Oriental Journal of Computer Science & Technology Vol. **3**(1), 165-170 (2010)

Appraise the recitation of intrusion detection system at training time

ANURAG LAL

CSE Department, CIT, Rajanadgaon (India)

(Received: April 23, 2010; Accepted: May 17, 2010)

ABSTRACT

Network Intrusion Detection aims at distinguishing the behavior of the network. It is an inseparable part of the information security system. Due to rapid development of attack pattern it is necessary to develop a system which can upgrade itself as new threats are detected. Also detection rate should be high because the rate with which attack is carried out on the network is very high. In response to this problem AdaBoost Based Algorithm is proposed which has high detection rate as well as low false alarm rate. In this algorithm decision stumps are used as weak classifier. The decision rules are provided for both categorical and continuous features. Weak classifier for continuous features and weak classifier for categorical features are combined to form a strong classifier. Strategies for avoiding the over fitting are adopted to improve the performance of the algorithm.

Keywords: Intrusion Detection System, Adaboost Algorithm, Security, Machine Learning, Neural Networks.

INTRODUCTION

There should be no question that one of the most pervasive technology trends in modern computing is an increasing reliance on network connectivity and inter-host communication. Along with the tremendous opportunities for information and resource sharing that this entails comes a heightened need for information security, as computing resources are both more vulnerable and more heavily depended upon than before.

One subset of information security research that has been the subject of much attention in recent years is that of intrusion detection systems. The National Institute of Standards and Technology classifies intrusion detection as "the process of monitoring the events occurring in a computer system or network and analyzing them for signs of intrusions, defined as attempts to compromise the confidentiality, integrity, availability, or to bypass the security mechanisms of a computer or network"[1]. This definition captures the essence of intrusion detection but fails to address the methods by which Intrusion Detection Systems (IDS's) automate this process. The concepts of false positive and false negative are essential to this classification process. False positives are those sequences of innocuous events that an IDS erroneously classifies as intrusive, while false negatives refer to intrusion attempts that an IDS fails to report: the reduction of both false positives and false negatives is a critical objective in intrusion detection.

There are two main approaches to design an IDS: misuse based IDS and anomaly based IDS[2]. In a misuse based intrusion detection system, intrusions are detected by looking for activities that correspond to know signatures of intrusions or vulnerabilities[3]. While an anomaly based intrusion detection system detect intrusions by searching for abnormal network traffic. The abnormal traffic pattern can be defined either as the violation of accepted thresholds for frequency of events in a connection or as a user's violation of the legitimate profile developed for normal behavior.

Modern IDS's are extremely diverse in the techniques they employ to gather and analyze data. Most rely, however, on a common architecture for their structure: a detection module gathers data that may contain evidence of intrusions, an analysis engine processes this data to identify intrusive activity, and a response component reports intrusions. One of the most commonly used approaches in expert system based intrusion detection systems is rule-based analysis using Denning's profile model[3]. Rule-based analysis depends on sets of predefined rules that are provided by an administrator. Expert systems require frequent updates to remain current. This design approach usually results in an inflexible detection system that is unable to detect an attack if the sequence of events is slightly different from the predefined profile[4]. Considered that, the intruder is an intelligent and flexible agent while the rule based IDSs obey fixed rules. This problem can be tackled by the application of soft computing techniques in IDSs. Soft computing is a general term for describing a set of optimization and processing techniques. The principal constituents of soft computing techniques are Fuzzy Logic (FL), Artificial Neural Networks (ANNs), Probabilistic Reasoning (PR), and Genetic Algorithms (GAs)[4].

Related Work

Supervised Learning Based Approaches

In recent years, methods from machine learning and pattern recognition have been utilized to detect intrusions. Both supervised learning and unsupervised learning are used. There are mainly supervised neural network (NN)-based approaches[5,6], and support vector machine (SVM)-based approaches[7,8] are used in supervised learning for intrusion detection.

a) NN-based approaches

Bonifacio *et al.* [9] have proposed an neural network for distinguishing between the behaviours of intrusions and normal. They unify the coding of categorical fields and the coding of character string fields in order to map the network data to an neural network. Rapaka *et al.* [10] use execution numbers of system calls in a host machine as the features of network behaviours to train the neural network. Zhang *et al.* [6] propose an approach for intrusion detection using hierarchical neural networks. Han and Cho[11] use evolutionary neural networks to detect intrusions.

b) SVM-based approaches

Mukkamala *et al.* [12,13] use SVMs to distinguish between normal and intrusions network behaviours and further identify important features for intrusion detection. Mill and Inoue[14] propose the TreeSVM and ArraySVM for solving the problem of inefficiency of the sequential minimal optimization algorithm for the large training data set in intrusion detection. Zhang and Shen[8] propose an approach for online training of SVMs for real-time intrusion detection based on an improved text categorization model. Also for intrusion detection, decision tree[15] and discriminate analysis[16] are applied. Comparisons between different classifiers and fusion of multiple classifiers for intrusion detection are studied in [18,19], and [17].

Unsupervised Learning Based Approaches

Supervised learning methods for intrusion detection can only detect known intrusions. Unsupervised learning methods can detect the intrusions that have not been previously learned. K-means-based approaches and self-organizing feature map (SOM)-based approaches are the examples of unsupervised learning for intrusion detection[20,21].

a) K-means-based approaches:

For intrusion detection, Guan *et al.* [22] propose a K-means-based clustering algorithm, which is named Y –means. Xian *et al.* [23] combine the fuzzy K-means method and a clonal selection algorithm to detect intrusions. Jiang *et al.* [24] use the incremental clustering algorithm that is an extension of the K-means algorithm to detect intrusions.

b) SOM-based approaches:

Hoglund *et al.* [25] extract features that describe network behaviors from audit data, and they use the SOM to detect intrusions. Kayacik *et al.* [21] propose a hierarchical SOM approach for intrusion detection. Specific attention is given to the hierarchical development of abstractions, which is sufficient to permit direct labeling of SOM nodes with connection type. Sarasamma *et al.* [26] propose a hierarchical SOM for intrusion detection. They use the classification capability of the SOM on selected

dimensions of the data set to detect anomalies. Their results are among the best known for intrusion detection [27].

While these existing methods can obtain a high detection rate (DR), they often suffer from a relatively high false positive rate (FPR), which wastes a great deal of manpower. Meanwhile, their computational complexities are also oppressively high, which limits their applications in practice, because an IDS would affect the regular tasks of the target systems if it employs too much resource.

Adaboost is one of the most prevailing machine learning algorithms in recent years. Its computational complexity is generally lower than SOM, ANN and SVM in the case that the size of the data set is voluminous while the dimensionality is not too high. For this and other advantages, we employ Adaboost algorithm for our network-based IDS.

System Architecture

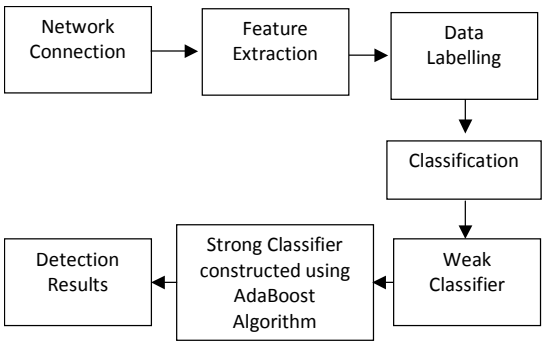

Fig. 1: System Architecture

- *Feature extraction:* For every network connection, we extract three major groups of features for detecting intrusions: "Basic features of individual TCP connections", "Content features within a connection suggested by domain knowledge" and "Traffic features computed using a two-second time window" [28]. The framework for constructing these features can be found in [29].
- *Data labeling:* Because Because the AdaBoost algorithm uses supervised learning, a set of data has to be labeled for training. This labeled data set should contain both normal samples labeled as "+1" and attack samples labeled as "-1." We explain two points: 1) In contrast to misuse detection, which utilizes signatures of attacks to detect intrusions, and anomaly detection, which detects intrusions by modeling normal behaviors, our algorithm models both attacks and normal behaviors to detect intrusions. 2) It is hard to obtain a large amount of labeled data in realistic settings
- *Weak classifiers design:* Adaboost requires a group of weak classifiers designed beforehand. "Weak (or basic)" means that the classifying accuracy of an individual classifier is relatively low.
- *Strong classifiers construction:* A strong classifier is obtained by combining the weak classifiers. The strong classifier has higher classification accuracy than each weak classifier.

After training, a strong classifier is obtained. Then a new network connection represented by the same three groups of features can be sending to the strong classifier and classified as either "normal" or "attack", shown in Figure 1 as detecting result.

Methodology

Weak Classifier Design

A group of weak classifiers has to be prepared as inputs of Adaboost algorithm. They can be linear classifiers, ANNs or other common classifiers. In our algorithm, we select "decision stumps" as weak classifiers due to its simplicity. For every feature f, its value range could be divided into two non overlapping value subsets C_p^f and C_n^f, and the decision stump on f takes the form as follow:

$$h_f(x) = \begin{cases} +1 & x(f) \in C_p^f \\ -1 & x(f) \in C_n^f \end{cases}$$

where *x(f)* indicates the value of *x* on feature *f*.

Algorithm

In the AdaBoost algorithm, weak classifiers are selected iteratively from a number of candidate weak classifiers and are combined linearly to form a strong classifier for classifying the network data.

Let $H = \{\tilde{h_f}\}$ be the set of constructed weak classifiers. Let the set of training sample data be $\{(x_1, y_1),...(x_i, y_i),....,(x_n, y_n)\}$, where x_i denotes the i^{th} feature vector, $y_i \in \{-1,-1\}$ is the label of the feature vector, denoting whether the feature vector represents a normal behavior or not; and n is the size of the data set.

Let $\{w_1,...,w_i,...,w_n\}$ be the sample weights that reflect the importance degrees of the samples and, in statistical terms, represents an estimation of the sample distribution.

The AdaBoost-based algorithm for intrusion detection is described as follows:

Step1) Initialize Weights as:

$$w_i(1) \qquad (i = 1,...,n)$$

$$satisfying \qquad \sum_{i=1}^{n} w_i(1) = 1$$

Step 2) Observe the following for *(t = 1, ……,T).*

a) Let be the sum of the weighted classification errors for the weak classifier hj

$$\varepsilon_j = \sum_{i=1}^{n} w_i(t) I\left[y_i \neq h_j(x_i)\right] \quad (1)$$

Where,

$$I_{[\gamma]} = \begin{cases} 1, & \gamma = true \\ 0, & \gamma = false \end{cases} \quad (2)$$

Choose, from constructed weak classifiers, the weak classifier h(t) that minimizes the sum of the weighted classification errors

$$h\ (t) = \arg \quad \min_{hj \in H} \varepsilon_j \quad (3)$$

b) Calculate the sum of the weighted classification errors for the chosen weak classifier.

c) Let

$$\alpha(t) = \frac{1}{2}\log\left(\frac{1-\varepsilon(t)}{\varepsilon(t)}\right) \quad (4)$$

d) Update the weights by

$$w_i(t+1) = \left(\frac{w_i(t)\exp(-\alpha(t)\, y_i h(t)(x_i))}{Z(t)}\right) \quad (5)$$

where *Z(t)* is a normalization factor

$$Z(t) = \sum_{k=i}^{n} \exp(-\alpha(t)\, y_i h(t)(x_k)) \quad (6)$$

Step 3) The strong classifier is defined by

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha(t) h(t)(x)\right) \quad (7)$$

We explain two points:

- By combining the decision stumps for both categorical and continuous features into a strong classifier, the relations between categorical and continuous features are handled naturally, without any forced conversions between continuous and categorical features.
- The decision stumps minimize the sum of the false-classification rates for normal and attack samples. It is guaranteed that the misclassification rates for the selected weak classifiers are lower than 50% this ensures the convergence of the algorithm[30,31].

EXPERIMENTS

Intrusion Data Set

We utilize the KDD CUP 1999 data set[32] for our experiments. It was originated from MIT's Lincoln Lab and developed for IDS evaluations by DARPA[33]. Despite of several drawbacks mentioned in [28], it has served as a reliable benchmark data set for many researches on network based intrusion detection algorithms. In this data set, each TCP/IP connection has been labeled, and 41 features had been extracted, some of which are continuous and others are categorical. So we don't have to do the task of "Feature extraction" and "Data labeling" shown in Figure 1, then we can focus on the effectiveness and efficiency of the core algorithm of our IDS framework.

There are four general types of attacks appeared in the data set: DOS (denial of service), U2R (user to root), R2L (remote to local) and PROBE. In each of the four, there are many low level types of attacks. Detailed descriptions about the four general types can be found in [33, 34]. The number of samples of various types in the training set is listed in Table 1.

Table 1: Number of Samples in Training Data Set

	Attack				
	DOS	**U2R**	**R2L**	**PROBE**	
Normal	391458	52	1126	4107	**Total**
97278			396743		494021

Experimental Results

First, we run the classical Adaboost algorithm, whose result is shown in Table 2, which shows the performance of Adaboost algorithm with the help of confusion matrix. The overall accuracy with training data set is found to be 99.95%.

Table 3 shows the False Positive Rate and the Detection Rate in training data set with proposed algorithm.

Table 4 shows the comparative accuracy of our proposed algorithm with different intrusion detection models in training data set. The experimental result reveals that our proposed method performs more accurately. From this we can say that the performance of proposed algorithm is much better than others.

Table 2: Performance of Algorithm in Training Data Set (Confusion Matrix)

	Normal	**DOS**	**R2L**	**U2R**	**PROBE**	**%**
Normal	97218	19	9	0	32	**99.93**
DOS	20	391413	3	4	18	**99.98**
R2L	15	0	1102	4	5	**98.04**
U2R	5	0	0	45	2	**88.46**
PROBE	40	11	9	0	4047	**98.53**
%	**99.92**	**99.99**	**98.22**	**85.18**	**98.58**	

Table 3: Detection Rate in Training Data Set

Training Set	
FPR (%)	DR (%)
0.06	99.7

Table 4: Comparison of Accuracy with different IDS Models in Training Data Set

	SVM [36]	**RP [19]**	**SCG [19]**	**OSS [19]**	**SOM [26]**	**Genetic Clustering [37]**	**Bagged C5 [38]**	**RSS-DSS [39]**	**Proposed Algorithm**
Normal	98.42	99.57	99.57	99.64	95.7	79	81.23	89.23	**99.93**
DOS	99.45	97.47	72.01	91.78	92.13	80.12	82.44	90.45	**99.98**
R2L	97.33	95.73	98.57	97.15	20	75.01	84.92	88.79	**98.04**
U2R	64.00	48.00	80.89	16.00	43.33	77.22	76.34	85.23	**88.46**
PROBE	98.57	92.71	85.57	92.71	83.5	72.6	83.23	89.54	**98.53**

CONCLUSIONS

The AdaBoost Algorithm is an extremely powerful mechanism for automatic mathematical characterization of acceptable system activity. In the above paper we have described how we can use AdaBoost algorithm for building an Intrusion Detection System. We have explained the system architecture and the flow diagram for the Adaboost algorithm.

We have constructed an intrusion detection system with Adaboost, a prevailing machine learning algorithm, and described how each part of the whole system works in this paper. An improvement concerning about getting low FPRs and balancing the importance of normal samples and attack samples have been proposed. Hence the performance of the system is improved in training data set. The experiment results have shown that our IDS obtain an extremely low false positive rate with a fairish detection rate.

REFERENCES

1. V. K. Pachghare, Dr. Parag Kulkarni, and Deven Nikam, "Overview of Intrusion Detection Systems", International Journal of Computer Science and Engineering Systems, Vol. 3, No. 3, 265-268, 2009.
2. V. K. Pachghare, Dr. Parag Kulkarni, "Network Security Based On Pattern Matching: An Overview", International Journal of Computer Science and Network Security, Vol. 8, No. 10, 314-318, Oct. 2008.
3. Y. G. Liu, K. F. Chen, X. F. Liao, and W. Zhang, "A genetic clustering method for intrusion detection," Pattern Recognit., vol. 37, no. 5, pp. 927– 942, May 2004.
4. D. Song, M. I. Heywood, and A. N. Zincir-Heywwd, "Training genetic programming on half a million patterns: An example from anomaly detection," IEEE Trans. Evol. Comput., vol. 9, no. 3, pp. 225–239, Jun. 2005.
5. C. Zhang, J. Jiang, and M. Kamel. "Intrusion detection using hierarchical neural networks". Pattern Recognition Letters, In Press, Corrected Proof, Available online, November 2004
6. P. Hong, D. Zhang, and T. Wu. "An intrusion detection method based on rough set and svm algorithm". In Proceedings of International Conference on Communications, Circuits and Systems, volume 2, pages 1127–1130, June 2004.
7. Z. Zhang and H. Shen, "Online training of SVMs for real-time intrusion detection," in Proc. Int. Conf. Adv. Inf. Netw. Appl., 2004, vol. 1, pp. 568–573.
8. S. J. Han and S. B. Cho, "Evolutionary neural networks for anomaly detection based on the behavior of a program," IEEE Trans. Syst., Man, Cybern. B, Cybern., vol. 36, no. 3, pp. 559–570, Jun. 2006.
9. T. Abbes, A. Bouhoula, and M. Rusinowitch, "Protocol analysis in intrusion detection using decision tree," in Proc. Int. Conf. Inf. Technol.: Coding Comput., 2004, vol. 1, pp. 404–408.
10. M. O. Depren, M. Topallar, E. Anarim, and K. Ciliz. Network-based anomaly intrusion detection system using soms. In Proceedings of the IEEE 12th Signal Processing and Communications Applications Conference, pages 76– 79, April 2004.
11. J. Xian, F. Lang, and X. Tang, "A novel intrusion detection method based on clonal selection clustering algorithm," in Proc. Int. Conf. Mach. Learn. Cybern., 2005, vol. 6, pp. 3905–3910.
12. S. Jiang, X. Song, H.Wang, J. Han, and Q. Li, "A clustering-based method for unsupervised intrusion detections," Pattern Recognit. Lett., vol. 27, no. 7, pp. 802–810, May 2006.
13. S. T. Sarasamma, Q. A. Zhu, and J. Huff. "Hierarchical kohonenen net for anomaly detection in network security". IEEE Transactions on Systems, Man and Cybernetics, Part B, 35(2):302–312, April 2005.
14. C. Rudin, I. Daubechies, and R. E. Schapire, "The Dynamics Of Adaboost: Cyclic Behavior And Convergence Of Margins", J. Mach. Learn. Res., vol. 5, pp. 1557–1595, Dec. 2004.
15. S. Mukkamala, A. H. Sung, and A. Abraham, "Intrusion Detection Using An Ensemble Of Intelligent Paradigms", Network and Computer Applications, 28(2):167–182, April 2005.