2011 Seventh International Conference on Computational Intelligence and Security

Fast network traffic anomaly detection based on iteration

Hua Jiang[1], Liaojun Pang[2*]
[1.] *Telecommunication and Engineering Institute,Air Force Engineering University,Xi'an,710077,China*
[2.] *School of Life Sciences and Technology, Xidian University, Xian 710071, China*
E-mail: dzc_djjh@tom.com

***Abstract* - As the Internet environment becomes more and more complex and the network scale expands greater and greater, the unexpected anomaly occurs constantly. Due to lag and subjectivity in building a model and the shortage in terms of speed for traditional detection methods, a fast anomaly detection scheme is proposed in this paper, which is based on research into "self-similar" feature. Estimation results show that our scheme has clear advantage in detection speed and accuracy compared with other schemes based on self-similarity.**

Keywords-Anomaly detection, Network traffic, Fast, Self-similar, Hurst parameter

I. INTRODUCTION

With the development of the Internet technology, the environment of the Internet becomes more and more complex than ever before, and various sudden anomalies and attacks often make detection system unprepared. The fast and real-time of the anomaly detection becomes more and more important, which makes us need a more rapid and accurate detection method to deal with new problems emergence.

Bellcore's monitoring of traffic packets on the LAN (Local Area Network) as well as many research institutions' analysis of the Internet traffic data on the WAN (Wide Area Network)[1][2] show that exact or asymptotical self-similarity model is more suitable for us to describe "burst arrival traffic", and this feature is generally characterized with the Hurst parameter. The traffic load changes will affect the "self-similar" feature[3], and thus affect the value of Hurst. A fast anomaly detection scheme is proposed in this paper, which is based on research into "self-similar" feature.

Through the iterative estimation of Hurst parameter, the traffic anomaly can be determined. At first, we made research on iterative algorithm, and proposed an improved method for the accuracy of the algorithm, and then we validated its advantage in accuracy using fractal Gaussian noise, and analyzed its effectiveness. And then, we made simulation in MATLAB. Using real network traffic data[4][5], we did research on two cases of the normal traffic and the abnormal traffic. The results show that our scheme has clear advantage in the speed and accuracy of detection compared with other schemes based on self-similarity, so it can serve as a new way in fast network anomaly detection.

II. IMPROVED ALGORITHM

If a wide-sense stationary discrete random process $X=\{X_1,X_2,\cdots,X_i\}$, $i=1,2,\cdots,n$ has the feature of "self-similar", then its autocorrelation function $\rho(k)$ meets the following formula when $k\to\infty$:

$$\rho(k)=H(2H-1)k^{2H-2},k\to\infty \quad (1)$$

We transform the above equation and obtain the following formula:

$$H_{i+1}=\sqrt{0.5\times(\rho_k k^{2-2H_i}+H_i)},\quad k\to\infty \quad (2)$$

We can see from above formula that, H can be obtained by self-loop. which i means the numbers of iteration. So we can estimate the current value of H through solving the autocorrelation function of random sequence with fixed sampling interval k and appropriate initial value of H .

We need to set the initial value in calculation, generally use 0.5 as the initial value of H for iteration. To make the calculation easy and fast , we set k to 1, then the following formula can be obtained:

$$H_{i+1}=\sqrt{0.5\times(\rho_k+H_i)},\quad k\to\infty \quad (3)$$

Formula (3) is the original iteration formula of Hurst parameter.

Now we analyze the algorithm again, and we notice a problem in the above algorithm. The established conditions of formula $H=\sqrt{0.5\times(\rho_k k^{2-2H}+H)}$ is $k\to\infty$. That's to say, whether this formula hold depends on the value of k . However iterative algorithm shows that the value of k don't have to take to infinity. We can use simplified iterative formula to solve the Hurst parameter when the value of

* Corresponding author
ljpang@mail.xidian.edu.cn (Liaojun PANG)

978-0-7695-4584-4/11 $26.00 © 2011 IEEE
DOI 10.1109/CIS.2011.225

k is 1. and we notice that k ranges from 1 to infinity , that's to say 1 is the minimum value of k .

So we use the fractal Gaussian noise data with the Hurst parameter are respectively 0.5,0.7,0.8 to test[7], which k ranges from 1 to 7, then we have got the following experiment results:

Table 1 Values of Hurst comparison

	K=1	K=2	K=3	K=4	K=5	K=6	K=7
H=0.5	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	0.5000
H=0.7	0.7213	0.7207	0.7210	0.7006	0.7003	0.7002	0.7001
H=0.8	0.8159	0.8021	0.8008	0.8004	0.8003	0.8001	0.8000

We can see from above data that, when the value of Hurst parameter is 0.5, the estimated values do not change with k .However, when the value of Hurst parameter are 0.7 and 0.8, although the estimates are almost the same, we can figure from the overall trend that, as the value of k increases, the estimation precision is also rising, when the value of k is 7, the estimated value is almost accurate.

Therefore we can increase the value of k in iteration algorithm to make the results more accurate. So the original iterative algorithm changes as follows:

$$H_{i+1} = \sqrt{0.5 \times (\rho_7 \times 7^{2-2H_i} + H_i)} \qquad (4)$$

We notice that, when the value of k is 7, it will increase the index computation and multiplication, which will increase the computational complexity and affect the speed of iteration. We test the iterative speed and compare changes of speed under different value of *k*, experiment results are as follows:

Table 2 iteration time comparison

	K=1	K=2	K=3	K=4	K=5	K=6	K=7
H=0.5	0.0160	0.0160	0.0160	0.0160	0.0160	0.0160	0.0160
	1	1	1	1	1	1	1
H=0.7	0.0260	0.0320	0.0310	0.0470	0.0330	0.0350	0.0400
	4	5	5	6	5	5	6
H=0.8	0.0150	0.0320	0.0.320	0.0240	0.0310	0.0310	0.0260
	4	6	6	5	6	6	5

The above table shows that the number of iterations does not change with k significantly. Similarly, the iteration time does not change much. The iteration time is relatively shorter when the value of k is 1, but the biggest time difference is also less than 0.02s. Compared to the time of wavelet which is 0.245s, this delay can be ignored.

III. FAST NETWORK TRAFFIC ANOMALY DETECTION SCHEME

The flow chart can be shown in figure 1:

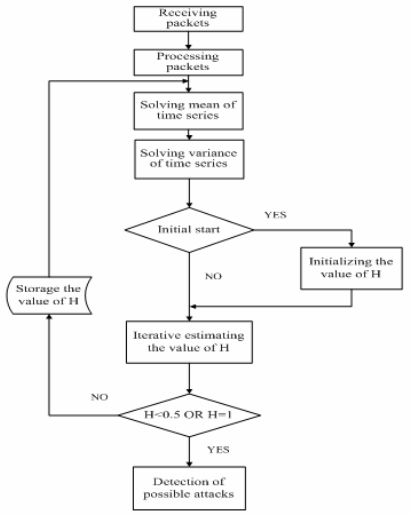

Fig.1. The flow chart of the proposed method

Our scheme can be described in four steps: Receiving Traffic Packets; Processing Traffic Packets; Estimating the Value of H by Iteration; Determining the Network Traffic Anomaly.

Now we will describe these four steps of their principle and ideas of implementation in details.

A. Receiving traffic packets

First we must capture network packets, and here we use Winpcap(windows packet capture) library to achieve this. Winpcap works on Windows platform, and we use it to capture network original packets, collect and count network traffic information. The main algorithms involved are shown as follows:

```
Capture(CardName)
{
//get the NIC list
 pcap_findalldevs();
//open the NIC, and set to promiscuous mode
pcap_open_live();
//establish file for the storage of the captured network packets
pcap_dump_open();
//read the packets information from NIC or files continuously
while(re=pcap_next_ex()>=0)
   {
  // store the captured packets into file
  pcap_dump();
  }
}
```

Here, CardName is a NIC name the user chooses to capture network packets. After a period of time since

starting a thread, it can be seen that packets have been successfully captured, and stored into file.

B. Processing traffic packets

Then we get on processing these packets. According to requirements of this scheme, we will extract the packet arrival time and length information from the captured packets, and then store these information in a linked list for next processing[7-9].

The packet arrival time and length information we extract from the original packets can be stored in the form of structure, specifically defined as follows:

```
struct DATA{
            unsigned int Data_Long;
            time_t Arrive_time;
            };
```

The algorithm for time-series division is shown as follows:

```
void Sequence(CPtrList list,int* counter)
{
 // get the first place of the linked list
 POSITION pos = list.GetHeadPosition();
 // Traverse the linked list
 while(pos)
    {
     DATA* pNode;
     //get a node in the linked list
     pNode = (DATA*)list.GetNext(pos);
     //get the arrival time information
     time_t time = pNode->Arrive_time;
     //determine the time interval.
     double temp = difftime(time,BeginTime);
     int c = temp/STEP;
     counter[c]++;
    }
}
```

Here, STEP represents the time interval, and difftime() is a library function for determining the time interval.

After processing, we can get a group of self-similar sequence $X = \{X_j : j = 1,2,\cdots\}$,which, each random component X_j indicates total number of packets within the j-th time slot. In normal conditions, the sequence has the "self-similar" feature. That is statistical scale invariance.

C. Estimating the value of Hurst by iteration

We have to solve the mean, variance, and autocorrelation coefficient of sequence derived from the last step, and then get the estimate of Hurst using the autocorrelation coefficient.

The flow chart is shown in figure 2

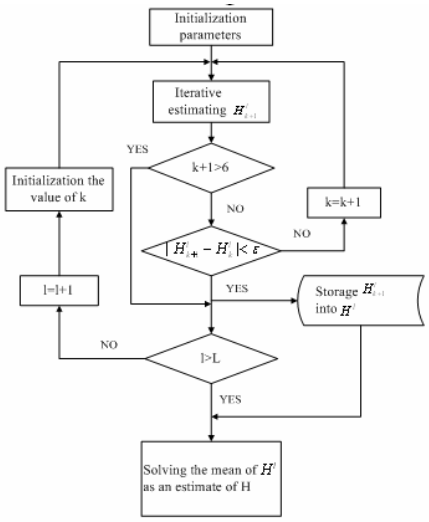

Figure 2 The framework of iterative estimating the value of H

In the algorithm, the role of "flag" is to indicate that whether to continue to solve the exact value of Hurst or not. Which, L indicates the number of iterative solving the value of Hurst. And the final estimate is the mean of L estimates, and this has been done to improve the estimation accuracy.

D. Determining network traffic anomaly

In the normal condition, the value of Hurst ranges from 0.5 to 1. The value of H increases with the increase of the degree of self-similarity. When the attack occurs, the "self-similar" feature decreases or even disappears, and the value of H will tend to 0.5. When extreme burst network traffic occurs, the value of H will jump to 1[10]. Thus we can detect occurrence of anomaly by the changes of Hurst parameter.

IV. SIMULATION AND ANALYSES

A. Data Sources

We need two types of data for experiment. One is the normal network traffic data, and another is the abnormal data. The normal network traffic data can be obtained in two ways:

(1) record the real network traffic data as static data;

(2) simulate the process of FGN or FARIMA according to self-similar theory, and generate static data.

In this experiment, we make use of real network traffic data for accuracy. Data for experiment is measured over Bell Labs 's Ethernet in 1989[11]. There are 4 sets of data in total: BC-pAug89, BC-pOct89, BC-Oct89Ext, BC-Oct89Ext4, and Each group of data

contains 10 million data as shown in table3. We use the fourth set of data as the normal traffic data.

Table 3 information on background traffic

attributes / Data sources	Start Time	Duration Time	Description
BC-pAug89	8/29/1989 11:25	3141.82s	LAN traffic
BC-pOct89	5/10/1989 11:00	1759.62s	LAN traffic
BC-pOct89Ext	3/10/1989 23:46	122797.83s	WAN traffic
BC-pOct89Ext4	10/10/1989 14:00	75943.08s	WAN traffic

Studies show that the characteristics of network traffic changes with the time scale. And 1000 sample points after processing is shows in figure3:

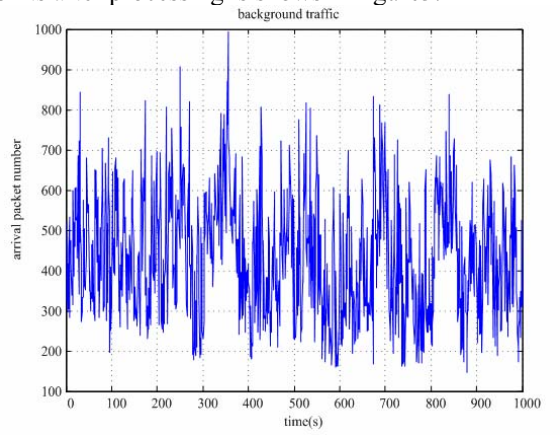

Figure 3 background traffic

B. Simulation and result analysis

The performance indicators include accuracy and computing speed. The experiment result is shown in figure 4:

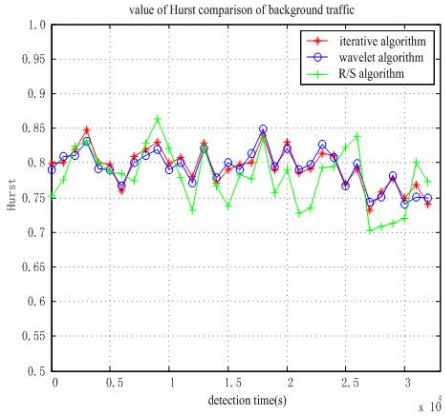

Figure 4 value of Hurst comparison of background traffic

We can notice from figure above that, the accuracy of iteration algorithm is very close to wavelet algorithm, while R/S algorithm's performance is not very good compared with the first two algorithms.

Then, we calculate out used time of these two algorithms for each value of Hurst parameter, and the result is shown in figure 5:

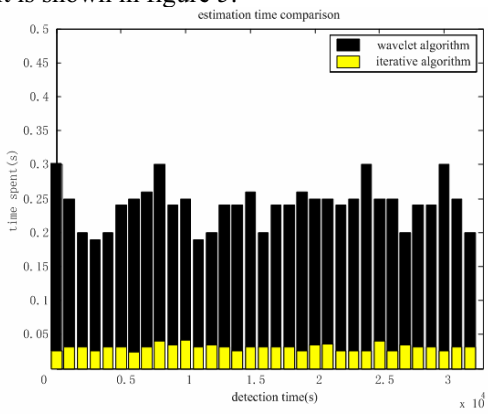

Figure 5 estimation time comparison

Form the figure above, we can see that iteration algorithm is superior to wavelet algorithm in computing speed. The mean of computing time of wavelet is 0.245s, while iteration is 0.0311s. Meanwhile, changes of used time for iteration is relatively stable compared with wavelet.

Above we use three methods to solve values of Hurst parameter in normal network conditions. Now we will add attack flow in the normal network traffic, and do the experiment again. We use network traffic attack generator to generate DOS attack[10], and add it into the normal network traffic. Shown in figure6:

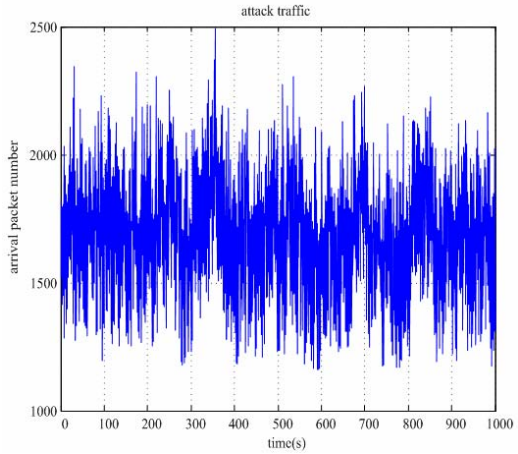

Figure 6 attack traffic

We can see from figure 6 that, arrival packets per second have increased significantly after adding attack traffic.

We compute out 33 values of Hurst parameter again, and the result is shown in figure 7:

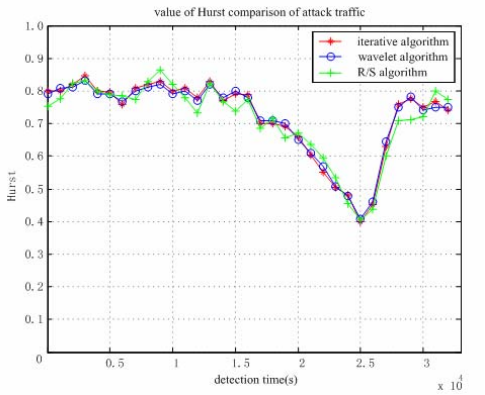

Figure 7 value of Hurst comparison of attack traffic

We can get from above figure that, values of Hurst parameter began to decline when DOS attack was added in the 17000-th sample point, and in the 25000-th sample point the value of Hurst parameter has dropped to about 0.4. After attack, values of Hurst parameter recovered to normal quickly.

V. CONCLUSIONS

In this paper, a fast network traffic anomaly detection scheme is proposed. We start from the fractal characteristics presented by network traffic, and research into the "self-similar" feature. Thus we can improve lag and subjectivity in the model establishment and increase the flexibility of anomaly detection. Apart from that, this scheme is superior to other similar scheme in accuracy and computing speed. In future work, we would like to research on implementation of overall scheme and application development in real network.

ACKNOWLEDGEMENT

This work is supported by the National Natural Science Foundation of China under Grant No. 60803151.

REFERENCES

[1] Leland WE, Taqqu MS, Willinger W, Wilson DV. On the self-similar nature of ethernet Traffic (extended version). IEEE/ACM Trans. on Networking, 1994,2(1):1−15.

[2] Paxson V, Floyd S. Wide area traffic: the failure of Poisson modeling. IEEE/ACM Trans. on Networking, 1995,3(3):226−244.

[3]Dang T D ,Molnar S. On t he Effect s of Non-Staionarity in Long Range Dependent Test s[R] . Budapest Univ Technology and Economics Tech. Rep. Budapest ,Hungary ,1999

[4] Teng, H.S., Chen, K., Lu, S.C.: Adaptive real-time anomaly detection using inductively generated sequential patterns. IEEE Symposium on Security and Privacy (1980) 278–284

[5]Lane, T., Brodley, C.: Approaches to online learning and conceptual drift for user identification in computer security. ECE and the COAST Laboratory Tech. Rep. (Coast TR 98-12), Purdue University (1998)

[6] Li Lin-feng Qiu Zheng-ding An Iterative Method to Estimate Hurst Index of Self-similar Network Traffic. Journal of Electronics & Information Technology, (2006)12-2371-03

[7]Wang Xin, Fang Bin-xing: An Exploratory Development on the Hurst Parameter Variety of Network Traffic Abnormity Signal.

[8]Cannady, J.: Next generation intrusion detection: Autonomous reinforcement learning of network attacks. 23rd National Information Systems Security Conference (2000) 1–12

[9]Hossain, M., Bridges, S.M.: A framework for an adaptive intrusion detection system with data mining. 13th Annual Canadian Information Technology Security Symposium (2001)

[10]Grossglauser M, Bolot J. On the relevance of long-range dependence in network traffic. Computer Communication Review, 1996, 26(4):15 −24.

[11]Adas A. Traffic models in broadband networks. IEEE Communications Magazine, 1997, 35(7): 82−89.