RULE BASED LEARNING INTRUSION DETECTION SYSTEM USING KDD AND NSL KDD DATASET

Sandip Sonawane
Assistant Professor, Department of Information Technology,
R. C. Patel Institute of Technology, Shirpur.
Email: sandipsonawane2006@rediffmail.com

ABSTRACT

With the rapid expansion of computer networks during the past few years, security has become a crucial issue for modern computer systems. A good way to identify malicious use is through monitoring unusual user activity. To identify these malicious activities various data-mining and machine learning techniques have been deployed for intrusion detection. The manual tuning process required by current systems depends on the system operators in working out the tuning solution and in integrating it into the detection model. This paper proposes RULE LEARNING Intrusion Detection System (RLIDS) to make tuning automatically. The key idea is to use the binary SLIPPER as a basic module, which is a rule learner based on confidence-rated boosting. This system is evaluated using the NSL KDD intrusion detection dataset. An experimental result shows the RLIDS 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 than that of on KDD.

Keywords Intrusion, Attacks, Misuse Detection, Anomaly Detection, False Prediction, Confidence Value, Tuning.

I. INTRODUCTION

Attacks on network infrastructure presently are the threats against network and information security [1]. With rapidly growing unauthorized activities on the network, Intrusion Detection System (IDS) is very necessary because traditional firewalls cannot provide the complete security against the intrusion. Intrusion Detection (ID) is an active and important research area of network security. The goal of Intrusion Detection is to identify all the true attacks and negatively identify all the non-attacks [2].

The goals of the IDS provide the requirements for the IDS policy. Potential goals includes [3, 4]

- Detection of attacks
- Prevention of attacks
- Detection of policy violations

- Enforcement of use policies
- Enforcement of connection policies
- Collection of evidence

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

II. RELATED WORK

Sabhnani and Serpen et al. [5] built a multiclassifier system using multilayer perceptons, K-means clustering, and a Gaussian classifier after evaluating the performance of a comprehensive set of pattern recognition and machine learning algorithms on the KDDCup'99 dataset. This paper evaluates performance of a comprehensive set of pattern recognition and machine learning algorithms on four attack categories as found in the KDD 1999 Cup intrusion detection dataset. Results of simulation study implemented to that effect indicated that certain classification algorithms perform better for certain attack categories. A specific algorithm specialized for a given attack category. 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.

L. Khan and et al. [6] proposed an approach with a scalable solution for detecting the various attacks and anomalies. For classification of attack they used Support Vector Machines (SVM). The approach was compared with the Rocchios Bundling technique and random selection in terms of accuracy loss and training time gain using a single benchmark real data set. Accuracy rate of this SVM + DGSOT is the best for DOS type of attack, which is 97% and it is better as compared to pure SVM. FN is lowest (3% for DOS) for SVM + DGSOT and FP 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 FP 100% and FN 76%.

Tsong and et al. [7] introduced a three-tier architecture of intrusion detection system which consists of a blacklist, a whitelist and a multi-class support vector machine classifier. They designed a three-tier IDS based on the KDD'99 benchmark dataset. Thus to build a blacklist at the first tier and a whitelist at the second tier. Then they used one against one multiclass SSVMs classification method at the third tier to classify those anomalies 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.

Weiming Hu and et al [8] proposed an intrusion detection algorithm based on the AdaBoost algorithm. The discrete AdaBoost algorithm was selected to learn the classifier. In their algorithm, they selected 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%.

Agarwal and Joshi [9] 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 (P-rules) that predict presence of the class, and negative rules (N-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 TMC is 74 058, and the cost per example is 0.2381 when tested on KDDCup’99 dataset.

Amit Kumar Choudhary and et al [10] proposed a neural network approach to improve the alert throughput of a network and making it attack prohibitive using IDS. For evolving and testing intrusion the KDD CUP 99 dataset were used. They proposed the Generalized Regression Neural Network (GRNN) paradigm as an alternative to the popular Back propagation training algorithm for feed forward neural networks. The promising results of the present study shown the potential applicability of ANNs for developing high efficiency practical IDSs. This Neural Network model solved normal attack attack patterns, and the type of the attack. When given data was presented to the model, the results obtained revealed a great deal of accuracy app. 100%.

Stefano Zanero and et al. [11] 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 and et al. [12]. 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%.

Stefano Zanero et al. [13], presented a tool for network anomaly detection and network intelligence which was named as ULISSE. It uses two tier architecture with unsupervised learning algorithms to perform network intrusion and anomaly detection. It was concluded that their architecture can reach the same detection rate of 66.7% with a false positive rate below 0.03%, thus an order of magnitude better than PAYL, or on the other hand reach a 88.9% detection rate with no more than a 1% rate of false positives.

From the literature survey it is observed that most of the researchers may used a KDDCup'99 dataset and RIPPER binary rule algorithm for evaluating the performance of existing IDS.
KDD dataset suffers from two deficiencies:

A. Redundant Records

The first important deficiency in the KDD data set is the huge number of redundant 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 learning algorithms to be biased towards the more frequent records, and thus prevent it from learning infrequent records which are usually more harmful to networks such as U2R attacks.

B. Distribution of Connection Types

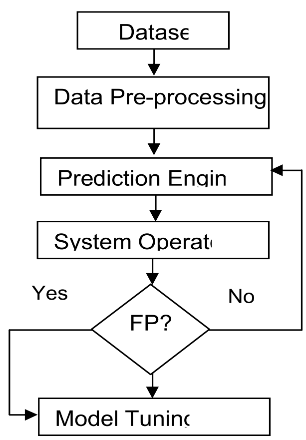
The second shortcoming of the Data set lies with the distribution of its 5 classes – Normal connections and the 4 intrusion types: DOS, probe, U2R, R2L. The first two classes comprise a whopping 98% of the entire original data set, and 97% of the improved dataset, after removing duplicate instances.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

RIPPER was used in MADAM ID [14] to select features and build classifier models. This algorithm also facing some problems as follows:

- The rulesets produced by RIPPER & IREP are larger in a size
- It achieves higher error rates
- Less efficient on the larger size datasets

III. PROPOSED WORK

From above figure data preprocessor prepares the binary training dataset from the original training dataset and then create the ruleset by using SLIPPER algorithm. Then next prediction engine analyzes and evaluates each obtained data record 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 automatically tunes the model according to the feedback received from the system operator.



IV.

Figure 1 Flowchart of RLIDS

The RLIDS uses NSL KDD dataset and SLIPPER 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 [16]:

- It does not include redundant records in the train set, so the classifiers will not be biased towards more frequent records.
- There are no duplicate records in the proposed test sets and train set; therefore, the performances of the learners are not biased by the methods which have better detection rates on the frequent records.

- The number of selected records from each difficulty level group is inversely proportional to the percentage of records in the original KDD data set.

STEPS OF IMPLEMENTATION

Pre processing of Data

To build a binary classifier for each class, preprocessing is done on training data to generate proper training data for each class. An optimized preprocess procedure to reduce disk read is shown in figure 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 ∈ { c.cname | c ∈ C }
For each training example t ∈ T
For each class c ∈ 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

To learn the set of binary classifier from the binary training dataset SLIPPER algorithm is used. Formally, it is based on confidence-rated boosting, a variant of AdaBoost. SLIPPER is fast, robust, and easy to use, and its hypotheses are compact and easy to understand.

Train the weak-learner using current distribution D:

Split data into GrowSet and PruneSet

GrowRule: Starting with empty rule, greedily add conditions to maximize the equation

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

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

Return as Rt either the output of PruneRule or the default rule, whichever minimizes the equation

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

Construct ht: X R

Let CR be given by

$CR = 1/2 \ln ((W + +1/(2n))/(W- +1/(2n)))$ ------------ (3)

Then

ht(x)={(CRt ,if &xε Rt@0,& otherwise)┤ ------ (4)

Update:

For each xi ε Rt, set D(i) D(i)/exp (yi. CRt)

Let Zt =∑_(i=1)^m D(i)

For each xi, set D(i)= D(i)/ Zt

Output final hypothesis

H(∞)=sign (∑_(Rt:x εR t@Rt:x∈Rt)CRt) -------- (5)

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

CR={((1)/2 In ((W+)/(W-)),&if∞ R@0,&if∞ R)┤ ------------------------- (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.

Prediction Engine

The prediction engine in this system consists of five binary prediction engines together with a final arbiter. Each binary prediction engine outputs a prediction result on the input data 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(∞)=sign (∑_(@Rt:x∈Rt)CRt)----------- (7)

Model Tunner

During tuning, the associated confidence values is improved 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.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

Creating Rule set

In the experiment, Binary classifiers are learned from the Simple learner with iterative pruning to produce error reduction (SLIPPER). Output of binary classifiers is rule set which contains the rules for particular type of attack and default rule.

False Prediction

In the experiment, the KDD dataset is used with the RIPPER learning algorithm for finding the false prediction count. It is calculated 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 learning algorithm 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 automatic tunning is not happen.

TABLE III TUNNED CONFIDENCE VALUE ON KDD DATASET

Max Confidence

Attacks	Confidence	Tunned Confi...
DoS	66.69443	66.69443
R2L	100.4819075	100.4819075
U2R	62.9627025	62.9627025
Probe	99.31768757	99.31768757

Detection rate is: **93.77932282881474**

FalseAlaram Rate is: **6.220677171185255**

TABLE IV TUNNED CONFIDENCE VALUE ON NSL-KDD DATASET

Max Confidence

Attacks	Confidence	Tunned Confi...
DoS	66.69443	46.6861009999...
R2L	100.4819075	100.4819075
U2R	62.9627025	62.9627025
Probe	99.31768757	99.31768757

Detection rate is: **97.2085290249859**

FalseAlaram Rate is: **2.791470975014093**

Here the NSL-KDD dataset is used with SLIPPER algorithm to determine the confidence value and tunned confidence value. Here the model tunning algorithm is used to improve the tunned confidence value.

D. Graph

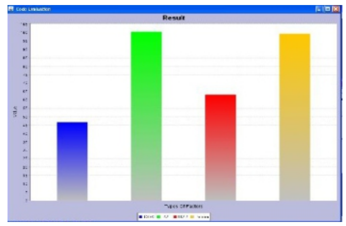


Figure 7.Graph showing confidence value on NSL-KDD Dataset

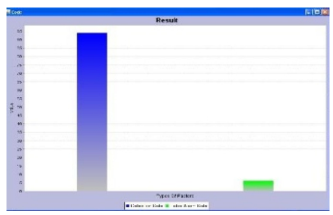


Figure 8 .Graph showing Detection Rate and False Alarm Rate on NSL-KDD Dataset

The figure above shows the confidence value, detection rate and false alarm rate on NSL- KDD.

TABLE V PERFORMANCE COMPARISON ON DATASETS

Parameter	KDD	NSL-KDD
Detection Rate	93.77 %	97.20 %
False Alarm Rate	6.22 %	2.79 %
T M C Value	63449	54291
C P E Value	0.2038	0.1745

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. 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. Experimental results and analysis shows that the RLIDS by using SLIPPER algorithm as a basic module on NSL-KDD gives better performance in terms of

1. High detection rate which is increased by 3.43 %
2. Low false alarm rate which is decreased by 3.41 %
3. Less Misclassification cost
4. Less Cost per example

REFERENCES

[1] Jamie Fellner, Joanne Mariner, USA 2001, Maximum Security, Third edition, Sams Publications, Indianapolis, Indiana.

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

[3] 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.

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

[5] M. Sabhnani and G. Serpen, "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, 2004

[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, "A Three tier IDS via Data Mining Approach", in Workshop on Mining Network Data (MineNet), June 12, 2007.

[8] Weiming Hu, ,SteveMaybank, APRIL 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, "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, Apr. 2001. [Online]. Available: http://www.siam.org/meetings/sdm01/pdf/sdm01_30.pdf

[10] Amit Kumar Choudhary, Akhilesh Swarup, "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, APRIL 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 Zanero, "Network Intrusion Detection System", In proceedings of the 4th annual workshop on Cyber security and information intelligence research, CSIIRW '08 Ma 12-14, 2008.
[14] Mansour M. Alsulaiman, Aasem N. Alyahya,Raed A.Alkharboush, Nasser S. Alghafis (2009). Intrusion Detection System using Self-Organizing Maps, Third International Conference on Network and System Security, 397-402.
[15] The NSL-KDD Data Set, http://iscx.ca/NSL-KDD/
[16] Wenke Lee et al, "A Framework for Constructing Features and Models for Intrusion Detection Systems", in ACM Transactions on Information and System Security, Vol. 3, No. 4, November 2000, pp 227–261.
[17] J. McHugh, "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, 2000.
[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, "A Detailed Analysis of the 2000 Symposium on computational intelligence in KDDCUP 99 Data Set", in proceeding of IEEE security and defense application CISDA, 2009.
[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., 1(2), 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, Intrusion Detection System Using Self Organizing Maps, 978-1-4244 IEEE, 2009.
[24] Stefan Axelsson, "Combining a Bayesian Classifier with Visualisation Understanding the IDS", Washington. Oct 29, 2004, pp 81-104.
[25] Liberios Vokorokos, Anton Balaz, Martin CHOVANEC, 2006 "Intrusion Detection System Using Self Organising Map", Act a Electro technicaet Informatica No. 1, Vol. 6,pp.1-6.
[26] Kayacik, H. G., & Zincir-Heywood, A. N. (2006, October). Using self-organizing maps to build an attack map for forensic analysis. In Proceedings of the 2006 International Conference on Privacy, Security and Trust: Bridge the Gap Between PST Technologies and Business Services (p. 33). ACM.
[27] Sabhnani, M., & Serpen, G. (2003, June). Application of Machine Learning Algorithms to KDD Intrusion Detection Dataset within Misuse Detection Context. In MLMTA (pp. 209-215).