International Journal of Engineering Sciences & Emerging Technologies, Oct. 2014.
ISSN: 22316604 Volume 7, Issue 2, pp: 527-539 ©IJESET

THE RESEARCH OF DISTRIBUTED PROXY CLUSTER MANAGEMENT SYSTEM

Chao Chen[1], Chao Li[1], Fei Gao[1], Yanxiang Yang[2], Chuan Liang[1]
[1]Process Engineering Research Department,
Petrochina Petrochemical Research Institute, BeiJing, China
[2]Strategy and information Department,
Petrochina Petrochemical Research Institute, BeiJing, China

ABSTRACT

Distributed proxy server cluster is used to establish communications across different network segments, and then improve the efficiency of communications, monitor the network traffic effectively. Distributed proxy cluster management system is a management system that is able to monitor system performance, display system control situation, manage system information and analyze historical data. This system consists of three components which are management and monitoring subsystem, testing subsystem and historical data analysis subsystem. Management and monitoring subsystem is used to display the real-time state of cluster, and the information of network, manage black and white list. Testing subsystem is used to execute a stress testing the proxy clusters. Historical data analysis subsystem is used to detect DDoS attacks according to historical data. We also proposed a novel access-based DDoS detecting algorithm with higher accuracy. Based on full enough of experiments, the results show the efficiency and stability of our management system and the DDoS detecting algorithm.

INDEX TERMS: *Cluster monitoring; Network behavior management; DDoS detection.*

I. INTRODUCTION

With the Internet constantly expanding, more and more network terminals in local area network (LAN) access to the Internet through proxy servers. Proxy server is a special web service, which can build an indirect connection between two network terminals. Based on proxy service, terminals can establish communications across different network segments, and then improve the efficiency of communications, monitor the network traffic effectively [1][2]. Proxy services deployed in different segments constitutes a distributed proxy server cluster. This cluster needs a management platform to analyze and regularize LAN user behavior, improve transmission efficiency, prevent leakage of internal confidential information, ensure network safety [3][4]. In this paper, we propose a distributed proxy server cluster management system that is able to manage proxy server cluster, evaluate and analyze network user behavior, measure the stability of the system.

This paper is organized as follows: in the section 2, we briefly introduce related work; In the section 3, we describe the function and architecture of our system; In the section 4, we present the design of our system; In the section 5, we propose a novel access-based DDoS detecting algorithm with higher accuracy. In the section 6, we test the functionality and performance of our system; In the section 6, we makes a summary of this paper.

II. RELATED WORK

Web proxy aims to use the cache to speed up network access, lower the network transmission distance, reduce network link flow pressure [1]. By controlling the user's access, can effectively minimize security risks; achieve the purpose of network security. Security audit is a key technology in

network security[6], the traditional network security audit system[7] is based on IPv4, and it monitors the centralized data, unable to deal with a lot of information. Distributed auditing system[8][9] can split tasks and in turn handle the classified tasks, then improve the efficiency of data processing. In practical cases, however, due to the instability of the network traffic and the distribution of the task has no regularity, if mechanically divide the audit tasks, it will greatly improve the network load. Cooperative collaboration is hot topic that concerned by target tracking[10][11] and network defense[12]. Most systems adopt distributed multi-agent mechanism[13] to achieve cooperative collaboration. Resource synergy of distributed network security audit system[14] not only realized the IPv6 support and coordinated a number of different audit resources, but also improves the audit efficiency and resource utilization. All of the above management platforms lack a visual management and test platform.

III. SYSTEM ARCHITECTURE

The functions that distributed proxy cluster management system provides are shown in figure 1, our distributed proxy server cluster management system consists of the following three subsystems:

A. Management and monitoring subsystem.

The functionality of this subsystem is enumerated as follow.

- Managing black and white list of target LAN;
- Analyzing user behavior;
- Monitoring the dataflow of proxy server and cache service in real time;
- Monitoring the connections of proxy server and cache service in real time;
- Monitoring the I/O and CPU usage of proxy server and cache service in real time.

B. Testing subsystem.

This subsystem is responsible to execute the tests, including function test, stress test, load balancing test, static and dynamic cache test and list management test. In addition, it must be able to display the results of the analysis.

C. Historical data analysis subsystem.

This subsystem is responsible to detect DDoS attack with attack sources and attacked hosts according to historical access data. The detection algorithm employed in this system contains priority algorithm, visited spurt algorithm, IP source clustering algorithm.

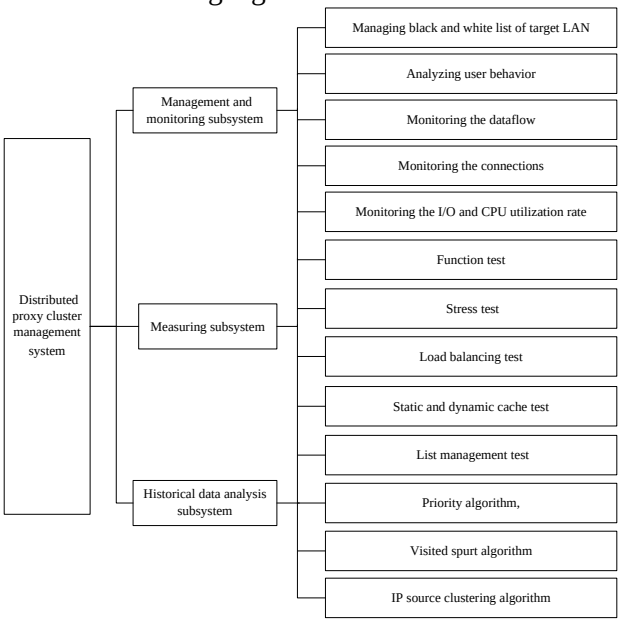

Figure 1. The functions that distributed proxy cluster management system provides

Our distributed proxy cluster management system can be divided into four levels from bottom to top, as shown in Figure 2: (1) Operation platform. This layer is responsible to provide necessary software

environment. Our system can run on most of operating systems, such as Linux, Windows, Mac OS, etc, which have been installed database and server software. (2) Underlying database. This layer is responsible to provide data storage capacity for our system. Specifically, our system employs Mysql database as the underlying storage unit. (3) Data processing and service management. This layer is responsible to preprocess and analyze user behaviors. Furthermore, it has to provide a set of standard methods for administrators to operate related information of user behaviors. (4) User interface. This layer is responsible to provide users with an input interface. In addition, the left side of the structure is used to guarantee the safety of the system from top to bottom.

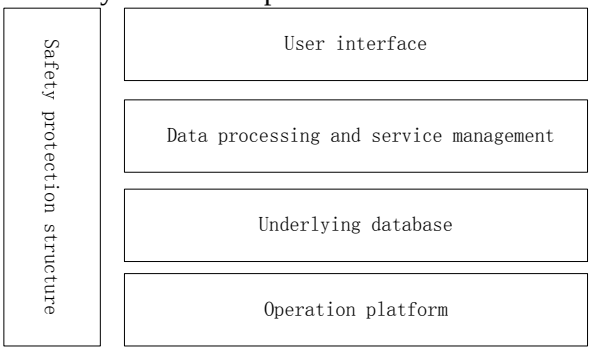

Figure 2. The hierarchical structure of the system

IV. SYSTEM FUNCTION DESIGN

As we described in above section, our system consists of three related subsystems, which are Service management subsystem, measuring subsystem, historical data analysis subsystem. Give a definite function design to each of the three subsystems is the first step of system design. The functions of each subsystem are listed as follow.

A. The functions of service management subsystem

Traffic Monitor: Traffic Monitor measures data traffic consumed by terminals in a cluster. In order to demonstrate traffic efficiently, this subsystem has to provide several additional functions as well, which includes generating flow dynamic figure, counting traffic peak and average, contrasting the traffic in different time periods.

URL statistics: URL statistics analyzes the URLs involved in data traffic consumed by terminals in a cluster, specifically, statistics of URLs accessed recently, statistics of URLs accessed by a terminal during a certain time, statistics of one URL accessed by multiple terminals, classification of URLs accessed in a time period, analysis of URL distributions in the Cache server.

Server monitor: The target of server monitor contains two different types of servers, proxy server and cache server. Several states of these servers are monitored in real time, such as access connections, CPU and memory usage. A proxy server is a server that acts as an intermediary for requests from inside clients seeking resources from other outside servers. A cache server is a dedicated server or service acting as a server that saves Web pages or other Internet content locally. By placing previously requested information in temporary storage, or cache, a cache server both speeds up access to data and reduces demand on an enterprise's bandwidth.

Blacklist and white management: Black and white list management prevents inside clients from accessing to the URLs in the black list, but those in the white list.

B. The functions of testing subsystem

Pressure test: The test program can generate more than ten thousand URL requests in per second, and be able to test the average delay and maximum delay of these requests. Also the test program can check the peak and average number of connections within recent 24 hour, and at the same time by real-time display the change of the number of connections.

Load balancing test: program can dynamic display the operation process of each new user in the system, as well as the distribution of the cache server. Also the program can real-time display the traffic variogram to show the effect of load balancing.

Function test: The URL can be visited when it has not been added to the blacklist, when the URL has been added to the blacklist, it cannot been visited, also the program display the bock time. After using the test tools to request, the program can straightly jump to the monitoring interface, to display the current request is properly executed or not. The program can produce more than 1Gflow data, at the same time, monitor whether the system can run in good condition under 1G flow data.
Static Cache effect test: Through the same data accessing, contrast the access time in the presence of static Cache, to show the superiority of static Cache.
Dynamic Cache effect test: Periodic displays the number of each cache server connections.
List management Test: Add a user into the blacklist, this user can't access any website. Delete a user from the blacklist, these users can access any website. Add a site into the blacklist, users can't visit this site again, delete a site from the blacklist, users can access this website.

C. The functions of historical data analysis subsystem

In view of the historical data in the database, analyze the potential threat, according to the current demand, mainly complete the DDoS attack detection in the potential threat. Through the access to the history data, find the DDoS attack, and display the monitoring results in the form of topology. In next section, we will propose a novel access-based DDoS detecting algorithm with higher accuracy.

V. AN ACCESS-BASED DDOS DETECTING ALGORITHM

A DDoS(Distributed Denial-of-Service) attack is one in which a multitude of compromised systems attack a single target, thereby causing denial of service for users of the targeted system. The flood of incoming messages to the target system essentially forces it to shut down, thereby denying service to the system to legitimate users. DDoS attacks have three common features: (1) The accesses of targeted system surge suddenly; (2) The accesses of targeted system are very short and regularity; (3) The source IP of these accesses is a small amount.
Based on above features, we propose a novel access-based DDoS detecting algorithm based on Traffic Measurement of a cluster. This algorithm consists of three procedures: (1) construct priority queue for suspicious IPs; (2) detect this queue by access surge detecting algorithm to discover suspected attacks; (3) analyze IP sources to distinguish real DDoS attacks from access surge caused by hot issues.

A. Construct priority queue

The priority queue for suspicious IPs is an ordered IP queue that is sorted by suspicious priority in a descending order. Suspicious priority is a value that represents possibility of an IP that launch an attack. The suspicious priority of a given IP can be attained through the following formula (the suspicious priority is denoted as p(n)),

$$p(n) = \alpha \times \text{increase} + \beta \times \text{visit} + \gamma \times p(n-1), \text{ where } \alpha > \beta > \gamma. \quad (1)$$

"increase" is the value of surge, "visit" is the value of accesses, and "p(n-1) is the value of the suspicious priority computed last time. α, β, γ are coefficients whose value means its importance in the formula. Note that we use difference variance of access accumulation instead of absolute count for variable "increase" and "visit" in this formula, because it is easy to record the access accumulation of a server IP. According to the curve of the difference variance of the access accumulation, an access surge of the server IP can be discovered intuitively. Specially, when an obvious raised are appeared in the curve of its difference variance, as shown in Figure 4, it implies there is an access surge of the server IP.

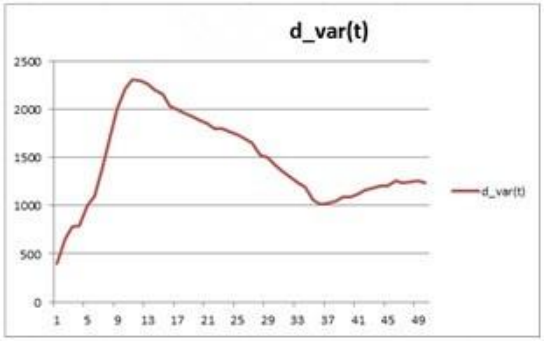

Figure 3. Access surge of the server IP

The processes to construct a priority queue are proposed as follow. Firstly, build a list of server IPs, for each IP in the list we count its access accumulation in real time. Then, found out those access accumulations whose difference variances increase steadily and calculate their access increments to attain suspicious priorities. Finally, insert server IPs into priority queue sorted by their suspicious priorities in ascending order.

B. Access surge detecting Algorithm

Access surge detecting algorithm is responsible to detect access surge of some server IPs that may be caused by real DDoS attacks or access surge of hot issues.

Assume that *v(n)* is the total access accumulation until time period *n*; *c(n)* is the access accumulation in time period *n*, as shown in formula (2). *v_avg(n)* is the average of the total access accumulation, as shown in formula (3). *z(n)* is the difference values between period *n* and *n*-1, as shown in formula (4). *d_var(n)* is the difference variance of *z(n)*,as shown in formula (5). *d_avg(n)* is the average of *z(n)*.

$$c(n) = v(n) - v(n-1) \quad (2)$$

$$v\_avg(n) = \frac{1}{n}((n-1) * v\_mean(n-1) + c(n)) \quad (3)$$

$$z(n) = c(n) - c(n-1), n \geq 1 \quad (4)$$

$$d\_var(n) = \frac{1}{n-1}((n-2) * d\_var(n-1) + z(n)^2) \quad (5)$$

We define function to detect a DDoS attack, as shown in formula (6). This definition borrows the concept of "membership function" in Fuzzy Arithmetic, but much simpler.

$$\mu(n) = \begin{cases} 0, & c(n) < l\_avg(n) \\ \frac{c(n)}{(hv-lv)\times v\_avg(n)} - \frac{lv}{hv-lv}, & c(n) \in [l\_avg(n), h\_avg(n)] \\ 1, & c(n) > h\_avg(n) \end{cases} \quad (6)$$

Where *l_avg(n)=lv* and *h_avg(n)=hv*. In the definition, *l_avg(n)* means the lower limit of "great" access accumulation; if the access accumulation is less than *l_avg(n)*, it can be considered to be "not great", namely the weight of "great" is 0. If *c*(*n*)∈[*l_avg*(*n*), *h_avg*(*n*)], its weight of "great" is defined by function

$$\mu(n) = \frac{c(n)}{(hv-lv)\times v\_avg(n)} - \frac{lv}{hv-lv}.$$

And *h_avg(n)* means the upper limit of "great" access accumulation; if the access accumulation exceeds the global mean by *hv* times, it can be considered to be "very great", namely its weight of "great" is 1. In real applications, the parameters *lv* and *hv* should be assigned according to the performance of networks, For example, according to long-term measured data of the network, and the definition of *lv* may have more significant.

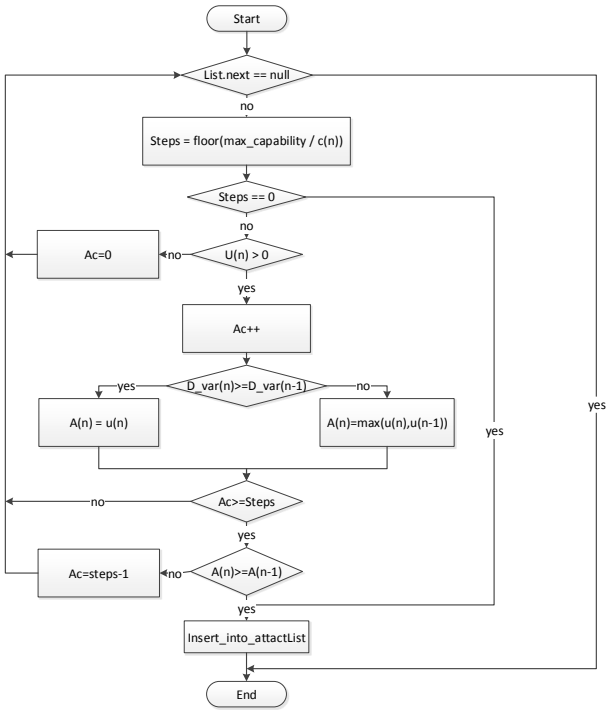

Figure 4 The processes of the access surge detecting algorithm

The processes of the access surge detecting algorithm are described as follow. Firstly, build an array with length n, each item of which is the access count of a server IP in different time periods. If the array is not null, we read each item denoted as $c(n)$ to detect a possible DDoS attack. Then, we suppose that "*max_capability*" is the upper limit of this server capability, "*Steps*" is the upper limit of attack accumulation. If "*Steps*" is zero, namely $c(n)$ is very "great" to reach the upper limit of this server capability, it means there is a suspected attack to this server. Otherwise, we have to identify the trend further to judge whether an attack happens. Specifically, we use inequality $\mu(n)>0$ to detect whether there is a potential attack. If yes, it is necessary to count the number of potential attacks with variable "Ac". After that difference variance in current period is computed and compared with that in last period. According to the comparison, attack possibility is decided which is denoted as A(n). If A(n) is no less than A(n-1) and Ac is no less than Steps, it means this possible attack is showing no sign of abating and last for some time, therefore, we can confirm that there is a suspected attack with target of current server IP.

C. IP behavior analysis Algorithm

IP behavior analysis algorithm is responsible to distinguish real DDoS attacks from access surge caused by hot issues.

We defined a formula to decide whether an access surge is a real DDoS attack, as shown in the following formula.

$$contribution = \frac{VisitCount(originIP,targetIP)}{VisitCount(originIP,allServer)} \quad (7)$$

Where *VisitCount*(*OriginIP*, *targetIP*) is the number of target IPs that accessed by the origin IP, *VisitCount* (*OriginIP*, *allServer*) is the number of servers that accessed by the origin IP.

The value of "*contribution*" represents the possible of a real attack, namely, the higher contribution of an origin IP, the more possible the origin IP launched a DDoS attack.

VI. EXPERIMENTS ON THE ACCESS-BASED DDOS DETECTING ALGORITHM

We compare our Access-Based DDoS Detecting Algorithm with the Flooding-Based DDoS Detecting Algorithm[15]. In order to evaluate the performance of the two algorithms, we build 20 different experimental environments as show in table 1. For each experimental environment, each algorithm

has been executed twenty times and the average value is computed as the final value. In our experiments, the size of a packet is 1KB, therefore, the traffic can be estimated based on the number of records.

Table 1 Experimental environments

The number of records	The number of original IPs	The number of attack IPs	The number of accessed IPs	The number of attacked IPs	Time range of experiment （second）	Time range of attacks （second）
36227	103	3	10	1	0~300	120~180
36227	103	3	10	2	0~300	120~180
34427	102	2	10	1	0~300	120~180
34427	102	2	10	2	0~300	120~180
32627	101	1	10	1	0~300	120~180
32627	101	1	10	2	0~300	120~180
18113	53	3	10	1	0~150	50~80
18113	53	3	10	2	0~150	50~80
17213	52	2	10	1	0~150	50~80
17213	52	2	10	2	0~150	50~80
16313	51	1	10	1	0~150	50~80
16313	51	1	10	2	0~150	50~80

In the first experiment, we measured the accuracy of the two algorithms, the result is demonstrated in the Figure 5. As shown in Figure5, the accuracy of our Access-Based DDoS Detecting Algorithm is much better than that of the Flooding-Based DDoS Detecting Algorithm. The latter fluctuates strongly, since the Flooding-Based DDoS Detecting Algorithm is unable to detect multiple attacked IPs at the same time.

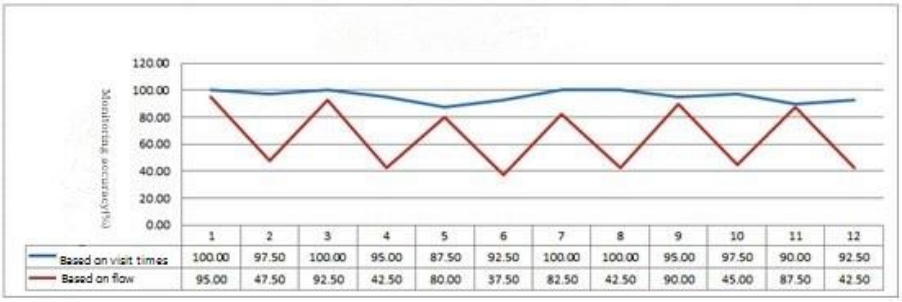

Figure 5 The comparison of the accuracy for the two DDoS detecting algorithms

In the second experiment, we measured the detecting time of the two algorithms, the result is demonstrated in the Figure 6. As shown in Figure6, the detecting time of our Access-Based DDoS Detecting Algorithm is longer than that of the Flooding-Based DDoS Detecting Algorithm, and the former reduced with the number of records. As our algorithm is mainly applied on history records, the detecting time is not the key factor, on the contrary, the accuracy is.

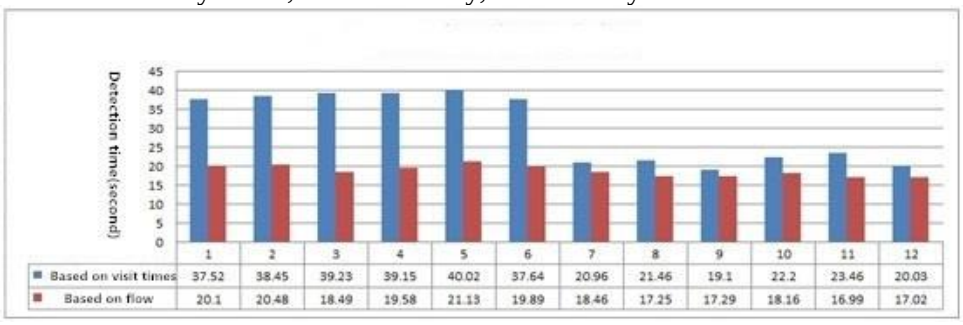

Figure 6 The comparison of the detecting time for the two DDoS detecting algorithms

The comparison of the two algorithms is summarized in Table 2.

Table2 The comparison of the two algorithms

	The Access-Based DDoS Detecting Algorithm	The Flooding-Based DDoS Detecting Algorithm
Detecting time	slow	fast
Real-time	no	yes
Accuracy	high	low
Detect multiple attacked IPs	yes	no
Trace IPs that launch attacks	yes	no

VII. SYSTEM TEST

A. Service management subsystem test

Flow and the number of connections monitoring interface mainly monitors the number of load balancing service connections, flow, as well as calculate and display the flow and the variance of connections number , test cases are shown in table 3

Table 3 Test cases of flow rate and the number of connections monitoring function

The Function description of test object	Real-time display the number of connections and flow, check the effect of load balancing by the variance of flow and the number of connections.	
Use case objective	Test real-time changes of flow and number of connections, the calculation and display of variance.	
The input	Start proxy server, writing data to the load balancing server	Click the sliding window, enter the variance display interface, check the effect of load balancing
The desired output	Real-time display the change effect of flow and number of connections	The sliding window can smoothly slide, display the changes of variance.
Actual result	curve changes Normally	Sliding smoothly, variance shows normally

The test should be executed in the cluster, getting the current monitoring state every 5 seconds, writing to the database, displayed the variance, peak value, the average information that are calculated by the program.

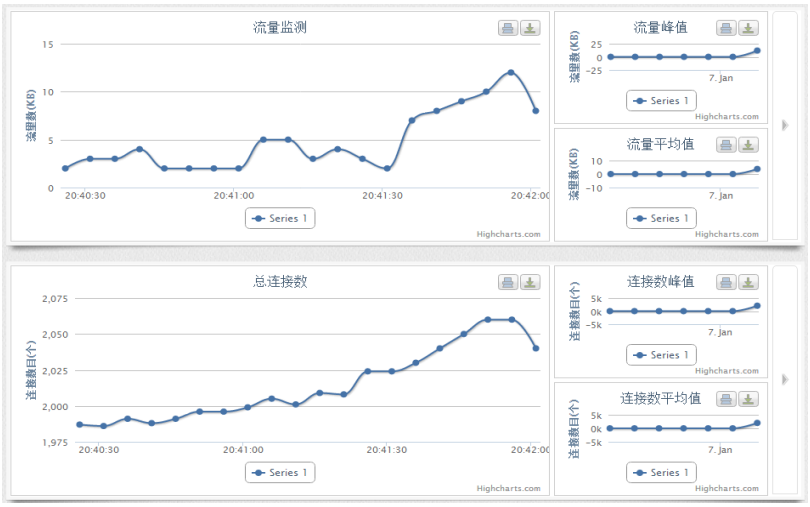

Figure 7 Test results of flow and number of connections

As shown in figure 7, showing the test result of flow and number of connections, the peak within a week cannot display due to the test time is too short.

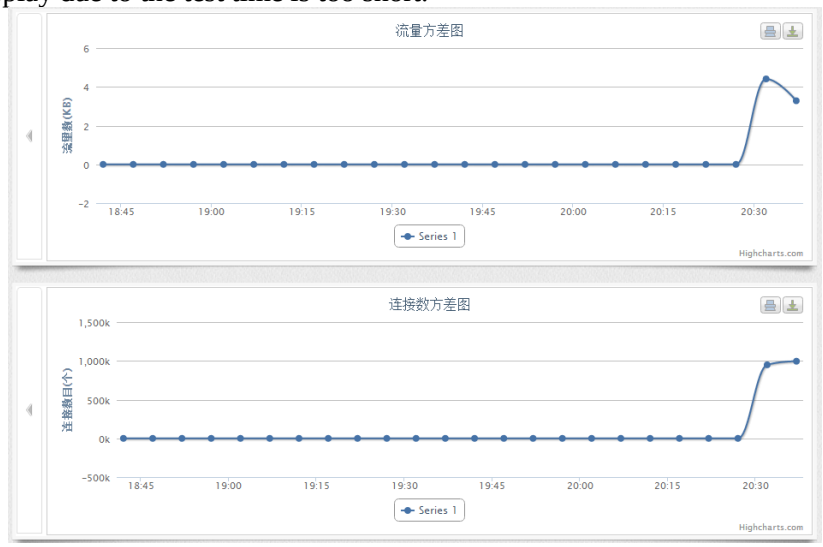

Figure 8 Test results of flow and number of connection variance

As shown in figure 8, shows the test results of flow and number of connection variance, the wave dues to being added a new server, arousing the variance changes a lot.

B. **Testing subsystem test**

Stress test the testing subsystem, stress test is used to test whether the system can work normally when the number of connections is wandering within the scope of a peak, the test case is shown in table 4.

Table 4 Pressure testing of test cases

The Function description of test object	Real-time display the number of connections.
Use case objective	Testing Real-time flow changes, whether reached the critical value when
The input/action	Start proxy server, writing data into the load balancing server every 5 seconds.
The desired output	Real-time display the effect of the number of connections changing.
Actual result	Curve changes Normally

As shown in figure 9, the testing subsystem can display normally even the number of connections is in a very high value, the test has reached the expected results.

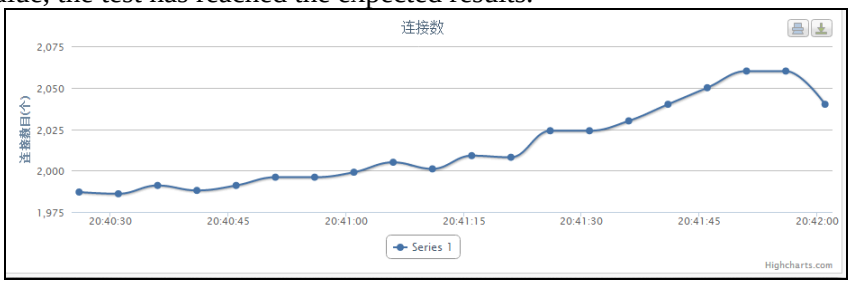

Figure 9 function test results of stress test

C. Historical data analysis subsystem test

Subsystem functions were detected by DDoS attacks, verify the execution efficiency and results of the algorithm, test case is shown in table 5.

Table 5 DDoS attack algorithm test case

The Function description of test object	Discover the DDoS attacks through analyzing the historical accessing data , returns the information of attack sources and attacked server.
Use case objective	Test the execution results and efficiency of the algorithm
The input/action	100 IP access 10 server efficiently 3 IP attacks one of the servers
The desired output	Find 3 attacking IP from the 103 IP Find the attacked server from these server
Test data	Normal user IP: 177.128.228.1 ~ 177.128.228.100, a total of 100 IP Attacking user IP: 133.128.228.1 ~ 133.128.228.3, a total of 3 IP Server IP: 202.118.227.1 ~ 202.118.227.10, a total of 10 IP Normal IP will access one of the server per second Attacking IP will access 202.118.227.530 times per second The total testing time is 300 seconds, the attacking time is from 120 to 180 seconds. The test starts at 2013-05-31 13:40:00
Actual result	The system can discover the attacking IP: 133.128.228.1~133.128.228.3. The system can discover the attacked server:202.118.227.5 The system can discover the attacking time is from 120 to 180 seconds.

The DDoS attacking process is shown in figure 10, during 120 seconds to 180 seconds, this three attacking sources were accessing to the middle of the 202.118.227.5 server 30 times per second. Other accessing source, a total of 100, random access to these 10 servers.

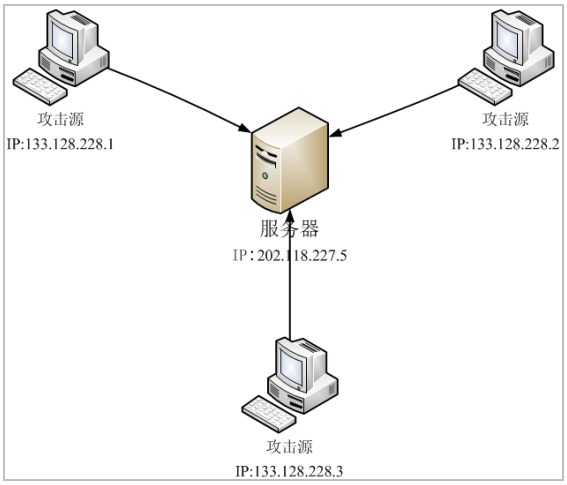

Figure 10 DDoS attack process

Test results are shown in table 4, the test results are accurate.

	Expected outcome	Actual test	Contrastive analysis
Execution time	10 seconds	18 seconds	By 8 seconds
Attack IPs	133.128.228.1 133.128.228.2 133.128.228.3	133.128.228.1 133.128.228.2 133.128.228.3	fully compliance
Attacked server	202.118.227.5	202.118.227.5	fully compliance
Attack Time	2013-05-31 13:42:00 ~2013-05-31 13:42:59 A total of 60 seconds	2013-05-31 13:42:00~2013-05-31 13:42:55 A total of 4 seconds	4 seconds within the error range

VIII. CONCLUSIONS

This paper designed and implemented a distributed proxy cluster management system, used to monitor the user behavior, test the proxy cluster performance, ensure the safety of the proxy cluster. System consists of service management subsystem, testing subsystem, historical data analysis subsystem. In management plateform, different with the traditional detection algorithm which base on traffic surges, DDoS detection algorithms base solely on historical access records. Using priority algorithm, visits suddenly increasing algorithm, IP source clustering algorithm to identify and verify the attack source IP. Furthermore, It can alse detect multiple to be attacked destination IP. In order to detect DDoS attacks, we propose a novel access-based DDoS detecting algorithm with higher accuracy. We verify the effectiveness and stability of the system through abundant test cases.

In the future, we prepare to build a visual Distributed Proxy Service Management platform that employs *Zend Framework2* as a framework, *PHP* for transaction processing, *JSON* formate for transmitting data, *HTML5* for displaying web pages, *CSS3* for controlling style, Javascript and *JQuery* for dynamic process. Meanwhile pages use *HighCharts* and *FusionCharts*, which is the free plugins, to display real-time change maps.

REFERENCES

[1] Wenchao Cui; Changsong Zhao; Ling Zheng; Zhenwei Wang. Application of proxy technology in power system operation and maintenance auditing system. Computer Science and Network Technology (ICCSNT), 2012 2nd International Conference on,On page(s): 973 – 976.

[2] S. Sait, M. Kumar, and H. Murthy. User traffic classification for proxy-server based internet access control. Signal Processing and Communication Systems (ICSPCS), 2012 6th International Conference on, 2012, 1-9.

[3] Z. Duan and Z. Gu. Ewpcc: An elastic web proxy cache cluster basing on cloud computing. Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on, vol. 1. IEEE, 85-88.
[4] Rice, J.L.; Phoha, V.V.;Robinson, P. Using Mussel-Inspired Self-Organization and AccountProxies o Obfuscate Workload Ownership and Placement in Clouds. IEEE Transactions on Information Forensics and Security, 2013, 8(6): 963 – 972.
[5] A. Luotonen. *Web Proxy Servers*. Prentice-Hall, Upper Saddle River, NJ, USA,1998.
[6] L. Fang, W. Zhiying, W. Dan. "Research on the Security Evaluation of Information Systems," Computer Engineering & Science, 2004, 10(78).
[7] Justin Kapp. How to Conduct A Security Audit [EB/OL](2001)[2007-2] . http://www.techsupportalert.com/search/t04123.pdf.
[8] Z. Yehang, D. Guanzhong, Mu Deju. Network Security Study Based on Content Audit and Filter. Application Research of Computers, 2006, 23(10):130-132.
[9] L. Zhifang, F. Xiaoping, X. Yueshan. Research on Building Audit-Online Network Modes. Journal of Computer Application, 2006, 26(4): 977-979.
[10] L. HockBeng, L. VinhThe, F. MaoChing. Adaptive Distributed Resource Allocation in Wireless Sensor Networks [EB/OL]. http://hdl.handle.net/1721.1/30203
[11] F. MaoChing, L. HockBeng, Z. Yulian. Impact of Distributed Resource Allocation in Sensor Networks[C] Intelligent Sensors, Sensor Networks and Information ProcessingConference, Singapore: IEEE, 2005:69-74.
[12] Z. Ran, Q. Depei, C. Heng. Collaboration Intrusion Detection Based on Coordination Agent. Parallel and Distributed Computing, Applications and Technologies. Xian: IEEE, 2003: 175- 179.
[13] Torrellas GAS. Modeling A Network Security Systems Using Multi-Agents Systems Engineering Systems. Man and Cybernetics. IEEE, 2003: 4268-4273.
[14] D.Qian，Z.Chunhua，L. Ying. Network Security Audit System Using Resource Coordination. Journal of Beijing Jiaotong University. 2007, 20.
[15] S. Yi, Y. Xinyu, Z. Huijun A Flooding-Based DoS/DDoS Detecting Algorithm Based on Traffic Measurement and Prediction. Advances in Information and Computer Security Lecture Notes in Computer Science Volume 4266, 2006, pp 252-267.

AUTHORS

Chao Chen was graduated from Harbin Institute of Technology, he has got master degree in software engineering. Now he works as a network engineer in Petrochina Petrochemical Research Institute, proxy cluster management is part of his work.

.

Chao Li was graduated from Heilongjiang University, he has got bachelor degree in petroleum engineering. Now he is a vice director of Process Engineering Research Department, the department has an important responsibility for information construction.

Fei Gao was graduated from Fushun Petroleum Institute, he has got master degree in petroleum engineering. Now he is a director of Process Engineering Research Department, the department has an important responsibility for information construction.

Yanxiang Yang was graduated from Beijing University of Chemical Technology, he has got master degree in engineering. Now he is a director of Strategy and information Department, this department has cooperated with Process Engineering Research Department on proxy cluster management.

Chuan Liang was graduated from Chinese Academy of Agriculture Sciences.He is an engi neer works in Petrochina Petrochemical Research Institute now.He works on normal operati ons of network environment and unified constructional systems.