Proceedings of the 4th WSEAS Int. Conf. on Information Security, Communications and Computers, Tenerife, Spain, December 16-18, 2005 (pp293-298)

A Link-level Intrusion Detection Approach

Hongjie Sun, Binxing Fang, Hongli Zhang
National Computer Information Content Security Key Library
Harbin Institute of Technology
Room A502, No. A3, Yumin Road, Chaoyang District, Beijing, 100029
CHINA

Abstract: - The more and more complicated and advanced network attacks greatly thread the security of network. In order to detect and locate the source of anomaly and attack in the beginning of their spreading, we abstract the required link-level properties of network performance using end-to-end measurements and propose a new approach using maximum likelihood estimation and neural network for anomaly link detection and location. Maximum likelihood estimation is used to estimate the distribution of link character, a mixing and optimizing neural network solution combining the BackPropagation (BP) Algorithm with the Simulated Annealing (SA) Algorithm is used for link activity profile learning and anomaly link detection. Comparing with single BP algorithm, the value calculation result shows that BP-SA mixing and optimizing solution has a higher speed and higher accuracy. Experiment results indicate the new approach is effective and of a definite practicability. It is a bran-new idea and has a further develop potential for large scale network anomaly detection.

Key-Words: - Intrusion detection, network security, backpropagation algorithm, simulated annealing algorithm, maximum likelihood estimation, expectation-maximization algorithm

1 Introduction

With the evolvement of the Internet over the last few years, the need for security has been rising with it mainly due to the openness and connectivity nature, and the number of information warfare attacks is increasing, and they are becoming increasingly sophisticated. People and organizations are faced with more challenges every day to secure their data and all other assets of value to them.

There will always be threats and actual intrusions. No system is totally secure, this is due to: (1) The wide variety of hardware and software systems used will complicate the security process, Software will always have bugs that compromise the system security. (2) The expansion of the Internet in the commercial market has forced many companies to offer Internet access to its employees. Along with this connectivity comes the concern of unauthorized use of these computer networks. (3) Networks are under increasing pressure from automated worms. There are many computer virus intentionally enter a computer without the user's permission or knowledge. Though some viruses do little but replicate, others can cause serious damage or effect program and system performance. In 2001, two widely reported Internet worms (Code Red and Nimda) each infected hundreds of thousands of nodes in less than a day and required countless hours to eradicate from systems. (4) Distributed denial-of-service (DDoS) is a rapidly growing problem. Though many DDoS countermeasures have been proposed recently, it is not clear that any one of them is able to stop Internet DDoS attacks in the foreseeable future.

Modern computer and communications networks have evolved into large and complex systems that are decentralized and loosely controlled. As a result, it has become increasingly difficult to monitor and assess their performance. Network monitoring, prediction and diagnosis are very important issues for network operators and designers. However, these are challenging problems due to several factors: (1) direct measurement of packet transport statistics are usually impossible because of internal nodes may not support such diagnostics or these diagnostics may be disabled to minimize overhead; (2) the internal parameters of ISP controlled links are usually inaccessible to outsiders.

Attackers constantly modify their approaches to handle new attacks. The variety of known attacks creates the impression that the problem space is vast, and hard to explore and address. On the other hand, existing defense systems deploy various strategies to counter the problem, and it is difficult to understand their similarities and differences. There is a growing need for intrusion detection and response systems to dynamically adapt to better detect and respond to attacks. Unfortunately, intrusion detection and response systems have not kept up with the increasing threat. In this paper, a link-level intrusion detection approach was proposed. We use maximum likelihood estimation to infer the queuing delay

distributions across internal links in the network based on end-to-end measurements, a mixing and optimizing neural network solution combining the BackPropagation Algorithm (BP) with the Simulated Annealing Algorithm (SA) is used for link activity profile learning and anomaly link detection.

The remainder of the paper is as follows. Section 2 provides a brief overview of some of the literature related to intrusion detection. Section 3 introduces link delay distribution estimation. Section 4 presents the BP-SA algorithm. Section 5 presents the experiments on link-level intrusion detection approach. The last section gives a brief summary of this research.

2 Related Work

The goal of intrusion detection is to monitor network assets to detect anomalous behavior and misuse. Beginning in 1980, With James Anderson's paper, the notion of intrusion detection was born[1]. In 1983, SRI International, and specifically Dr. Dorothy Denning, began working on a government project that launched a new effort into intrusion detection development[2]. Their goal was to analyze audit trails form government mainframe computers and create profiles of users based upon their activities. One year later, Dr. Denning helped to develop the first model for intrusion detection, the Intrusion Detection Expert System (IDES), which provided the foundation for the IDS technology development that was soon to follow[3]. In 1984, SRI also developed a means of tracking and analyzing audit data containing authentication information of users on ARPANET. In 1988, the Haystack project at Lawrence Livermore Labs released another version of intrusion detection for the US Air Force[4]. This project produced an IDS that analyzed audit data by comparing it with defined patterns. In 1989, the developers from the Haystack project for med the commercial company, Haystack Labs, and released the last generation of the technology, Stalker. The Haystack advances, coupled with the work of SRI and Denning, greatly advanced the development of host-based intrusion detection technologies. In 1990, Heberlein was the primary author and developer of Network Security Monitor (NSM), the first network intrusion detection system NSM.

Intrusion Detection is typically divided into two broad categories of misuse detection schemes and anomaly detection schemes. Misuse detection techniques are based on the assumption that, given known patterns of attack, the main advantage of misuse detection systems is that they focus analysis on the audit data and typically produce dew false positives. It is possible to detect when these patterns (or, more importantly, attacks) are occurring. Anomaly detection systems assume that there are patterns of normal behavior for a system. This assumption makes detecting intrusions a matter of comparing the behavior in question to the normal behavior. The main advantage of anomaly detection systems is that they can detect previously unknown attacks. If there is a significant difference, then it is likely that an intrusion is occurring or has occurred. If an event, or sequence of events, occurs that is not predicted by these rules of normal behavior, then it is anomalous and is most likely an intrusion.

Both misuse and anomaly detection schemes are valuable for detecting specific intrusions into a given system, but are not necessarily dynamically suited for today's detection needs. Problems exist for both misuse and anomaly detection systems, especially when only one technique is used exclusively. Misuse detection systems require prior knowledge of the general type of intrusion likely to occur, and these techniques fail to recognize novel attacks. Anomaly detection systems are often susceptible to an intruder slowly training the IDS by gradually varying from normal behavior.

Although intrusion detection has evolved rapidly in the past few years, many important issues remain. First, detection systems must be more effective, detecting a wider range of attacks with fewer false positives. Second, intrusion detection must keep pace with modern networks' increased size, speed, and dynamics. Finally, we need analysis techniques that support the identification of attacks against whole networks. In this paper, we propose a link-level intrusion detection approach. We use active probing to send probes across a network and keeping track of information such as the length of time it takes packets to travel from a root node to the receiver node. This information is then used to estimate internal link-level parameters (specifically delay distributions) from the end-to-end path-level measurements. Then we use BP-SA algorithm to do profile learning and abnormal detection.

3 Link Delay Distribution Estimation

We introduce a new methodology for intrusion detection based on end-to-end delays measurement, specifically, estimating the probability distribution of the queuing delay on each link.

The basic measurement and inference idea is quite straightforward. Suppose packets are sent from the source to many different receivers. The paths to these

receivers traverse a common set of links, but at some point paths diverge (as the tree branches). Packets should experience approximately the same delay on each shared link in their path. This facilitates the estimation of the delays occurring on each link. Under the assumptions that link delays are spatially and temporally independent, we propose a bias collected estimator for the internal link delay based on end-to-end delay measurements.

3.1 Link Delay Model

The network topology is represented as a weighted tree $T=(V,L,D)$ comprising a set of nodes V joined by links in L. D denotes the set of weight (delay, loss, traffic and delay jitter) on each link. A packet source is located at the root node 0, while a set of destinations are located at the leaf nodes R. The interior nodes of the tree represent the branch points of the routing tree from the source to the destinations, and the links L are the logical links that link these branch points.

The intuition behind internal delay estimation is that closely time-spaced packets should experience the same delay on each shared link in their path, and therefore delay aberrations at different receivers must be caused by delays on the individual links. Thus associated with each individual link in the network is a probability mass function. The delay distributions inferring problems can be roughly approximated by the linear model:

$$Y=A\theta+\varepsilon \quad (1)$$

Where Y is a vector of end-to-end delays; A is a routing matrix; θ is a vector of link delay; ε is a noise term which can result from random perturbations of θ about its mean value and possibly also additive noise in the measured data Y. A is a binary matrix (the i,jth element is equal to one or zero) that captures the topology of the network. The problem of large-scale network inference refers to the problem of estimating the network parameters θ given y and either a set of assumptions on the statistical distribution of the noise ε.

We focus on discrete delay distributions, on each link delay falls in the set $\{0,q,2q,\ldots,bq\}$, where q is the unit of measurement and b is an integer that defines the maximum delay for each link. Hence, for a path containing k links, the end-to-end delay takes values in $\{0,q,2q,\ldots,kbq\}$. We will consider inference under the stochastic assumption that the individual link delays X_k are mutually independent. Let $a_k(i)=P\{X_k=iq\}$, $i=1,\ldots,b$ and $k\in V$. In the rest of the paper, for convenience, we will drop the use of the universal measurement unit q. Further, we will denote by $\vec{a}_k=(a_k(1),a_k(2),...,a_k(b))'$, a column vector containing all of the link k delay probabilities. Let $\vec{a}=\{\vec{a}_k,k\in V\}$, a column vector containing all the parameters of interest that have to be estimated. Only the accumulated (end-to-end) delays at the receiver nodes are recorded, we observe only $\vec{Y}=\{Y_j;j\in R\}$. Note that for $j\in R$, $Y_j\in\{0,\ldots,Lb\}$ where L is the number of layers in the tree. Each multicast probe packet experiences a delay on each link along its path. Let $x=\{0,1,...,b\}^{|\varepsilon|}$ be the space of all possible link delays. Hence, $\vec{x}\in X$ is an $|\varepsilon|$-tuple describing the individual link delays that the probe experienced. Let $\vec{y}(\vec{x})$ be the multicast end-to-end measurement that results when $\vec{x}$ occurs. Note that this is a many-to-one function; there are several $\vec{x}$ outcomes that result in the same $\vec{y}$. Denote by $Y=\{\vec{y}(\vec{x});\vec{x}\in X\}$ the space of all possible multicast results.

3.2 Maximum Likelihood Estimation

In this section, we develop the nonparametric MLE of the delay distribution and describe the expectation-maximization (EM) algorithm for computing the MLE.

Let $N_{\vec{y}}$ be the number of probes that resulted in outcome $\vec{y}\in Y$. Let $g(\vec{y};\vec{a})=P\{\vec{Y}=\vec{y}\}$. Then the observed data correspond to a multinomial experiment in terms of the observed end-to-end link delays, and the log-likelihood can be expressed as $l(\vec{a})=\sum_{\vec{y}\in Y}N_{\vec{y}}\log[g(\vec{y};\vec{a})]$. This likelihood is a complicated function and is difficult to maximize directly.

The EM algorithm is a natural approach for computing the MLE in this kind of missing data problem. It is an iterative algorithm that starts with some initial estimate of the desired parameter values. The process is repeated until the likelihood converges to a maximum. Each step of the algorithm is guaranteed to increase the likelihood. Let $M_{\vec{x}}$ be the number of times that a particular individual link delay set occurred. Given an estimate of $\vec{a}$, we can calculate new estimates of the $\vec{a}$ and repeat the process. Formally, let the q-th step estimate of the delay distribution of all the links in the tree topology be denoted by $\vec{a}^{(q)}$. Using this estimate, we can compute $P^{(q)}\{\vec{X}=\vec{x}\}$ and $P^{(q)}\{\vec{Y}=\vec{y}(\vec{x})\}$. With these values, we can now impute the required quantities in the E-step:

$$M_{\vec{x}}^{(q+1)} = N_{\vec{y}} \frac{P^{(q)}\{\vec{X} = \vec{x}\}}{P^{(q)}\{\vec{Y} = \vec{y}(\vec{x})\}} \quad (1)$$

If we let $X_{k,i} = \{\vec{x} \in X \mid x_k = i\}$, then the M-step is :

$$a_k^{(q+1)}(i) = \frac{1}{n} \sum_{\vec{x} \in X_{k,i}} M_{\vec{x}}^{(q+1)} \quad (2)$$

4 BP-SA Algorithm

We use a mixing and optimizing neural network solution BP-SA algorithm to do link activity profile learning and anomaly link detection.

BP algorithm is a relatively perfect feed-forward neural network algorithm in theory, and it is also widely used in practice. It has some drawbacks, the BP algorithm provides a slow training to neural network and is easy to fall into local extremum while the SA algorithm gives a good performance in overall optimization searching.

Simulated annealing is a randomized technique for finding a near-optimal approximate solution of difficult combinatorial optimization problems. The SA algorithm starts with a randomly generate candidated solution. Then, it repeatedly attempts to find a better solution by moving to a neighbour with higher fitness, until it reaches a solution where none of its neighbours have a higher fitness. Such a solution is called locally optimal. In order to avoid getting trapped in poor local optima, simulated annealing strategy occasionally allows for uphill moves to solutions of lower fitness by using a temperature has a high value and then a cooling schedule reduces its value. The new solution is kept if it has a better fitness than the previous solution, otherwise it is accepted with a probability depending on the current temperature. As the temperature becomes cooler, it is less likely that bad solutions are accepted and that good solutions are discarded. In this way it should be possible to avoid getting trapped into local minima early in the execution and to explore the search space in its entirety.

The BP-SA algorithm shows as follows:

(1) Intialize: $N_{input}=b$, $N_{middle}=b/2$, $N_{output}=b$, all $\omega(0)$ to zero, $k=1$, $t=t_1$, $s=s_1$.

(2) BP algorithm
for n-th input：
①forward：for node j of level l

$$y_j^{(l)}(n) = \sum_{i=0}^{T} w_{ji}^{l}(n) y_i^{l-1}(n) ;$$

$$O_j(n) = y_j^{(L)}(n) ;$$

$e_j(n)=x_j(n)-O_j(n)$
②back calculation:

for input nodes:

$$\delta_j^{(l)}(n) = e_j^{(L)}(n) O_j(n)[1 - O_j(n)]$$

for hidden nodes:

$$\delta_j^{(l)}(n) = y_j^{(l)}(n)[1 - y_j^{(l)}(n)] \sum_k \delta_k^{(l+1)}(n) w_{kj}^{(l+1)}(n)$$

③weighted correct:

$$w_{jk}^{(l)}(n+1) = w_{ji}^{(l)}(n) + \eta \delta_j^{(l)}(n) y_i^{l-1}(n)$$

(3) SA algorithm:
①s_j=generate(s), $w'(k)=w'(k)+\beta$, $\beta \in (-1,1)$, $\Delta c=w(k)-w'(k)$;
②compute accept probability
P_k=min[1,exp(-$\Delta c/t_k$)]
③if P_k>random(0,1), then $w(k)=w'(k)$
④$t_{k+1}=vt_k$, v∈(0,1).

(4) repeat(2)(3)，until satisfy the threshold.

For each link k, $a_k^t(i)$ is the link delay=iq probability in window t. $\alpha_k'^t(i)$ is forecast value. $e_{k,i}^t = \alpha_k^t(i) - \alpha_k'^t(i)$， the total error of link k is：

$$e_k^t = \sum_{i=1}^{b} \left| e_{k,i}^t \right| \quad (3)$$

Let θ is threshold, where $e_k^t > \theta$, the link k is taken as an anomaly link.

5 Experimental Results

We used ns2 to perform the network simulation and test the link-based intrusion detection approach. We use a three-layer tree as the network topology in figure 1. Node 0 is root node, and 4, 5, 6, 7, 8 are leaf nodes. We use end node's number of a link as the number of the link, such as link number between node 0 and node 1 is 1. Link 1 has bandwidth

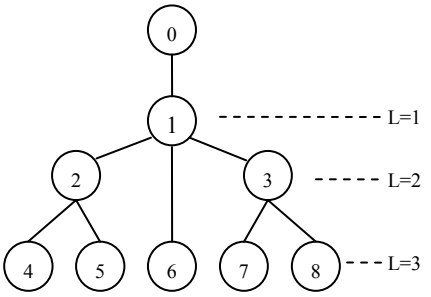

Fig.1 Network topology

5Mb/sec with latency 4ms. Link 2 has bandwidth 4 Mb/sec with latency 5ms. Link 3 has bandwidth 3 Mb/sec with latency 6ms. Link 4, link 6 and link 7 all have bandwidth 2 Mb/sec with latency 8ms. Link 5 and link 8 have bandwidth 3 Mb/sec with latency 7ms. Each link was modeled as a Drop-Tail queue.

We use multicast probing schemes, sending probes across the network according to a Poisson process with mean interarrival time being 0.02s and keeping track of the length of time it takes packets to travel from the root node to the leaf nodes. In order to close to the real IP network environment, we use 200 TCP flows to comprise the exponential on-off background traffic. At t=0s, background traffic was sent, at t=1s, multicast probe packet was sent to do the measurement, while t=80s, an abnormal traffic is simulated from node 8 to node 3, we send a CBR flow, rate is 1000kbps and lasting for 3s; while t=88s, simulating a diffusible behavior, a CBR flow was sent from node 5 to node 2, rate is 1500kbps and lasting for 4s. At t=100s, we close the experiment.

Experiment put on under subscribed case q=1ms, b=10, θ=0.09 and we set window size to be 4s. A set of initial link delay distribution values is given as follows: $\alpha_k(1)$=0.025, $\alpha_k(2)$=0.05, $\alpha_k(3)$=0.075, $\alpha_k(4)$=0.15, $\alpha_k(5)$=0.3, $\alpha_k(6)$=0.15, $\alpha_k(7)$=0.1, $\alpha_k(8)$=0.75, $\alpha_k(9)$=0.05, $\alpha_k(10)$=0.025, $k \in \{1,2,\ldots,8\}$. We assume all links have delay during the experiment, so $\alpha_k(0)$=0.

Figure 4 shows convergence of the estimates of in the first window, with the increasing number of probing packet, the $\vec{\alpha}_8$ convergence to the stable true value. Figure 5 is comparison between $\alpha_8(7)$ and its forecast value, during t=80s~84s, just in the No.20 window, the intrusion traffic leads the delay on link 8 to a large value, the distributing of link delay largely changed on link 8, and $e_8^{20} > \theta$, link 8 is detected as an abnormal link. We can conclude from figure 6, the BP-SA algorithm is more accurate than BP algorithm at forecast the distribution of link delay.

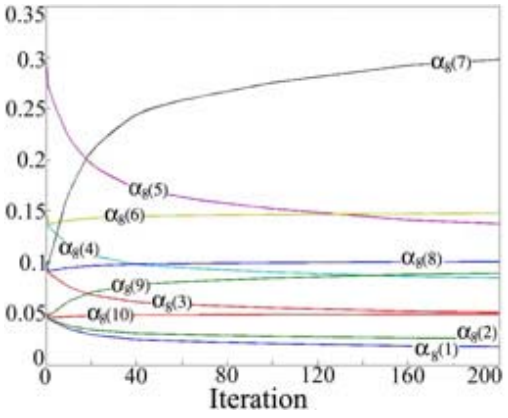

Fig.4 Convergence of the estimates of $\vec{\alpha}_8$

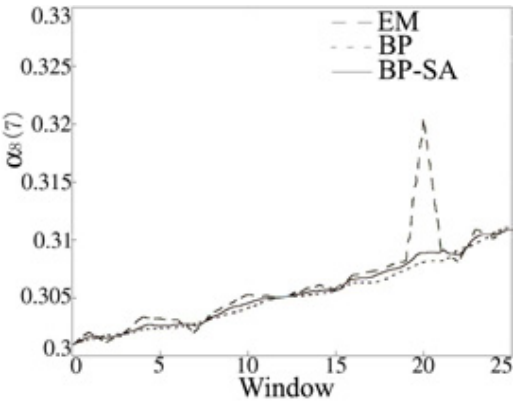

Fig. 5 Comparison between $\alpha_8(7)$ and $\alpha_8'(7)$

Figure 6 shows curve of e_1、e_2、e_3、e_4、e_5、e_6、,e_7、e_8 during the experimental time. We have the conclusion from the figure, link 5 and link 2 take alert the same time in the No.22 window , from the analysis of the topology, within all the links attaching to link 2, only delay distribution on link 5 changes enormously, and we conclude the abnormal traffic comes from link 5.

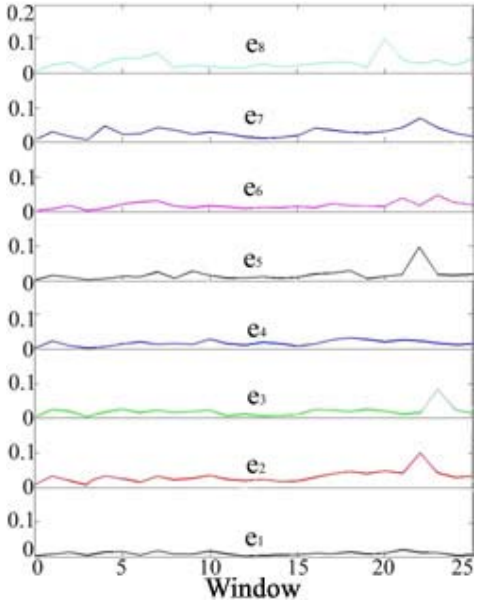

Fig.6 Curve of e_1、e_2、e_3、e_4、e_5、e_6、,e_7、e_8

6 Conclusion

Current intrusion detection systems and intrusion response systems have a limited ability to adapt their detection and response capabilities to the increasingly sophisticated and distributed attacks. In this paper, we have proposed a link-level intrusion detection approach based on end-to-end probing. BP-SA algorithm is used for profile learning and abnormal link detection, maximum likelihood estimation is used for estimate the distribution of delay on each link. Experiment results show that we can detect the abnormal traffic based on link-level intrusion detection approach and we also can locate the source of the abnormal link from the network topology. We also conclude the BP-SA algorithm is more accurate than BP algorithm. The link-based intrusion detection approach is a bran-new idea and would be quite useful for identifying and localizing anomalous behavior in networks.

There are several directions that will be pursued as part of future work. The discrete delay problem addressed here is an approximation to reality, and we plan to study non parametric estimation of underlying continues delay distributions. A more accurate and quickly algorithm for profile learning is also another work for us.

References:

[1]J.P.Anderson, "Computer Security Thread Monitoring and Surveillance", *Technical Report*, James P. Andersong Company, Fort Washington, Pennsylvania,April 1980.

[2]S.Snapp et al, DIDS(Distributed Intrusion Detection System)-Motivation, Architecture and An Early Prototype, *In Proc. 1th National Computer Security Conference*, 1991.

[3]Sunderam, Aurobindo, An Introduction to Intrusion Detection, Downloaded from http://www.cs.prudure.edu/coast/archive/data/categ24.html 1999.

[4] A.Richard,Kemmerer,Giovanni Vigna, Intrusion Detection:A Brief History and Overview, *Reliable Software Group,Computer Science Department*,University of California Santa Barbara, SECURITY& PRIVACY-2002.

[5]Y.liao and V.R. Vemuri, Using text categorization techniques for intrusion detection, *In Proceedings of the 11th USENIX Security Symposium*, August 2002.

[6]CERT Coordination Center, ""Results of the Distributed-Systems Intruder Tools Workshop", Available http://www.cert.org/reports/dsitworkshopfinal.html 2000.

[7] Adriano Cansian,Thiago Siqueira,Cesar Atilio, Dynamic analysis of malicious code: a windows operational system approach, *WSEAS transactions on computer,* Issue 5, Volume 3. November 2004, pp.1442-1450.

[7]C.A.Waldspurger, Memory resource management in VMware ESX Server, *In Proc. of 2002 Symposium on Operating Systems Design Implementation*(OSDI),2002.

[8]A.Wespi, M.Dacier and H. Debar, Intrusion detection using variable length audit trail patterns. *In RAID* 2000, 2000, pp.110-129.

[9]Peng Ning, X. Sean Wang, and Sushil Jajodia, Modeling requests among cooperating intrusion detection systems, *Computer Communications*, 2000, 23(17), pp.1702-1715,

[10]Pend Ning, X. Sean Wang, and Sushil Jajodia, A query facility for common intrusion detection framework, *In proceedings of the 23rd National Information Systems Security Conference*, Baltimore, MD, Oct 2000, pp.317-328.

[11]F.Lo.Presti, N.G.Duffield, J.Horowitz, et al, Multicast-based Inference of Network-Internal Delay Distributions, *IEEE/ ACM Transactions on Networking*, 2002, vol.10, pp.761-775.

[12]R.A.Redner,H.F.Walker, Maximum Likelihood and the EM Algorithm, *SIAM Review,* 1984, 26(2), pp.195-239.

[13]S.David Chen,C J.Ramesh. A Robust Back Propagation Learning Algorithm for Function Approximation. *IEEE Tranx. ON NN*. 1994, 5(3), pp.467-479.

[14]Ingber L, Simulated Annealink, *Practice Versus Theory*, Mathe Comput Modeling, 1993, pp. 29~57.