Proceedings of the Sixth International Conference on Machine Learning and Cybernetics, Hong Kong, 19-22 August 2007

AN UNSUPERVISED INTRUSION DETECTION METHOD COMBINED CLUSTERING WITH CHAOS SIMULATED ANNEALING

LIN NI[1], HONG-YING ZHENG[2]

[1]College of Mechanical Engineering, Chongqing University, Chongqing 400030, China
[2]College of Computer Science, Chongqing University, Chongqing 400030, China
E-MAIL: nilin71@163.com, zhenghongy@263.net

Abstract:

Keeping networks security has never been such an imperative task as today. Threats come from hardware failures, software flaws, tentative probing and malicious attacks. In this paper, a new detection method, Intrusion Detection based on Unsupervised Clustering and Chaos Simulated Annealing algorithm (IDCCSA), is proposed. As a novel optimization technique, chaos has gained much attention and some applications during the past decade. For a given energy or cost function, by following chaotic ergodic orbits, a chaotic dynamic system may eventually reach the global optimum or its good approximation with high probability. To enhance the performance of simulated annealing which is to find a near-optimal partitioning clustering, simulated annealing algorithm is proposed by incorporating chaos. Experiments with KDD cup 1999 show that the simulated annealing combined with chaos can effectively enhance the searching efficiency and greatly improve the detection quality.

Keywords:

Chaos; Intrusion detection; Partitioned clustering; Simulated annealing

1. Introduction

Information technology has been growing with unprecedented speed in the past two decades. Computer networks of all shape and sizes are becoming ubiquitous. Normally, a computer system should provide confidentiality, integrity and assurance. However, due to the increasing connectivity and the vast financial possibilities that are opening up, more and more systems are prone to attack by intruders. These subversion motivations try to exploit flaws in the operating system as well as in application programs. Intrusion Detection Systems (IDS) extract information from a computer or a network of computers, and attempt to detect the presence of intrusions from external sources, as well as system abuses by authorized users [1]. Intrusion detection can be divided into two main categories: misuse detection and anomaly detection. Misuse detection is trying to discover intrusions by searching for distinguishing patterns or signatures of known attacks [5], whereas anomaly detection is based on the assumption that intrusive behavior deviates significantly from previously learned normal behavior [2]-[3].

The previous works in intrusion detection mainly focused on learning knowledge from the labeled data. In [4], intrusion detection based on k-nearest neighbor supervised algorithm is used to classify behavior as normal or intrusive, the k-nearest neighbor is a classical supervised algorithm that finds k examples in training data that are closest to the test example and assigns the most frequent label among these examples to the new example. The only free parameter is the size k of the neighborhood. Besides these, support vector machine is presented in [5]. These algorithms need to be trained with labeled samples. Unfortunately, labels can be extremely difficult or impossible to obtain. Analysis of network traffic or audit logs is very time-consuming and usually only a small portion of the available data can be labeled. Furthermore, in a real application, one can never be sure that a set of available labeled examples covers all possible attacks. If a new attack appears, examples of it may not have been seen in the training data.

To overcome these obstacles, unsupervised intrusion detection methods that do not need any prior knowledge about training data and new attacks have been addressed recently. These detection methods are based on two basic assumptions about data. First, the number of normal instances vastly outnumbers that of anomalies. Second, data instances of the same classification (type of attack or normal) should be close to each other in the feature space under some reasonable metrics, and instances of different classifications are far apart.

In general, clustering is the unsupervised classification of input items into groups (clusters) without any prior knowledge. It is promising to detect unknown attacks in intrusion detection automatically. Furthermore, the clusters

1-4244-0973-X/07/$25.00 ©2007 IEEE

data can be analyzed for more information, i.e. the signature of new attacks. Therefore, to find a near-optimal partitioning instead of local optimizing results, in this paper, a new detection method, IDCCSA, is proposed. As a novel optimization technique, chaos has gained much attention and some applications during the past decade. For a given energy or cost function, by following chaotic ergodic orbits, a chaotic dynamic system may eventually reach the global optimum or its good approximation with high probability. To enhance the performance of simulated annealing which is to find a near-optimal partitioning clustering, simulated annealing algorithm is proposed by incorporating chaos, introducing chaos system using logistic equation completes the perturbation procedure. Through the process, we can find the minimum of the cost function, produce a good result and enhance the detection rate of intrusion.

The remaining parts of this paper are arranged as follows. Section 2 presents the related work of intrusion detection based on unsupervised method. The proposed IDCCSA is described in section 3. The experiment results on KDD cup 1999 data are in section 4. Some conclusions and future work are given in section 5.

2. Related Work

In unsupervised learning, the data are not labeled. The aim of unsupervised learning could be regarded as to fit a generative model that gives a high likelihood to the observed data. From the perspective of machine learning, the searching for clusters is unsupervised learning. To perform clustering is to try to discover the inner nature of the data structure as a whole, and to divide the data into groups of similarity. The γ-algorithm is a recently proposed graph-based outlier detection algorithm in [8]. It assigns to every example the γ-score which is the mean distance to the example's k nearest neighbors. In [9], the author proposes an anomaly detection method, which utilizes a density-based clustering algorithm DBSCAN for modeling the normal behavior of a user's activities in a host. Document [10] presents a new density-based and grid-based clustering algorithm that is suitable for unsupervised anomaly detection. The system can be trained with unlabelled data and is capable of detecting previously unseen attacks. Paper [11] presents a novel anomaly detection and clustering algorithm for the network intrusion detection based on factor analysis and Mahalanobis distance. Factor analysis is used to uncover the latent structure of a set of variables. The Mahalanobis distance is used to determine the similarity of a set of values from an unknown sample to a set of values measured from a collection of known samples. A Genetic SOM Clustering Algorithm (GSOMC) is proposed in [12], by combining SOMs network and genetic algorithm, intrusion detection can be completed. Genetic algorithm is used to train the synaptic weights of SOMs. Paper [13] proposes a novel method for calculating cluster radius threshold and defines the outlier factor of cluster to measure the degree of a cluster deviating from the whole where anomalous classes can be distinguished from normal ones.

Hence it is possible to introduce unsupervised learning into intrusion detection. In this paper, the procedure of intrusion detection is composed of the three parts, that is ,(1) creating clusters from unlabeled training datasets; (2) labeling clusters as 'normal' or 'anomalous'; (3) using the labeled clusters to classify network data. During clustering, chaos simulated annealing algorithm is used to optimize clustering results and obtain near-optimizing results.

3. Methodologies

3.1. Clustering Using Chaos Simulated Annealing

3.1.1. Clustering

Clustering is a process of partitioning data into clusters of similar objects. It is an unsupervised learning process of hidden data. It has a wide application in data mining, document retrieval, image segmentation, and pattern classification. There are two types of clustering algorithm, namely, partitioned clustering and hierarchical clustering. Partitioned clustering divides data sets Q into non-overlapping clusters. In document [14], partitioned clustering is described as an assignment problem as follows. Each cluster has a unique cluster label in $(1,\cdots,K)$, and the vector c assigns a cluster label $c_i \in (1,\cdots,K)$ to the i-*th* object in data set Q. Figure 1 shows an assignment instance.

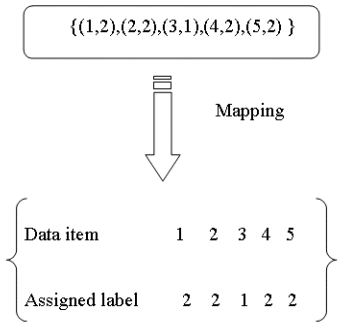

Figure 1. An assignment Instance

In instance, the length of data set is 5,|Q|=5, K=2,

c=(2,2,1,2,2). The vector c shows that the data set is divided into two clusters, the first is composed of items 1,2,4,5 and the other is item 3. For an assignment, we can evaluate the clustering quality by clustering criterion. General clustering algorithms use "squared error" as clustering criterion. For example, squared error based on cluster mean is formulation (1) in k-means [6]-[7]. m_k defines the mean of the k-*th* cluster and $\|\bullet\|$ is Euclidean distance. Besides this, we can use the other clustering criterion that is total within-cluster distance as formulation (2).

$$J = \sum_{k=1}^{K} \sum_{c_i=k} \|\chi_i - m_k\| \quad (1)$$

$$J = \sum_{k=1}^{K} \sum_{\substack{c_i=k \\ c_j=k}} \|\chi_i - \chi_j\| \quad (2)$$

For a given data set, $C=\{c:\forall i\in(1,\cdots,N), c_i\in(1,\cdots,K)\}$ is the set of all feasible clustering, we should find the minimum of c about clustering criterion to enhance clustering quality. Therefore, partitional clustering is converted into the optimizing problem described as the following formulation (3) and (4).

$$Minimize\ J(c) \quad (3)$$

$$Subject\ to\ \ c\in C \quad (4)$$

3.1.2. SA for Partitioned Clustering

Simulated annealing (SA) is a powerful optimization technique that attempts to find a global minimum of a function using concepts borrowed from Statistical Mechanics. The algorithm was originally intended for simulating the evolution of a solid in a heat bath to thermal equilibrium. As it was first described, the algorithm starts with a "substance" composed of many interacting individual molecules arranged in a random fashion. Then, small random perturbations to the structure of the molecules are attempted, and each perturbation is accepted with a probability based on the associated "energy" increase, △E. If △E is at least 0, then the perturbation is accepted with probability exp (-△E/T). If △E is less than 0, then the perturbation is accepted with probability 1. Eventually, after a large number of trial perturbations, the energy settles to equilibrium for the temperature. SA uses both the high- and the low-temperature properties of the Metropolis algorithm to find low energy, regardless of the initial structure. The cooling is made slow to overcome the high dependence of low temperature equilibrium energies on the initial state. Simulated annealing exploits the obvious analogy between process annealing and combinatorial optimization problems, where the "molecules" are the variables in the data structure and the "energy" function is the objective function. Algorithm 1 shows simulated annealing algorithm and table 1 describes the meanings of the notations in algorithm 1.

Procedure SA (δ, MaxIt, T0, α, Tf)
T=T0
REPEAT
FOR I=1 TO MaxIt DO
c'=δ(c)
Δ=J(c') - J(c)
IF (Δ<0) OR (EXP(-Δ/T)>U[0,1]) THEN
c=c'
ENDFOR
T= αT
UNTIL T ≤ Tf

Algorithm 1. Simulated Annealing algorithm

Table 1. THE BASIC FEATURE NAMES

Notations	Meaning
δ	Be the randomized perturbation operator
MaxIt	Be the number of iterations of the Metropolis algorithm
T0	Be the initial temperature
Tf	Be the final temperature
α	α is the attenuation constant for reducing the temperature and α ∈ (0,1)
c	Be the current clustering, c∈C
c'	Be the perturbed clustering, c'∈C
J	Be the clustering criterion
U[0,1)	Be a function that returns a random number between 0 and 1

In this paper, introducing chaos system using logistic equation completes the perturbation procedure. Due to the special characters of the chaotic system such as ergodic property and stochastic property, simulated annealing algorithm based on chaos is more likely to converge to the global optimal solution.

3.1.3. Perturbation Based on Chaos

The following well-known one-dimensional logistic map defined by formulation (5) can produce chaotic system.

$$z^{m+1} = f(\mu, z^m) = \mu z^m (1 - z^m) \quad (5)$$

Where z^m is the value of the variable z at the m-th iteration and in the interval [0,1], μ is a so-called bifurcation parameter of the system, m is the integer number such as 0, 1, 2, 3. Figure 2.shows the chaotic graphs of the map. Where $z^0 = 0.01$, $\mu = 4$, m=200.

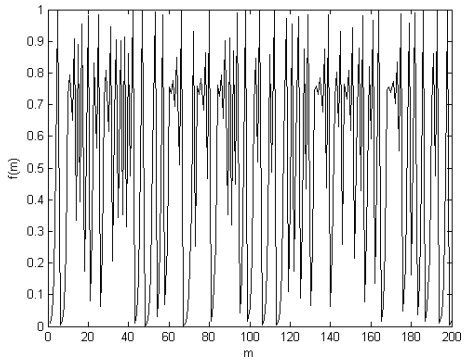

Figure 2. Chaotic graphs of the map

In partitional clustering, the vector c assigns a cluster label $c_i \in (1,\cdots,K)$ to the i-th object in data set Q, the length of K is *l*. we can select randomly I ($I \leqslant |Q|$) data items and perturb them. The procedures of chaotic perturbation can be illustrated as follows:

Step 1: Setting m=0, selecting I labels $c_i^m, i = 1,2,3,...,I,$ from vector c of the data set and mapping these labels to chaotic variable z^m located in the interval (0, 1) using the following equation.

$$z^m = \sum_{i=1}^{I} c_i \times 10^{-i\ell} \tag{6}$$

Step 2: Determining the chaotic variable z^{m+1} for the next iteration using the logistic equation according to formulation (5).

Step 3: Converting the chaotic variables z^{m+1} to labels c_i^{m+1} using repeated the following equations.

$$c_i^{m+1} = round(z^{m+1} \times 10^{i\ell}) \tag{7}$$

$$z^{m+1} = z^{m+1} - c_i^{m+1} \times 10^{-i\ell} \tag{8}$$

Step 4: Mapping the label c_i^{m+1} to the interval (1, K) using the equation (9).

$$c_i^{m+1} = (c_i^{m+1} \bmod k) + 1 \tag{9}$$

Where round ( ) and mod ( ) are function that return an integer and a remainder respectively. For a given vector c0= (2 2 1 2 2) in figure 1, $z^0 = 0.22122$, after perturbation when $\mu = 4$, $z^1 = 0.68912$, c1=(1 1 2 1 2). $z^m \notin \{0.25,0.5,0.75\}$.

We can evaluate the new solution using clustering criterion and accept it or reject it using SA algorithm.

3.2. Labeling Clusters

If data with the same cluster are close together, those with different clusters are far apart, then after clustering we obtain a set of clusters with data of a single type in each of them. Since we are dealing with unlabeled data, therefore, it is necessary to find some way to determine which clusters contain normal data items and which contain attacks.

Let an assumption that normal data items constituting an overwhelmingly large portion of the training dataset be satisfied. We therefore label some percentage N of the clusters containing the largest number of data items associated with them as 'normal', the rest of the clusters are labeled as 'anomalous'. Besides this, there is another problem that should been taken into consideration. That is, there may be many different kinds of normal network activity, this, in turn, might produce a large number of such 'normal' clusters which will have a relatively small number of data items. Therefore, for labeling a cluster, we should calculate the number of data items as well as consider the distance from other clusters. If the number of data items in a cluster is lowest and the distance from the other clusters is largest, we labeled it ‘anomalous’.

3.3. Detection

Once the clusters are created from a training set, the system is ready to perform detection of intrusions. Given a data item d, we should standardize d to d’ and find a cluster which is closest to d’ under the Euclidean distance, i.e. a cluster c in the cluster set C, such that for all C-c, dist (d’, c) $\leqslant$ dist (d’, C-c) .Assign type of cluster c (either normal or anomalous) to data item d.

4. Experiments

4.1. Performance Measures

There are two basic measures, namely, the detection rate and the false positive rate. Detection rate is the percentage of attacks that are detected. False positive rate is the percentage of wrongly classified attacks against the total number of data that are classified as attacks.

$$detection\ rate = \frac{the\ number\ of\ attacks\ detected}{the\ number\ of\ attacks} \tag{10}$$

$$false\ positive\ rate = \frac{the\ number\ of\ false\ positive}{false\ positive + true\ positive} \tag{11}$$

4.2. Data formalization

The KDD Cup 1999 Data [14] is used as the experimental data set. There are total 41 features of each session in KDD Cup 99. Table 3 shows the basic features of KDD Cup 99.

41 features can be divided into 4 categories ('Boolean', 'String', 'Count', 'Rate'). Category 'Boolean' is 0 for 'no' and 1 for 'yes'; category 'Count' is formalized onto [0,1] according to formula (7); category 'Rate' remains unchanged. Category 'String' is mainly used to analyze features of clusters. μ is the j-*th* feature mean and σ is the standard deviation.

TABLE 2 THE BASIC FEATURE NAMES

Feature name	Description	Type
Duration	length (number of seconds) of the connection	continuous
Protocol_type	type of the protocol, e.g. tcp, udp, etc.	discrete
Service	network service on the destination, e.g., http, telnet, etc.	discrete
Src_bytes	number of data bytes from source to destination	continuous
Dst_bytes	number of data bytes from destination to source	continuous
Flag	normal or error status of the connection	discrete
Land	1 if connection is from/to the same host/port; 0 otherwise	discrete
Wrong_fragment	number of ``wrong'' fragments	continuous
Urgent	number of urgent packets	continuous

$$\chi_{ij}^{'} = \frac{\chi_{ij} - \mu}{\sigma} \tag{12}$$

4.3. Detection results

To evaluate our algorithm, we construct a training set and 5 testing sets. Figure 3 shows the data distribution of training set about the two dimensions of count and srv_count, in which "+" and "." denote attacks data items and normal data items labeled after training respectively.

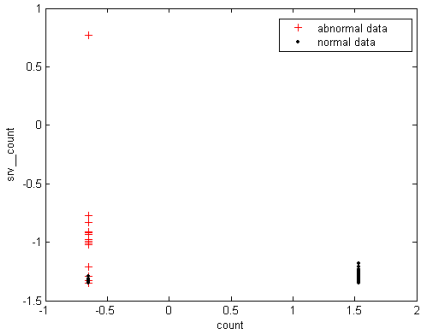

Figure 3. The data distribution with attributes count and srv_count

In order to obtain a better cluster results, the selection about K is important. Several times of training, we let K=30, t_0=300, T_f=1, α=0.95, $M_{ax}I_t$=1000, $\mu = 4$. Figure 4 explain the convergence procedure of J(C) when K=30.

For 5 testing data sets, we calculate their correct labeling rates. The determination of K is very important. Results are showed in figure 5, when K=20, correct labeling rate is obvious higher than that of K=15.

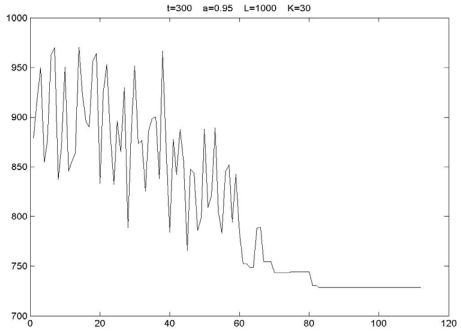

Figure 4. The convergence process of J(c) when K=30

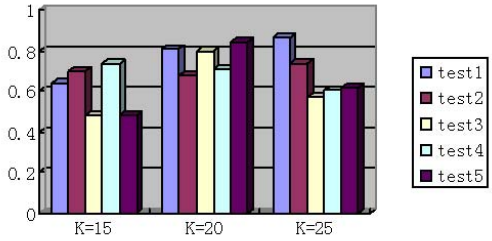

Figure 5. The correct labeling rate for 5 testing sets

To evaluate the algorithm, we are interested in two major indicators of performance: detection rate and false positive rate. In the experiment, the performance of our approach is reported in figure 6 and figure 7.

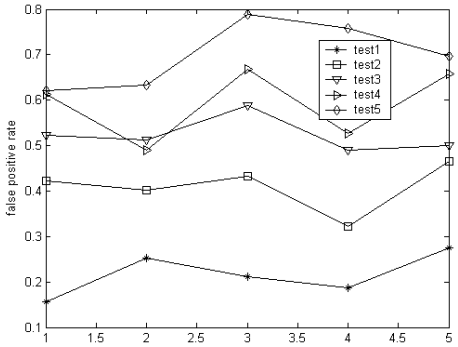

Figure 6. The detection rate of 5 data sets

From experiments, we can find that the performance of algorithm depends on the final value of many arguments, for example, K, MaxIt, α. On the other hand, the construction of training data set is very important. It should show the real distribution of normal data set items and the attack data items.

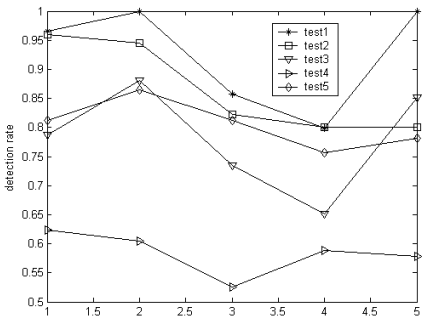

Figure 7. The false positive rates of 5 data sets

5. Conclusion

In this paper, a new detection method, IDCCSA, has been proposed. For a given energy or cost function, by following chaotic ergodic orbits, a chaotic dynamic system may eventually reach the global optimum or its good approximation with high probability. To enhance the performance of simulated annealing which is to find a near-optimal partitioning clustering, simulated annealing algorithm is proposed by incorporating chaos. Simulation results with KDD cup 1999 show that the simulated annealing combined with chaos can effectively enhance the searching efficiency and greatly improve the detection quality.

Acknowledgements

This work was supported by the National High Technology Research Program, the Science & Technology Supporting Program of China(2006BAH02A09) and the Science & Technology Program of Chongqing(2006AB2025)

References

[1] Aurobindo Sundaram. An Introduction to Intrusion Detection [J]. Crossroads, 1996, 2 (4): 3 – 7.

[2] Juan M. Estévez-Tapiador. Measuring normality in http traffic for anomaly-based intrusion detection [J]. Computer networks, 2004, 45: 175-193.

[3] Y.Qiao,X.W.Xin,Y.Bin and S.Ge. Anomaly intrusion detection method based on HMM [J]. Electronics letters, 2002, 38(13): 663-664.

[4] Yihua Liao. Use of k-nearest neighbor classifier for intrusion detection [J]. Computers & security, 2002,21(5): 439-448.

[5] Srinivas Mukkamala, Andrew H. Sung. Identifying Important Features for Intrusion Detection Using Neural Networks [C], Proceedings of the 15th international conference on Computer communication (2002): 1132-1138.

[6] Sanghamitra Bandyopadhyay, Ujjwal Maulik. An evolutionary technique based on K-Means algorithm for optimal clustering in RN[J]. Information Sciences, 2002, 146:221 – 237.

[7] Yiu-Ming Cheung. k*-Means: A new generalized k-means clustering algorithm [J], Pattern Recognition Letters ,2003,24:2883 – 2893.

[8] Harmeling, S., et al. From outliers to prototypes: ordering data [DB/OL]. (http://ida.first.fhg.de/charmeli/ordering.pdf), 2004.

[9] Sang Hyun Oh, et al. An anomaly intrusion detection method by clustering normal user behavior [J]. Computer & security, 2003,22(7):596-612.

[10] Kingsly Leung, et al. Unsupervised Anomaly Detection in Network Intrusion Detection Using Clusters [C]. Proceedings of the Twenty-eighth Australasian conference on Computer Science, 2005,38.

[11] Ningning Wu, Jing Zhang. Factor-analysis based anomaly detection[C]. Proceedings of the 2003 IEEE Workshop on Information Assurance, 2003: 108-115.

[12] Zhenying ma. A Genetic SOM Clustering Algorithm for Intrusion Detection [J]. Lecture notes in computer science, ISNN2005, 3498:421-427.

[13] ShengYi Jiang , Xiaoyu Song,A clustering-based method for unsupervised intrusion detections , Pattern Recognition Letters 2006,27:802 – 810

[14] Donald E. Brown. A Practical Application of Simulated Annealing to Clustering [J]. Pattern Recognition,1992,25(4):401—412.

[15] KDD-99 Cup Data Set. http://kdd.ics.uci.edu/databases/kddcup99/ kddcup99.html