODUCTION

Securing important data from malicious users has been a long time concern for many both in the industry as well as in research. Nowadays many applications which access large databases over a network the needs detection of unauthorized intrusion. The inspection process and event monitoring of the network infrastructure is mostly performed using Intrusion Detection Systems (IDSs) [C. Lin and J. Leneutre (2009)], including network-based IDS (NIDS) [Zhang Jiong et al. (2008)] and host-based IDS (HIDS) [J. K. Hu et al. (2009)]. In recent years, to protect the computers and networks the numbers of different intrusion detection systems has developed by the intrusion detection community. Upon more IDSs are developed, network security administrators are confronted with the task of analyzing enormous of alerts resulting from the analysis of different event streams, but still there are some issues [Eric Maiwad (2001)] that should be consider in the current IDS like low detection rate, high misclassification cost, and high false positives.

The rest of this paper is organized as follows. Section **II** covers the related work in IDS. Section **III** describes proposed work and datasets used in this system in briefly. Section **IV** explains rule set creation and experimental results and finally, this paper ends with concluding remarks in section **V.**

II. RELATED WORK

Multiclassifier system [M. Sabhnani and G. Serpen (2004)] built a using multilayer perceptons, K-means clustering, and a Gaussian classifier and machine learning algorithms on the KDDCup'99 dataset. This approach evaluates performance of pattern recognition and machine learning algorithms on four attack categories of attacks as found in the KDD 1999 Cup intrusion detection dataset. The TMC of this multiclassifier system is 71 096, and the cost per example is 0.2285. However, the significant drawback of their system is that the multiclassifier model was built based on the performance of different sub classifiers on the test dataset.

An approach [Latium Khan et al. (2007)] proposed for detecting the various attacks and anomalies. For attack classification they used Support Vector Machines (SVM). This approach was compared with the Rocchios Bundling technique. Accuracy rate of this SVM + DGSOT is the best for DOS type of attack, which is 97% and it is improved as compared to pure SVM. False Negative rate is lowest (3% for DOS) for SVM + DGSOT and False Positive rate is as low as pure SVM (2%) whereas for U2R type of attacks the performance is poor. In this case the accuracy is found only 23% with False Positive 100% and False Negative 76%. Tsong and et al. [Hwang et al. (2007)] presents a three-tier architecture of intrusion detection system which consists of a blacklist, a whitelist and a multi-class SVM classifier. They designed three-tier IDS based on the KDD'99 benchmark dataset. They prepare a blacklist at the first tier and a whitelist at the second tier. They used multiclass SSVMs classification method at the third tier to classify anomalies those detected by whitelist into the four attack categories. The detection performance was found up to 94.71% and the false alarm rate was only 3.8%. They concluded that their results are better than those of KDD'99 winner's.

Proposed method for [Weiming Hu et al. (2008)] an intrusion detection algorithm based on the AdaBoost algorithm. To learn the classifier he uses the discrete AdaBoost algorithm. In their algorithm, they used a decision stumps as weak classifiers. By using algorithm False alarm rate ranges from 0.31-1.79% with detection rate 90.04%-90.88% as compared to Genetic Clustering method giving 0.3% false alarm rate with detection rate as 79%. and RSS-DSS method giving 0.27%-3.5% false alarm rate with detection rate varying from 89.2% to 94.4%. [R. Agarwal and M. Joshi (2008)] proposed an improved two stage general-to specific framework (PNrule) for learning a rule-based model and developed a new solution framework for the multi-class classification problem in data mining. The method is especially applicable in situations where different classes have widely different distributions in training data. They applied the technique to the Network Intrusion Detection Problem (KDD-CUP'99). The proposed model consists of positive rules that predict presence of the class, and negative rules that predict absence of the class. For multiclass classification, a cost-sensitive scoring algorithm was developed to resolve conflicts between multiple classifiers using a misclassification cost matrix, and the final prediction was determined according to Bayes optimality rule. The Total Misclassification Cost (TMC) is 74 058, and the Cost Per Example (CPE) is 0.2381 when tested on KDDCup'99 dataset.

[Kumar et al. (2009)] applied RIPPER to KDDCUP'99 dataset. RIPPER binary learning algorithm is an optimized version of IREP algorithm to reduce error on large datasets. RIPPER was selected to train a model on the 10% subset of the training dataset, and tested on entire test set. The Total Misclassification Cost is 73622, and the Average Misclassification Cost is 0.2367, which is same as the third rank of the contest. [Stefano Zanero et al.(2004)] proposed a novel architecture which implements a network-based anomaly detection system using unsupervised learning algorithms. They described how the pattern recognition features of a Self Organizing Map algorithm can be used for Intrusion Detection.Their final goal was to detect intrusions, separate packets with anomalous or malformed payload from normal packets The prototype was ran over various days of the 1999 DARPA dataset. A 66.7% detection rate with as few as 0.03% false positives was obtained. The detection rate was maximum up to 88.9% for threshold 0.09% with a false positive rate 0.095%. [Zhenwei YU et al.(2007)]. They presented an automatically tuning intrusion detection system, which controls the number of alarms output to the system operator and tunes the detection model on the fly according to feedback provided by the system operator when false predictions are identified. The system was evaluated using the KDDCup'99 intrusion detection dataset. They proposed an adaptive and automatically tuning intrusion detection system, ADAT: Here, a prediction filter is used to push only the most suspicious predictions to the system operator to be verified.. Second, the system tunes the detection model when false predictions are identified and adjusts the tuning strength based on monitoring the performance of the detection model on earlier data. ADAT reduced total misclassification cost (52294 as compared to 70177 of MC Slipper) by 25.5%, while increasing overall accuracy by 1.78%. Compared to the automatically tuning IDS with delayed tuning, ADAT reduced TMC by 6.76%. To build the optimal decision forest [Stefano Zanero et al.(2004)]. Levin proposed Kernel Miner. The tool won the second place in the KDD'99 contest. A global optimization criterion was used to minimize a value of the multiple estimators including the total MC. The 10% subset of the training dataset was used to build the decision forest. The Total Misclassification Cost (TMC) is 73243, and the Average Misclassification Cost is 0.2356.

From the literature survey it is observed that all of above proposed system were used a two most popular benchmarks i.e. KDDCUP'99 dataset and RIPPER binary rule algorithm for evaluating the performance of existing IDSs, but these benchmarks has the several drawbacks and they are as follows:

1. KDDCup' 99 dataset suffers from two deficiencies:

A. Duplicate Records

The first important drawback of the KDD data set is the huge number of duplicate records. Analyzing KDD train and test sets, it may found that about 78% and 75% of the records are duplicated in the train and test set, respectively. This large amount of redundant records in the train set will cause the classifier to be biased more towards the more frequent records, and thus prevent it from learning less frequent records which are usually more harmful to networks such as U2R attacks.

B. Unequal Distribution of Connection Types

The second drawback of the data set lies with the distribution of its 5 classes – Normal connections and the 4 intrusion types: DOS, probe, U2R, R2L. The DOS & normal connection comprise a 98% of the entire original data set, and 97% of the improved dataset, after removing duplicate instances. This imbalance makes it very difficult to train classifiers on the training set, and results in having extremely poor detection rates.

2. RIPPER was used in MADAM ID [Mansour M et al. (2009)] to select features and build classifier models. This algorithm also facing some problems as follows:

- The rule sets produced by RIPPER & IREP are larger in a size
- It achieves higher error rates
- Less efficient on the larger size datasets
- Less efficient in terms of determining false positive.

3. In most of the existing IDS system, tuning is not performed and if performed it should be done manually and existing IDSs uses all the 41 features of dataset records. But it is observed that some of the features are not essential while creating the rule sets.

III. PROPOSED WORK

The figure given below shows the flowchart of proposed work. From the figure the data preprocessor prepares the binary training dataset from the original training dataset and then create the rule sets by using SLIPPER algorithm. Then next prediction engine analyzes and evaluates each obtained data record and makes the prediction according to the prediction model and reports the prediction result to system operator. System operator then verifies the result and marks false predictions which are then fed back to the model tuner. The model tuner tunes the model automatically according to the feedback received from the system operator.

It uses NSL KDD dataset and SLIPPER as a a binary rule learning algorithm.

NSL KDD DATASET DESCRIPTIONS:

NSL-KDD is a data set [15] suggested to solve some of the inherent problems of the KDDCup'99 data set and has some advantages over KDDCup99. This dataset is a solution to solve the two issues mentioned in last section. This data set has the following advantages over the original KDD data set [Wenke Lee et al (2000)]:

- The learners will not be biased more towards the more frequent records since it does not include redundant records in the dataset.
- The performances of the learners are not biased by the methods which have better detection rates on the frequent records because of absent of redundant records in the train set

- Equal distribution of connection type i.e. the number of selected records from each difficulty level group is inversely proportional to the percentage of records in the original KDD data set.

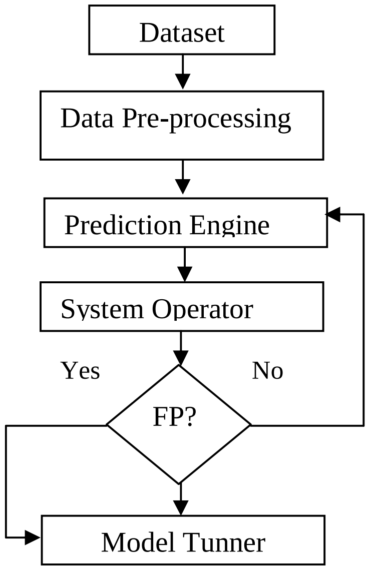

Figure 1 Flowchart of system

- **STEPS OF IMPLEMENTATION**

A. Data Pre processing

Initially preprocessing is done on original training data sets to build a binary classifier for each class and it generates proper training data for each class. An optimized preprocess procedure to reduce disk read is shown in algorithm given below. For each training example, if the label is not the target class name, then change the it to an unused class name, such as "other", otherwise, keep the label same.

Training Set T: {(feature $_i$, label $_i$)}, i= 1….N &
Class Set C:{(cname $_j$, counter $_j$, fname $_j$)},
j= 1…. M, where label i Є { c.cname | c Є C }
For each training example t Є T
For each class c Є C
If t.label ≠ c.name then
assign "other" to t.label
c.Counter + +
output t to c.fname
restore t.label

Optimized preprocessing algorithm

B. Creation of Rule set

SLIPPER algorithm is used to learn the set of binary classifier from the binary training dataset. Formally, it is based on confidence-rated boosting, a variant of AdaBoost. Rulesets created by SLIPPER are comprehensible, moderate in size. Following are the steps of SLIPPER algorithm:-

1. *Train the weak-learner using current distribution D:*

   a) Split data into GrowSet and PruneSet
   b) GrowRule: Starting with empty rule, greedily add conditions to maximize the equation

$$Z = \sqrt{W+} - \sqrt{W-} \quad \text{--------------------- (1)}$$

c) PruneRule: Starting with the output of GrowRule, delete some final sequence of conditions to minimize where C_R is computed using equation (3) and GrowSet

d) Return as R_t either the output of PruneRule or the default rule, whichever minimizes the equation

$$Z = 1 - (\sqrt{W+} - \sqrt{W-}) \quad \text{------------- (2)}$$

2. *Construct ht: X* ➔ R

Let C_R be given by

$$C_R = \frac{1}{2} \ln \left(\frac{W+ +1/(2n)}{W- +1/(2n)} \right) \quad \text{------------ (3)}$$

Then

$$ht(x) = \begin{cases} CRt, & if\ x \varepsilon Rt \\ 0, & otherwise \end{cases} \quad \text{------ (4)}$$

3. *Update:*

a) For each xi ε Rt, set D(i) ← D(i)/exp (yi. C_{Rt})

b) Let $Z_t = \sum_{i=1}^{m} D(i)$

c) For each xi, set D(i)= D(i)/ Z_t

Output final hypothesis

$$H(\infty) = sign \left(\sum_{Rt:x\,\varepsilon R\,t} CRt \right) \quad \text{-------- (5)}$$

In SLIPPER, a rule *R* is forced to abstain on all data records not covered by *R* and predicts with the same confidence C_R on every data record *x* covered by *R*

$$CR = \begin{cases} \frac{1}{2} \ln \left(\frac{W+}{W-}\right), & if\, x \in R \\ 0, & if\, x \in R \end{cases} \quad \text{------------------------ (6)}$$

W_+ and W_- represent the total weights of the positive and negative data records, respectively, covered by rule *R* in the round of boosting the rule, which was built in.

C. *Prediction Model*

The prediction model in this system consists of five binary prediction engines together with a final arbiter. After the analysis and evaluation on to the obtained input data, each binary prediction engine gives a prediction result according to its binary classifier, and the final arbiter determines and reports the result to the system operator.

The binary prediction engine is the same as the final hypothesis in SLIPPER, which is

$$H(\infty) = sign \left(\sum_{Rt:x\varepsilon Rt} CRt \right) \text{----------- (7)}$$

D. *Model Tunner*

During tuning, the associated confidence values are changed to adjust the contribution of each rule to the binary prediction. Consequentially, tuning ensures that, if a data record is covered by a rule in the original model, then, it will be covered by this rule also in the tuned model and vice versa. To limit possible side effects, change the associated confidence values of positive rules as a default rule covers every data record.

During tunning, tunned confidence value is obtained by

$$C'R = \begin{cases} p.CR, & if\ R\alpha P \\ q.CR, & if\ R\alpha N \end{cases} \quad \text{------------ (8)}$$

IV. IMPLEMENTATION AND RESULTS

A. *Creating Rule set*

In the experiment, Output of binary classifiers is rule set which contains the rules for particular type of

International Journal of Research in Advent Technology (IJRAT)
Vol. 1, Issue 3, October 2013, ISSN: 2321–9637

attack and default rule. The proposed work creates the rulesets for five types of attack and for creating the rulesets only essential features are used. Rulesets created by SLIPPER are comprehensible and moderate in size. SLIPPER uses only the essential features to create the ruleset.

B. False Prediction

In the experiment, the KDD dataset is used with the RIPPER learning algorithm for finding the false prediction count. It is determined by comparing the inputs files in the datasets with the output files. Here the selected rule with positive confidence is compared with a default rule with negative confidence to determine the result of boosting.

Table I False Prediction on KDD dataset

False Prediction Output

Attack	Input	Output	False Pre...
DoS	391194	363420	27774
R2L	1061	1081	20
U2R	52	43377	43325
Probe	4436	11443	7007
Normal	97228	74651	22577
Total	493971	493972	100703

Table II False Prediction on NSL- KDD dataset

False Prediction Output

Attack	Input	Output	False Pre...
DoS	377556	349191	28365
R2L	444	453	9
U2R	26	35444	35418
Probe	3272	10965	7693
Normal	74522	59773	14749
Total	455820	455826	86234

In the experiment, the NSL-KDD dataset is used with the SLIPPER for finding the false prediction count. It is calculated by comparing the inputs files in the datasets with the output files.

C. Tunned Confidence Value

Here the KDD dataset is used with RIPPER algorithm to determine the confidence value and tunned confidence value. Here the tunning is done manually. The detection rate is 93.78 % and false alarm rate is 6.2 %.

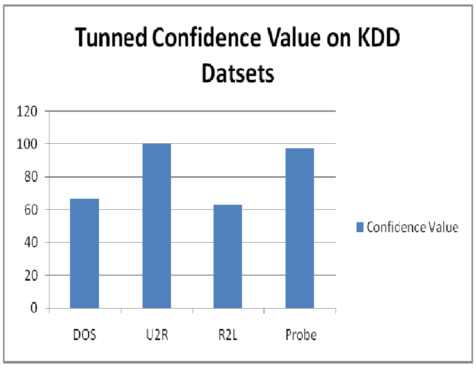

Figure 2 Tunned Confidence value on KDD Dataset

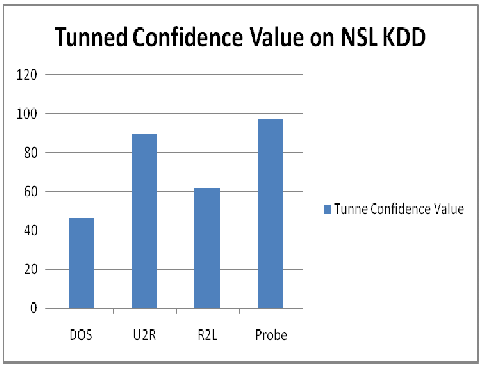

Figure 3. Tunned Confidence value on NSL KDD Dataset

From above figure the NSL-KDD dataset is used with SLIPPER algorithm to determine the confidence value and tunned confidence value. Here the tunning algorithm is used to improve the tunned confidence value. The detection rate is increased up to 97.20 % and false alarm rate is decreased up to 2.79 %.The detection rate and false alarm rate are determined by using following formulas:

Detection Rate = Number of attacks detected divided by no. of attacks present in the datasets.

FAR= Number of normal connections wrongly detected as attack divided by total no. of normal connections.

D. Performance Comparison Graph

The figure below shows the confidence value, detection rate and false alarm rate on KDD

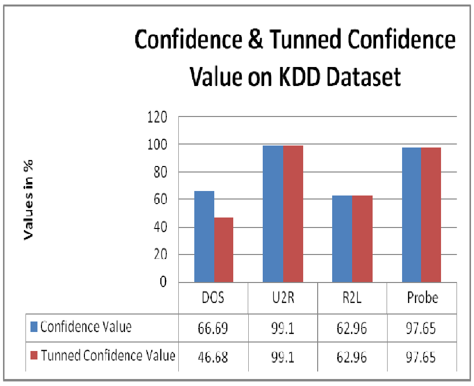

Figure 4.Confidence & Tunned Confidence value on KDD Dataset

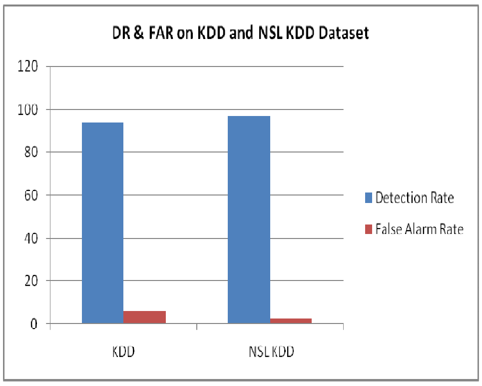

Figure 5. Detection Rate and False Alarm Rate on KDD & NSL KDD Dataset

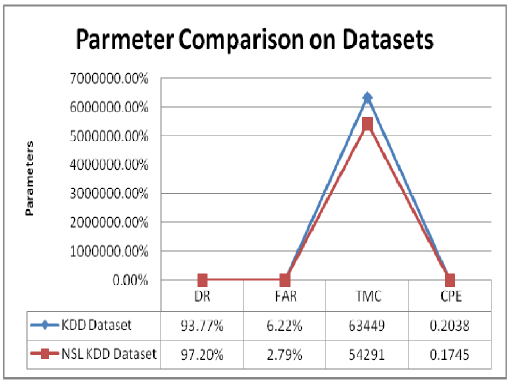

Figure 6. Graph showing performance comparison on Datasets

Above table shows performance comparison of various parameters on KDD & NSL KDD Datasets. The detection rate is increased by 3.43 % on NSL-KDD dataset and false alarm rate is decreased by 3.41 % on NSL-KDD dataset. Total Misclassification Cost (TMC) and Cost Per Example (CPE) are also decreases. The result on NSL-KDD dataset with the SLIPPER algorithm is better than that of on KDD with RIPPER algorithm.

V. CONCLUSION

Attacks on the network infrastructure presently are main threats against network and information security. Therefore the security is one of the crucial issues in modern computer system. Intrusion detection plays one of the key roles in computer security techniques and is one of the prime areas of research. The proposed work aims at discovering an efficient binary rule learning algorithm and applying that algorithm on NSL KDD dataset. In this approach tuning is to be done automatically by using model tuning algorithm. In order to allow tunned the model easily and precisely without affecting the rest of the model, It uses rules to represent the prediction model and it uses only essential 17 features of each data set record. Implementation and result shows that the this system by using SLIPPER algorithm as a basic module on NSL-KDD gives detection rate as high as possible and false alarm rate, total misclassification cost and cost per example as low as possible when compared to that on KDD dataset.

References

[1] C. Lin, J. Leneutre (2009): "A Game Theoretical Framework on Intrusion Detection in Heterogeneous Networks," *IEEE Transactions on Information Forensics and Security*, vol.4, no.2, pp.165-178.

[2] Zhang Jiong, M. Zulkernine, A Haque (2008): "Random-Forests-Based Network Intrusion Detection Systems," IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, vol.38, no.5, pp.649-659.

[3] J. K. Hu, X. G. Yu, D. Qiu, H. H. Chen (2009) "A simple and efficient hidden Markov model scheme for host-based anomaly intrusion detection", IEEE, Network, vol.23, no.1, pp.42-47.

[4] Eric Maiwad (2001): Network Security A Beginners Guide, Chief Technology Officer, TMH publications.

[5] M. Sabhnani and G. Serpen (2004): "Why machine learning algorithms fail in misuse detection on KDD intrusion detection data set," Journal-Intelligent Data Analysis ,vol. 8, no. 4, pp. 403–415.

[6] Latium Khan, Mamoun Awad, Bhavani Thuraisingham (2007): "A New Intrusion Detection SystemUsing Support Vector Machines And Hierarchical Clustering", The VLDB Journal 16, pp.507 – 521.

[7] Tsong Song Hwang ,Tsung JuLee,Yuh-Jye Lee (2007): "A Three tier IDS via Data Mining Approach", in Workshop on Mining Network Data (MineNet).

[8] Weiming Hu, ,SteveMaybank, (2008): "AdaBoost-Based Algorithm for Network Intrusion Detection", IEEE Transactions on System, Man and Cybernetics Part B: Cybernetics, Vol. 38, No. 2, PP.577-583

[9] R. Agarwal and M. Joshi (2008): "PNrule: A new framework for learning classifier models in data mining (a case-study in network intrusion detection)," in Proc.1st SIAM Conf. Data Mining.

[10] Amit Kumar Choudhary, Akhilesh Swarup (2009): "Neural Network Approach for Intrusion Detection", ACM International Conference Proceeding Series 403 ACM 2009, Seoul, Korea ACM 978-1-60558-710-3,pp 1297-1301.

[11] Stefano Zanero, Sergio M. Savaresi, "Unsupervised learning techniques for An intrusion detection system", SAC'04, Nicosia, Cyprus, ACM 1581138121/03/04, pp14-17.

[12] Zhenwei Yu, Jeffrey J. P. Tsai (2007): "An Automatically Tuning Intrusion Detection System IEEE Transactions on System, Man and Cybernetics Part B: Cybernetics, VOL.37, NO. 2, pp. 373-384.

[13] Stefano Zaner (2008): "Network Intrusion Detection System", In proceedings of the 4th annual workshop on Cyber security and information intelligence research, CSIIRW '08.

[14] Mansour M. Alsulaiman, Aasem N. Alyahya,Raed A.Alkharboush, Nasser S. Alghafis (2009): "Intrusion Detection System using Self-Organizing Maps", 2009 Third International Conference on Network and System Security,pp.397-402.

[15] The NSL-KDD Data Set, http://iscx.ca/NSL-KDD/

[16] Wenke Lee et al (2000): "A Framework for Constructing Features and Models for Intrusion Detection Systems", in ACM Transactions on Information and System Security, Vol. 3, No. 4, pp 227–261.

[17] J. McHugh (2000): "Testing intrusion detection systems: a critique of the 1998and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory," ACM Transactions on Information and System Security, vol. 3, no. 4, pp. 262–294.

[18] Zhenwei Yu, Jeffrey J.P. Tsai (2004): IEEE, "A Multi-Class SLIPPER System for Intrusion Detection", in proceedings of the 28th Annual International Compute Software and Applications Conference (COMPSAC'04), vol. 1, pp.212-217.

[19] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani (2009): "A Detailed Analysis of the 2000 Symposium on computational intelligence in KDDCUP 99 Data Set", in proceeding of IEEE security and defense application CISDA.

[20] S. J. Stolfo, W. Fan, W. Lee, A. Prodromidis, and P. K. Chan, "Cost base modeling for fraud and intrusion detection: results for the JAVA project",, In DARPA Information Survivability Conference and Exposition Hillton Head, SC vol.2.pp 130 – 144..

[21] I. Levin (1999): "KDD-99 classifier learning contest LLSoft's results overview,"ACM SIGKDD Explor., vol. 1, no. 2, pp. 67–75.

[22] A. H. M. Rezaul Karim,R, M. A. P. Rajatheva,Kazi M.Ahmed (2006): "An Efficient Collaborative Intrusion Detection System for MANET Using Bayesian Approach",pp.187-190.

[23] V. K. Pachghare, Parag Kulkarn, Deven M. Nikam (2009) "Intrusion Detection System Using Self Organizing Maps", 978-1-4244.

[24] Stefan Axelsson,(2004): "Combining a Bayesian Classifier with Visualisation Understanding the IDS",Washington. Oct 29, pp 81-104.

[25] Liberios Vokorokos, Anton Balaz, Martin (2006): "Intrusion Detection System Using Self Organising Map", Act a Electro technicaet Informatica No. 1,Vol. 6,pp.1-6.

[26] H. Günes Kayacık, A. NurZincir-Heywood (2006): "Using Self-Organizing Maps to Build an Attack Map for Forensic Analysis", In proceedings of the 2006 International Conference on Privacy, Security and Trust, Canada, pp 325-329.

[27] Jamie Fellner, Joanne Mariner, USA (2001): Maximum Security,Third edition, SamsPublications, Indianapolis, Indiana.

[28] Yuebin Bai, Hidetsune Kobayashi (2003): "Intrusion Detection System:Technology &Development", In Proceedings of the17th International Conference on Advanced Information Networking and Applications (AINA'03).

[29] A Murali M Rao (2005): ,A Survey on Intrusion Detection Approaches, First International Conference on Information and Communication Technologies, 2005. ICICT 2005. IEEE, pp 233-240.