Statistical Detection for Network Flooding Attacks

C. S. Chao, Y. S. Chen, and A.C. Liu

Dept. of Information Engineering, Feng Chia Univ., Taiwan 407, ROC.

Email: cschao@fcu.edu.tw

Abstract

In order to meet the high demands of network availability and reliability, today's IDS (Intrusion Detecting System) is often constructed in major portal to protect the managed network from attacks. Generally, current IDS use the pattern matching method as the primary scheme to detect attacks. However, after integrating IDS into router, the pattern matching method may cause extreme computing load to router which deviates working correctly, even crashing, when under attacking. In this research, the analysis of network traffic variation is combined with the pattern matching method for decreasing the load arose from the attached IDS of the router. In addition, an overall monitoring system is designed to cooperate with the backbone routers which equipped our attack detecting mechanism. Network managers can not only realize an occurrence of an attack but also where the attack passes through the network by using this system. Besides, the firewall would be also built into the router which can be started by managers to provide first line protection after an attack is detected.

Keywords: abrupt abnormal detection, statistical reference window, network flooding attacks, event correlation, network management

1. Introduction

Due to the explosive growth of network-related technology and generalization of the Internet, the demand of the Internet service quality becomes higher and higher day by day. However, the occurrences of network attacks and virus become the most serious problems at present. Disaster events such as attacks or virus not only cause servers' crash but also bring serious degradation of performance and quality of service in correlated network segments. Therefore, an effective monitor system for adaptive scale of over all networks is essential.

For security, element venders like CISCO and NetScreen incorporate intrusion detection system (IDS) with their products to provide first line protection. By applying such integrated elements, a regional detection and monitoring system can be easily constructed. Current IDS technique can de divided into following types:

- **Pattern matching**

Pattern matching method often applies many rules generated by analyzing each attack. IDS would check every packet with these pre-defined rules to determine whether an attack occurs or not. Most IDSs use pattern matching as main detection method. The most serious problems of pattern matching are its low processing speed and unable to detect novel attack. In order to improve processing performance, Fisk[1] and Coit[2] have individually proposed several new methods in order to speed up pattern comparing. Also, Snort 2.0[3] also uses rule optimizer and rule classifier for the purpose of detecting faster.

- **Abnormal behavior detection**

Abnormal behavior detection is an

unpopular method that detects the abnormal behavior with statistic. This method usually is used in network management for fault detection or prediction. In 1990, Maxion et al.[4] monitored the network traffic in Carnegie Mellon University for about 7 months and used those data to construct a traffic model to predict the traffic amount in next time slot. Then they set an appropriate threshold for the next time slot to check if traffic change abnormally. Hood et al.[5] collected all the logs for fault verifying and used Bayesian network to calculate the probability of fault occurrence in each time slot. Thotton et al.[6] used first order auto-regressive model to calculate white noise series. Generalized likelihood ratio was used to determine the degree of a parameter change. At the end, Thotton et al.[6] used spatial correlation to combine all parameter they chose to check if a fault happened. Unfortunately abnormal behavior detection is hard to verify whether the detected abnormal behavior is really abnormal or not. Therefore, managers need to check log or some other record to verify the abnormal behavior. But abnormal behavior detection is very suitable for doubting an error occurs.

- **Abnormal protocol detection**

Abnormal protocol detection method focuses on observing the structure and content of connection, most abnormal protocol attacks are aimed at connection protocol such as Telnet, HTTP, RPC, SMTP, and Rlogin. If we define the rules for connection protocol in detector, it is very easy to determine the invalid connection traffic such as unexpected data, unnecessary character and invalid character. That's why abnormal protocol attacks can be determined, for instance, CodeRed worm, which tries to use GET command to ask the infected server to execute malevolence program. IDS checks the content of HTTP connection protocol, thus abnormal protocol attacks can be easily determined. But sometimes this kind of attack may pass through pattern matching IDS. Therefore, abnormal protocol detection IDS can be used to complement the insufficient ability of pattern matching IDS to discover the attack behavior violating connection protocol.

- **Visualization**

Hiraishi et al.[7] used "Hyperbolic Tree" to visualized the log data after a user login. Managers can understand all the action and information of each user by observing "Hyperbolic Tree" and can easily discover invalid actions when the branches connect to "Danger" group grow explosively. Erbacher et al.[8] used two basic components to construct a connection graph for a managed host. The first part was an arrow that showed a connection before or after authentication, and the second part was a circle that every arrow directed to. They defined several kinds of arrows for Telnet, FTP, network file system (NFS), initial inetd connection and port scan separately. All the connection in that host can be observed easily by observing the connection graph. If there is an attack to that host, then we can easily discover it by monitoring the change in the graph in visual IDS.

In general, current IDSs use pattern matching method as main detection method. Consequently, in this research, pattern matching is chosen as a main detection method, and is built in router. After integrating IDS into router, pattern matching method may cause router too heavy load to work. In order to decrease the load caused by IDS, we try to combine pattern matching method and change detecting method to detect flooding attack. If the pure pattern matching method causes too heavy load in router,

the combination method of change detecting and pattern matching will be used.

2. Disaster Events Detecting

As described in section 1, disaster events would propagate over several network segments and all the range they pass through would be affected. Therefore, constructing a management system to monitor the entire managed network for understanding the range that attacks affect is very important. For this kind of management system, the detection and reasoning can be from point perspective to line perspective and from line perspective to surface perspective.

- **Line perspective**

When an attack occurs, the flooding traffic would pass through many network elements and cause the value of specific packet statistic to grow explosively. The traffic statistic of each interface of every network element will be monitored. SNMP protocol can easily provide us the information for determining whether the traffic rises abnormally or not. And the category of attack can be reasoned by verifying the packet type of traffic with abrupt change, then pattern matching method would be triggered by system to do further detection.

- **Line perspective**

After detecting the anomaly in an interface, the direction of attack traffic needs to be ascertained. Each attack flow would be transmitted from one element to another element until the traffic reaches the target host. Therefore, the direction of attack traffic can be reasoned by monitoring the incoming and outgoing traffic of each interface in every network element.

- **Surface perspective**

After having all attack traffic direction, we can go a step further to infer the location that the attack traffic is going to affect. Thus, the managers of downstream can be notified to prepare defense works.

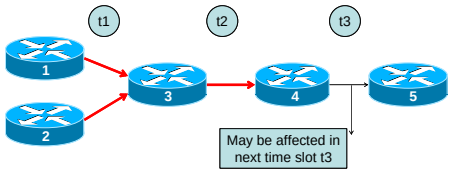

Figure 1. Propagation of attack traffic

As figure 1 shows, the attack traffic passes through link (1->3) and (2->3) in time t1, and gets across element 3 to element 4 in time t2. Thus, we can infer that the downstream of element 4 – link (4->5) will become the next path that attack traffic might pass through. Managers must complete defense works in time to prevent the network segment after element 5 from being affected.

In order to avoid the manager being unable to obtain information because of congestion caused by attack, out-of-bound management architecture represented in figure 2 is adopted. Every managed element should directly connect to manager to form an independent managed domain. No host can transfer data to managed domain besides manager and managed element, that is, only management traffic can exist in that domain. By adjusting the routing table in each router, we can easily create an independent domain for management.

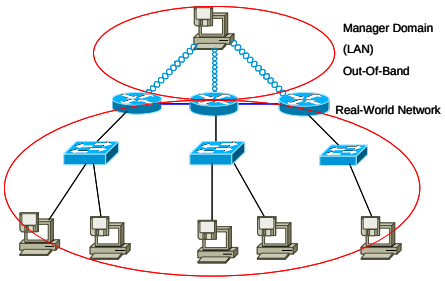

Figure 2. Management architecture

When an attack occurs, managers would receive large amount of alarms. Therefore, managers need some information to recognize these alarms and correlate. In order to differentiate which element and interface these

alarms belong to, we define the alarm format as the figure 3 shows.

Time Stamp	Type	Category	Element & Interface	Connect Element & Interface

Figure 3. Alarm format

Each field is described as follows:

- Timestamp: Recording the time that the alarm is generated;
- Type: Recording the type of alarm, “Send” or ‘Receive;
- Category: Recording the category of attack;
- Element & Interface: the element and interface that the alarm belongs to;
- Connect Element & Interface: Recording the element and interface at the other end of link, this information can help us quickly find out the corresponding alarm if it is generated.

The entire steps of location correlation are listed as follow:

1. Collecting all alarms in time t;
2. Paring these alarms according the connection correlation;
3. Finding out all anomaly links;
4. Classifying anomaly link according to attack type;
5. Mapping anomaly link and direction into topology.

In time consideration, in 2001, Chao et al.[9] have proposed that some faults can be modeled by two or three states finite state machine. A fault can be divided into “Start,” ”Last,” and “Stop” states. In the same year, Cabrera et al.[10] have reported the time line of a DDoS attack as shown in figure 4. As figure 4 shows, an attack start at t3 until the target is shutdown at t5. Thus it can be seen, DDoS attack also can be modeled by three state finite state machine with “START,” “LAST,” and “STOP” states.

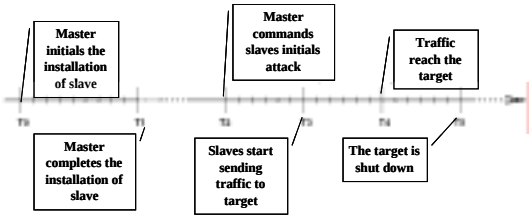

Figure 4. DDoS attack timing outline

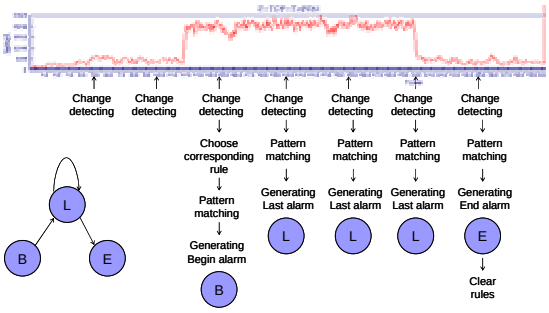

Figure 5. Example of detecting timing outline

Figure 5 is a simple example of detecting disaster events. After detecting an abrupt change, corresponding rules will be chosen to check corresponding packets. For instance, when an abrupt change of TCP traffic is detected, only TCP-related rules will be chosen and only TCP packets will be determined. If an attack is detected, a “Begin” alarms will be generated. In the following time slot, although no change is detected, pattern matching method will still be triggered to check if attack last. If the attack still last, a “Last” alarm will be generated. After another abrupt change is detected and no attack packet is determined, an “End” alarm will be generated and corresponding rule will be cleared.

3. Change Detection

In order to detect the abrupt change accurately, the time series data of each kind of packets would be processed by several change detecting methods. The accuracies of these methods were compared in section 4.

3.1 Series Segmentation

These time series data would be divided into “reference window” and “test window”

respectively before processed by change detecting method. In this research, two different methods were used to decide the reference window size. The first one is “Fixed” and the second one is “Variance”.

- **Fixed window size:**

The size of reference window is fixed in each comparison. Also, the size of reference window is the same as the size of test window. The window size is decided by user. In our experiment, we compare the accuracies of different window size to decide how many samples per window.

- **Variance window size:**

The size of reference window is variable; the time series data of test window is not only compared with the same length data, but also data with double size, triple size and even multiple size of test window. The method we use to decide reference is described as following three steps:

a. Initially, the size of reference window and test window is the same. Suppose the starting point of reference window is time T, the ending point of reference window and starting point of test window are time T + t, and the stopping point of test window is time T + 2t. Time T + t can be looked on as a partition point between reference window and test window.

b. While comparing reference window with test window, if the variation degree is not over the threshold, the partition point will shift to the end of test window. The system would get the next test window. This step repeats until the variation degree is over threshold.

c. When variation degree exceeds threshold, the change point will be set in the partition point. That means an abrupt change is detected in that time. The detection procedure will be restarted at the change point.

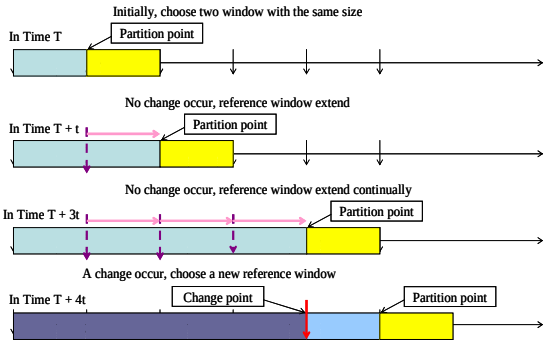

Figure 6. Various Window Sizes

3.2 Detecting Methods

- **Log likelihood ratio test with first order Auto-Regressive model**

In our experiment, the segmentation method proposed by Hood et al.[11] was used to be a change detector. After modeling two adjacent window with AR(1), two white noise series will be generated. The joint probability of that a change is detected l and the joint probability of that no change is detected l_0 can be calculate. The joint probability l is displayed as follows:

$$l = \left(\frac{1}{\sqrt{2\pi\sigma_R^2}}\right)^{N_R}\left(\frac{1}{\sqrt{2\pi\sigma_T^2}}\right)^{N_T}\exp\left(\frac{-N_R\sigma_R^2}{2\sigma_R^2}\right)\exp\left(\frac{-N_T\sigma_T^2}{2\sigma_T^2}\right) \quad \text{Eq. (3.1),}$$

where N_R and N_T are the numbers of random variable of white noise series of reference window and test window individually. The probability of that no change is observed between two windows is:

$$l_0 = \left(\frac{1}{\sqrt{2\pi\sigma_p^2}}\right)^{N_R+N_T}\exp\left(\frac{-(N_R+N_T)\sigma_p^2}{2\sigma_p^2}\right) \quad \text{Eq. (3.2),}$$

where σ_p^2 is the pooled variance. The log likelihood ratio is used to calculate the change degree as follows:

$$\eta = (N_R+N_T)\log\sigma_p - N_R\log\sigma_R - N_T\log\sigma_T \quad \text{Eq. (3.3).}$$

Another two trivial methods, standard derivation comparison and average comparison, are also used in our experiment. The change degree of standard derivation comparison

$$\eta = \frac{(\sigma)_{higher}}{(\sigma)_{lower}} \quad \text{Eq. (3.4),}$$

and the change degree of average comparison

$$\eta = \frac{Average_{higher}}{Average_{lower}} \quad \text{Eq. (3.5).}$$

4. Performance Evaluation

In order to monitor the behavior of attack, we constructed a small network with 3 PC routers, 4 PCs, 2 switches and 1 hub shown in figure 7. The hardware, software and setting we used are listed in table 1:

Table 1. Experimental Environment Description

Hardware	P200 + 64MB Memory	
OS	Redhat linux 7.3	
Routing daemon	Gated	
Routing Protocol	RIP	
Flow calculator	NeTraMet (Flow Meter MIB)	
Background Traffic	Poisson	200,400,600,800,1000 pkts/s
	WWW	40,80,120,160,200 users
Attack	1000,2000 pkts/s	
Window size	5,10,15,20,30 samples	
Threshold	2,3,5,10	

Figure 7 displays our experimental environment. There is bi-direction background traffic in backbone to simulate common traffic. In this experiment, the attack was generated by two attackers. Three change detecting methods described in section 3 were used to detect abrupt change caused by attack. In order to simulate multi-node environment, a multi-process packet generator and DDoS tools were constructed. We use two kinds of background traffic. The first one is Poisson, the inter-arrival rate follows Poisson distributed. The inter-arrival time will be

$$t = -\frac{\log(1-p)}{\lambda} \quad \text{Eq. (4.1),}$$

where λ is average arrival rate to send packet.

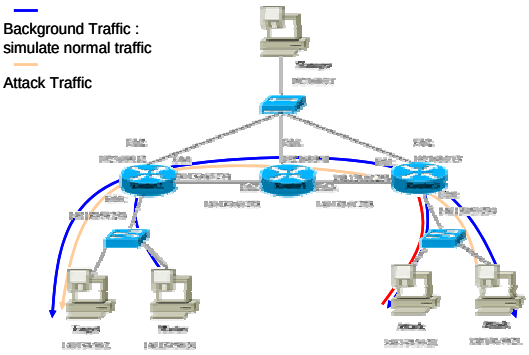

Figure 7. Experimental Environment

Figure 8 is an example of Poisson traffic. As we can see, the traffic seems too stable and unlike real world traffic. Therefore, the second kind background traffic WWW proposed by S. Deng[12] was used and is shown in figure 9.

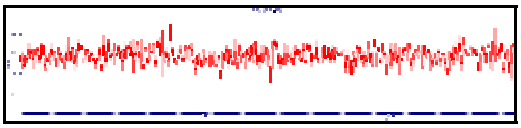

Figure 8. Poisson Traffic

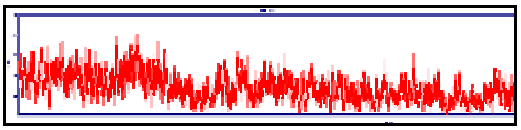

Figure 9. WWW Traffic

WWW traffic is generated by n sources with On and Off state. Weibull distribution was used to calculate the On period time shown in equation 4.2.

$$t = \theta[-\ln(1-p)]^{\frac{1}{k}} \quad \text{Eq. (4.2),}$$

where $\theta = e^{4.5}, k = 0.88$ and the value of p is random. In On period, S. Deng[12] also used Weibull distribution with different parameters to calculate the packet inter-arrival time. The parameters are listed as follows:

$$\theta = 1.5$$
$$k = 0.5$$
$$p : random$$

The Off period time is calculated by using Pareto distribution, the equation and parameters are listed as follows:

$$t = (1-p)^{-\frac{1}{\alpha}},$$

where $\alpha = 0.5$.

In our experiments, the comparisons between the accuracies of every method with different parameter were proceeded. The best result of each method is showed figure 10 and figure 11. For example, in figure 10, "m2-30-5" means using method 2 with 30 samples per window in threshold 5. That is, the window size 30 and threshold 3 are best settings of method 2. Method 1 ~ 6 are described as follows:

- m1: Log likelihood ratio test with AR(1) with fixed reference window size
- m2: Log likelihood ratio test with AR(1) with variance reference window size
- m3: average comparing with fixed reference window size
- m4: average comparing with variance reference window size
- m5: standard derivation comparing with fixed reference window size
- m6: standard derivation comparing with variance reference window size

Figure 10 represents the best result of each detecting method. In this experiment, the attack traffic is 2000 pkts/s, but we find most methods have good accuracies because the difference between attack and background traffic is very large. Therefore, we decrease the attack traffic to 1000 pkts/s and use 1000 pkts/s background traffic. The result is showed in "1000-2". In figure 10, we can see the method 2 and method 5 with window size 30 and threshold 5 are better than others in detecting change caused by attack. The experiment result can only reply to a stable network environment. If the background traffic is non-stable, the result may be different.

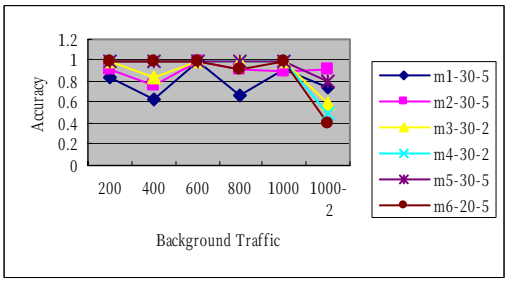

Figure 10. The best result of each detecting method in Poisson background traffic

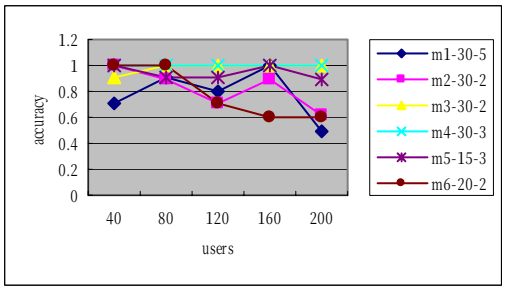

Figure 11. The best result of each detecting method in WWW background traffic

Figure 11 depicts the best result of each detecting method in WWW background traffic. From this experiment, we found the log likelihood ratio test can't provide a good detection of abrupt change with WWW background traffic. As we can see, the accuracies of method 3 and method 4 with 30 samples per test window and method 5 with 15 samples per test window in threshold 3 are higher than 80%. The best one of each method is method 4.

5. Comparison of Detecting Methods

In this section, we showed the comparison between pattern matching method and combination method of change detecting and pattern matching. We would discuss two conditions – normality and under attack. In traditional pattern matching method, the content of each packet would compare with every rule in

IDS after decoding shown as figure 12.

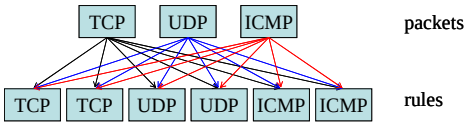

Figure 12. Traditional pattern matching method

Since decoding is an essential step for every method, therefore, we only choose the matching times in the comparison. The total number of matching times of traditional pattern matching method can be written as:

$$T = N\sum_{i=1}^{R} a_i \qquad \text{Eq. (5.1),}$$

where N is total number of packets, **R** is total number of rules and a_i is the number of item of rule i. In order to decrease the frequency of matching, a new pattern matching method classifies all packets before matching. The content of each packet only compares with corresponding rules as figure 13 shows. For instance, TCP packet only compares with TCP-related rules.

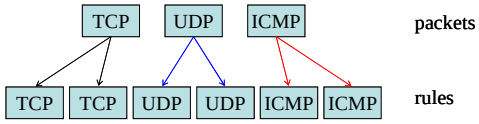

Figure 13. Improved pattern matching method

The number of matching times can be written as:

$$T = N * s + \sum_{i=1}^{x}\sum_{j=1}^{ni}\sum_{k=1}^{Ri} a_k \qquad \text{Eq. (5.2),}$$

where N is total number of packets, s is matching times of packet classification, a_k is the number of item of rule k, Ri is the number of rules of category i, ni is the number of packets of category i, and x is the number of classification.

In equation 5.2, $N*s$ means the total number of matching times in packet classification. $\sum_{k=1}^{Ri} a_k$ indicates the number of matching times of category i for one packet in category i, $\sum_{j=1}^{ni}\sum_{k=1}^{Ri} a_k$ is the number of matching times of category i for all packets in category I and $\sum_{i=1}^{x}\sum_{j=1}^{ni}\sum_{k=1}^{Ri} a_k$ is the number of matching times of category i for every packet. In pattern matching methods, the equation of matching times in normal condition is the same as the number of matching times when the network is under attack, because whether an attack occurs or not, these two methods always check every packet.

The combination method of change detecting and pattern matching only calculates the amount of each category of packet and detect if an abrupt change occurs or not, therefore, matching only is used for packet classification. The number of action after decoding is:

$$T = N(s+1) \qquad \text{Eq. (5.3),}$$

where N is total number of packets, s is matching times of packet classification.

In equation 5.3, the additional 1 is the action of packet counted after classification. And after a change is detected, the pattern matching method is triggered for 5 seconds every 2.5 minutes to check if really an attack has taken place until no attack packet is detected. And only attack-related rules would be chosen as figure 14 shows. For instance, if a change was detected in TCP packet, only TCP-related rules were chosen. Thus, if a UDP or ICMP packet comes, it would be ignored.

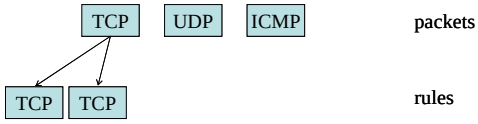

Figure 14. Pattern matching method combined with change detecting method

The number of action can be expressed as:

$$T = N(s+1) + \sum_{i=1}^{x}\sum_{j=1}^{mi}(p_i\sum_{l=1}^{ri} a_l) \qquad \text{Eq. (5.4),}$$

where $\sum_{i=1}^{x} mi = \lambda t$, N is total number of packets, s is matching times of packet classification, a_l is the number of item of rule l, p_i is probability of that a change is detected for packet category i, ri is the number of rules of category i, x is the number of classification, mi is the number of packets of category i, λ is average packet arrival rate, t is time of using pattern matching method.

$(pi\sum_{l=1}^{r_i} a_l)$ is the expected value of the number of matching times for one packet in category i. In this method, the content of each packet only compares with rules corresponding to the type of burst traffic. For example, the content of TCP packet is only compared with TCP-related rules, if category of the current burst traffic is TCP, the TCP-related rules are chosen and the number of matching times is $\sum_{l=1}^{rt} a_l$ (a_t is the number of item of TCP-related rule and r_t is the number of TCP-related rule). But if the category of current burst traffic is UDP or ICMP, and the number of matching times is 0. Hence the exception of number of matching times for a TCP packet is $p_t\sum_{l=1}^{rt} a_l + p_u * 0 + p_i * 0 = p_t\sum_{l=1}^{r_t} a_l$. The number of packet that needs to be compared with rules is λt. In our experiment, t is 5, that's because we only compare with rules for 5 seconds every 2.5monites. So the time the combination method needs to collect packets for comparing with rules is much smaller than the time pattern matching method needs. For each packet, the number of matching times of combination method are less than the number of matching times of new pattern matching method, because pi must <=1, $p_t\sum_{l=1}^{r_t} a_t < \sum_{k=1}^{R_i} a_k$. The amount of packets that should be compared is also less than pattern matching method because the packets collecting time is only 1/30. Thus, total matching times as well as its load will be reduced very much.

6. System Implementation

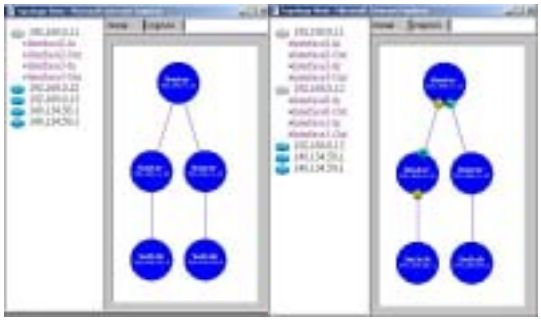

Figure 15. Topology view

In our implementation, the management system can be divided into two parts. One is Monitor constructed in Manager and the other is Detector constructed in PC routers. The topology view of Monitor is showed in figure 15, before Monitor started, a topology description must be constructed first. When monitor is started, topology description would be used to construct a topology list that contains each element and interface showed as tree structure in topology view. The topology list can be clicked to get more detail information about each interface. Monitor is responsible for collecting alarms generated by each element and mapping these alarms into topology after classifying. The symbol R means that abnormal input traffic is detected in this interface and the symbol S, the abnormal output traffic. By observing the topology view in Monitor, we can understand all the condition of managed network.

After clicking interface in topology list, detail information could be grabbed as figure 16 shows. When the amount of packet of flow is over threshold, for example 1/15 of TCP traffic, the flow will be showed in interface view. In this view, the flows of all detected attacks will be

shown first, then followed the flow exceed threshold. Every flow showed in this view will be check with normal rules to determine whether the flow is allowed in the network segment or not. If not, the state column will show “Abnormal”. For every flow, the “Drop” bottom can be click to trigger firewall to drop the packet of that flow. When an attack is detected, the interface that is closest to source can be found from monitor. That flow can be drop from that interface to protect the managed network.

Figure 16. Interface information

7. Conclusion

In this paper, we integrate the IDS into router, and propose management architecture to detect attack in large scale network. In order to avoid effect from the heavy load caused by pattern matching method, we combine change detecting method with pattern matching to release the heavy load of IDS.

The most serious disadvantage of our combination method is change detection. Because the pattern matching method is only triggered after detecting abrupt change, most packets would be ignored. Thus, the detecting method is only useful for detecting flooding attack. In high utilization links, the detecting method won’t detect any abrupt change even an attack occurs because the change degree is not large enough. Therefore, the management system we proposed should be looked on as first line protection. For specific service, host IDS should be adopted to provide advanced protection.

References

1. M. Fisk, and G. Varghese, Fast Content-Based Packet Handling for Intrusion Detection. *UCSD Technical Report CS2001-0670*, May 2001.
2. C.J. Coit, S. Staniford, and J. McAlerney, Towards Faster String Matching for Intrusion Detection or Exceeding the Speed of Snort. *DARPA Information Survivability Conference and Exposition (DISCEX II'01)*, Volume I-Volume 1**,** June 12 - 14, 2001.
3. Snort 2.0, Open Source Network Intrusion Detection System, http://www.snort.org.
4. R.A. Maxion, and F.E. Feather, A case study of Ethernet Anomalies in a Distributed Computing Environment. *IEEE Transactions on Reliability*, Vol 39, No. 4 October, 1990.
5. C.S. Hood, and C. Ji, “Proactive Network Fault Detection”, *INFOCOM '97. Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies*, Volume: 3, 7-12 Apr 1997, 1147 -1155.
6. M. Thottan, and C. Ji, Adaptive Thresholding for Proactive Network Problem Detection. *IEEE International Workshop on Systems Management,* 22-24 April 1998, Page(s): 108 -116.
7. H. Hiraishi, and F. Mizoguchi, Design of a visual browser for network intrusion detection. *Infrastructure for Collaborative Enterprises, 2001. WET ICE 2001*, Page(s): 132 -137.
8. R.F. Erbacher, K.L. Walker, and D.A. Frincke, Intrusion and misuse detection in large-scale systems. *Computer Graphics and Applications, IEEE*, Volume: 22 Issue: 1, Jan/Feb 2002, Page(s): 38 -47.
9. C.S. Chao, D.L. Yang, and A.C. Liu, A time-aware fault diagnosis system in LAN. *Integrated Network Management Proceedings*, 2001, Page(s): 499 -512.
10. J.B.D. Cabrera, L. Lewis, X. Qin, W. Lee, R.K. Prasanth, B. Ravichandran, and R.K. Mehra, Proactive detection of distributed denial of service attacks using MIB traffic variables - a feasibility study. *Integrated Network Management Proceedings* , Page(s): 609 -622.
11. C.S. Hood, and C. Ji, Beyond thresholds: an alternative method for extracting information from network measurements. *Global Telecommunications Conference*, 1997, Volume: 1, 3-8 Nov 1997, Page(s): 487 -491.
12. S. Deng, Empirical Model of WWW Document Arrival at Access Link. *ICC 96, Conference Record, Converging Technologies for Tomorrow's Applications. 1996* , Volume: 3 , 23-27 June 1996.