A Parallel Technique for Improving the Performance of Signature-Based Network Intrusion Detection System

Farzaneh Izak Shiri, Bharanidharan Shanmugam, Norbik Bashah Idris
Advanced Informatic School
Universiti Teknologi Malaysia
Kuala Lumpur, Malaysia
isfarzaneh2@live.utm.my
s.bharani@gmail.com
norbik@utm.my

***Abstract*— Nowadays, organizations discover that it is essential to protect their valuable information and internal resources from unauthorized access like deploying firewall. Firewall could prevent unauthorized access, but it cannot monitor network attacks. Another network security tool such as intrusion detection system is necessary to perform network activities monitoring. With the recent trend of high-speed networks, a large volume of data should be analyzed and processed with high-speed infrastructure. To promote the performance of network intrusion detection system and reduce the processing time of the traffic, present studies on network intrusion detection system for high-speed network focus on parallel techniques as an alternative. In this paper, a kind of parallelism is proposed to improve the performance of signature based intrusion detection system. The experimental results show that by the use of two signature based network intrusion detection systems running Snort in parallel with a portion of packets and a subset of rules, and distributing the traffic between them, the processing time of the traffic will be reduced. Consequently, the performance of the system will be improved.**

Keywords-Network Intrusion Detection System;Parallelism; nort;Function Paralle ;Signature-based

I. INTRODUCTION

During the *late 1960s,* some government computer scientists in the Advanced Research Projects Agency made a decision to design a network that would connect military bases to other military agencies. ARPANET designed to operate in a very small community in which members were trusted, the number of account holders was small, and many users of the network knew each other. With regard to the rapid growth of the internet, it is obvious that those days of the early internet are long gone. Many businesses and government organizations use the internet as a means of providing public access to public records and information [1].

Undoubtedly, in this network world, the needs for security and proper systems of control are obvious. The area of audit and intrusion detection has become an important part of computer and network security. To act appropriately against the attacks, currently security solutions are relying on the intrusion detection system. Allen .et al. [2] defined Intrusion Detection as the process of identifying unauthorized use, misuse, and abuse of the computer system. ID shows the 'Intrusion Detection' that is the possibility of finding the incorrect or inappropriate action. Intrusion detection is positioned as one of the consistent security measures against an incident. There are incident parts include threat, incident, occurrence of damage, and recovery. Security measures include detection, prevention, correction, deception, reduction, reaction and evaluation.

Every system include software or hardware , which is responsible for monitoring all the activities within the system or network in order to detect malicious activity, and making reports to the management system are named Intrusion detection systems (IDS) , also the aim of intrusion detection system (IDS) is to monitor network infrastructure for finding anomalous behavior and misuse. Such a goal has been recognized as significant for many years, but a huge grow in incorporation and popularity over the network asset has been happened recently.

Some kind of IDSs capture packets form network backbone to find attacks and some others analyze an information source that is created by the OS or maybe applications to find malicious activity. This paper presents efficient network intrusion detection system architecture. Section 2 introduces network intrusion detection system and its two major analysis methods. Section 3 shows the related work. Section 4 presents our proposed architecture. In Section 5 the experimental results are detailed and the paper is concluded by conclusion and future work.

II. NETWROK INTRUSION DETECTION SYSTEM

Network-Based intrusion detection systems (NIDS) are responsible for monitoring traffic on packet level. It can find an attack in the way that capture the packets and then analyze them. One or several sensors which are placed in the network for detecting malicious activities can be determined as the network intrusion detection systems. They are responsible for monitoring the traffic and perform analysis of the packets and make report to a central unit for further investigation [3]. There are two approaches that can carry out NIDS, first is anomaly detection and second is misuse detection.

A. Anomaly Detection

It is known as "behavior- based detection "that can be described as network intrusion detection system which models the normal behavior of the network, users, computer systems and raises an alarm whenever there is a deviation

978-1-61284-486-2/11/$26.00 ©2011 IEEE

from this normal behavior. Anomaly detection is able to detect any new attacks or any new potential attacks, but the problem of accuracy is still open to research. It generates a lot of false positive alarms.

Anomaly based NIDS collect a variety of statistics from the audit data and hold them into a profile that reflect legitimate actions of a user, host, application, or even network connection. They are built from all the information that capture during a period of time. Then, the analyst uses a variety of algorithms and techniques to decide when activities differ from the norm.

B. *Signature Based detection*

Signature detection (misuse detection) that has been used for detecting known attack; has the higher level of security than anomaly detection, but the major problem of signature-based NIDS is that every signature should have an entry in the database in order to compare with the arrived packets; therefore the process will be time-consuming and will slow down the throughput of the NIDS.

The majority of main commercial intrusion detection systems primarily use signature-based NIDS, so the progress of signature based intrusion detection system should continue. One of the prominent factors to achieve this goal is improving the performance of signature based intrusion detection systems in order to able process more traffic in less time.

III. Related work

One of the most important weaknesses of network intrusion detection system is that processing the whole traffic is so time-consuming, so as network speeds continue to increase, it is crucial that efficient approaches are developed until intrusion detection systems can process more traffic in less time. To solve this problem several nodes can be used to process the network traffic concurrently and in parallel, so each node is only responsible for processing one part of the traffic, hence by arranging several sensors to deal with the traffic in parallel, the intrusion detection system's speed can be significantly increased.

Culler and Singh [4] defined two types of parallelism, data parallelism and function parallelism. The first one refers to division of data to several parts that should be processed across processing sensors. The second; function parallelism, describes the occurrence of completely distinct calculations that may be performed concurrently on either the same or different data.

There are several efforts on the improving the performance of the network intrusion detection system such as early filtering, where a portion of packets are processed on the splitter instead of the sensors [5], locality buffering to increase the system performance [6], and using cluster in order to allow tasks to be executed in parallel in the cluster [7], while the others study distributed NIDS architecture and using parallelism at the sub-component level [8] [9]. With regard to use a kind of parallelism in this research, we are going to describe briefly the levels of parallelism which Wheeler [10] categorized them into node level, component level, and sub-component level.

A. *Node level parallelism*

At node level, there are several nodes (systems) that packets are transferred to these nodes by the use of traffic duplicator or maybe a traffic splitter, and each node operates in data or function parallel.

At node level data parallelism, each node should have complete set of rules and a round robin-like algorithm is used to split the network traffic (packets) among the nodes, also a session analyzer should be used to maintain integrity due to packet reassembly. Figure 1 shows this structure. One major advantage of this approach is that each packet is processed only once.

On the contrary, in the node level function parallelism, a set of rule groups will be allocated to each node. If a traffic duplicator is used, all packet will be processed by all nodes. In another method, each node has a complete set of rules that by using a packet duplicator, every packet will be sent to every node, if the node doesn't have the related rule to that packet, it will be dropped.

B. *Component level parallelism*

Component level parallelism is a kind of function parallelism that individual components of the IDS architecture are isolated and given their own processing elements such as pre-processing and multi- pattern matching. Consider, three sensors that the first one read the packet, reassemble it, the second one process the packet, and the third one detect the attack and start passive or active action.

C. *Sub-Component level parallelism*

Sub-component level parallelism is related to the parallelization of individual component of the intrusion detection system. Parallelization of Content matching component of intrusion detection system is a good example that may be use function or data parallelism. The combination of component level and sub-component level parallelism is one of the most effective ways. Figure 2 shows it.

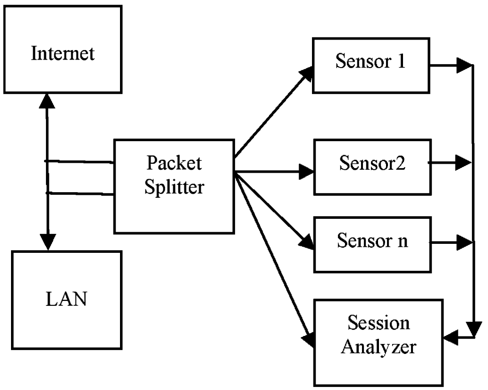


Figure 1.Node Level Data Parallelism Architecture

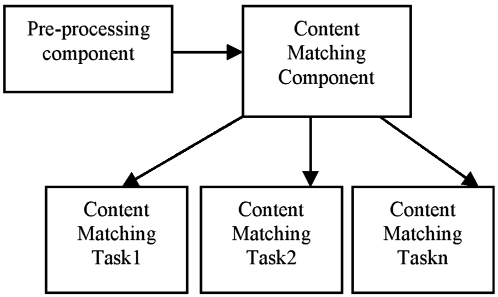


Figure2. Component & Sub-component Parallelism in Content Matching

IV. The Proposed Architecture

According to Wheeler [11], a node level function parallel approach is likely to perform best in comparison with a node level data parallel approach when the system is underloaded. This observation was made in [12, 13] that explored node level function parallelism in network firewall. Basically, the main idea for designing our architecture is derived from NL-FP-2 method proposed by Wheeler [11]. In this method, complete rule groups are spread across nodes. It is possible to use a packet duplicator to send every packet to every node for processing, or a traffic splitter to route each packet to the appropriate node. In this case, rules are clustered into rule groups based on source and destination ports. So, a traffic splitter could route packets based on port numbers.

In our proposed method, a switch or router can be used to split the incoming traffic between two sensors according to their switching or router table. Instead of a switch or router, it is possible to use a system with Linux that has iptable or Windows with a forward packet program on it as a gateway to send the incoming traffic to two sensors. Port forwarding examines packet header and forward it to another machine (host), according to its destination port. Additionally, it is a kind of redirecting to a different IP and/or port. It can be done by an application running on the destination host or hardware such as firewall, router, or proxy server [14].

Port forwarding is used more in offices, universities, and hospitals with lots of computers that are connected to the internet, and they use port forwarding for having several servers to divide the incoming traffic. Such as forwarding port 80 for web server and port 21 for ftp server. In the proposed model, a system with a packet forward program has been used as the gateway.

Each sensor is dedicated parts of the whole Snort rules. When the signature of a known attack is recognized by the detection engine based on the dedicated rules in the Snort, the alerts messages will be sent to the log file and also in our architecture to the Mysql database for further analysis. It is so hard to define all the 65536 ports in iptable or packet forwarding program, and also so time consuming, hence we chose a smaller range of ports for the implementation. Table I shows the division of the ports according to first or second Snort, and also dedicated rules to the Snorts. For example the Http sensor will only select packets sent to the http server in the network.

TABLE I. Ports and Rules Distribution

Sensor Number	Range of Destination ports	Dedicated Rules
Sensor 1 (snort1)	25,80,110,143,8080 (SMTP, HTTP, POP3, IMAP4)	Rules for all the packets with destination ports according to Sensor1
Sensor 2 (snort2)	21,22,23,53,3306 (Ftp, SSH ,Telnet, DNS Server ,MYSQL database system)	Rules for all the packets with destination ports according to Sensor2

The following steps indicate the proposed approach:

- Step1: capture the packets
  After sniffing the network traffic, using some tools such as Winpcap [15] that is a kind of tools that have access to link layer network to capture the network traffic in windows environment, the packets should be split and loaded across the dedicated NIDS sensors.
- Step2: Load the packets
  By the use of packet forwarding program, the packets should distribute between sensors based on their port numbers.
- Step 3: Processing the packets
  In order to detect the attack signature, each sensor takes appropriate rules. Each dedicated sensor has only the intrusion detection system rules set dedicated to the depicted packet. If the attack recognized, alert will be generated and sent to the control center, otherwise by the use of an Ethernet switch the packet will be sent to the local network and servers.

V. Experimental Results

There are several challenges for designing and implementing our architecture in real time and high speed networks, which two most important of them are explained here:

- Scalability: With the recent trend of high-speed networks and need to process a large volume of data by the network intrusion detection system, the architecture of the NIDS should be flexible in the way that adding several sensors to handle this huge amount of traffic is possible and they can process the traffic concurrently.
- Fault tolerance: There is a need to have several sensors in the time of downing one of NIDS's, and forwarding its traffic to another sensor to prevent malicious activities. To do this, the NIDS architecture must monitor the whole sensors.

The testbed is composed by two sensors that each one has Intel high core 2.6 GHz processor with 2 GB RAM. Two sensors have Microsoft Professional XP server pack 2 as operating system. Windows Server 2003 can be used as the operating system for the gateway system.

Pass port 1-0-1 that is a packet forwarding program is used on the gateway system. Snort 2.8.x as the signature-based NIDS platform.Winpcap 4.1.2 as monitoring software is utilized for capturing packets on the network. Snort is configured in the way that all the generated alerts send to Mysql 5.1.49, and KIWI which is a freeware *Syslog* Daemon. By the use of Basic analysis and security engine (BASE) that is a kind of security analyzer, all the alerts stored in MYSQL database can be used in control center for further analysis by the administrator.

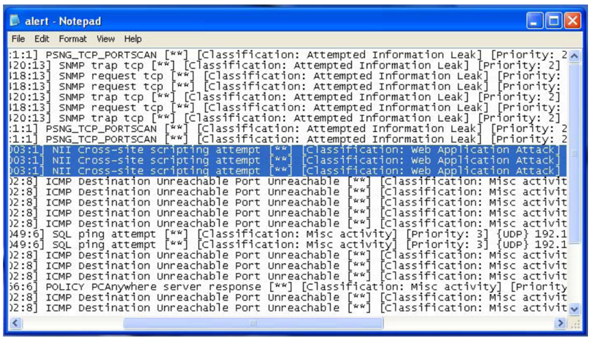
In order to validate the performance of the proposed architecture, two tests should be done. The first one explores the recognition of attacks by both sensors, and the second one compares the centralized architecture with our parallel architecture.

A. Testing the Recognition of Attacks by the proposed architecture

One of the main functions of the signature based network intrusion detection system is to find attack patterns. Hence, two sensors that work in parallel should recognize the attack signatures and take appropriate actions. In order to perform this test, two port numbers are selected, 3306 for the Mysql database server and 80 for the HTTP. By the use of packet forwarding program, the incoming traffic will be forwarded to the sensors. The suitable rules should be assigned to the both Snorts and defined them in the Snort configuration files .e.g. Cross Site Scripting and SQL injection rules are defined for the first Snort, and Mysql rule is allocated to the second Snort. All the port numbers are closed (by the use of a firewall such as COMODO) except 80,443, 3306. So, this sensor should recognize all the attacks that aimed these port numbers.

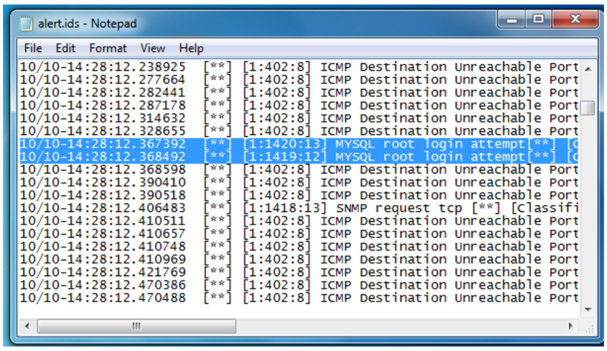
The next step is to choose two tools for attacking the Snorts on the sensors. Acuentix that is a web vulnerability scanner, and shadow scanner that is a network security vulnerability scanner are selected. First one for attacking to the HTTP port (80,443) and second one for the Mysql port (3306) .It is necessary to install WAMP on both sensors that is a kind of web server software all in one package to have Apache, PHP, and Mysql on the windows system. Finally, there is required to install Joomla in the first sensor as it is a content management system utilized for creating websites.

The aim of using Acuentix is to attack the websites, hence there is required to have Joomla. Start to run the Acuentix from another machine, as the port forwarding is running on the gateway, all the incoming traffic will be sent to the first Sensor. It starts to test all the vulnerabilities in the system. Snort recognizes the attacks and takes appropriate action according to its configuration file. Figure 3 is the log file of the first Snort that shows the cross site scripting has been recognized; hence the first Snort works well. For the second Snort, starts running shadow scanner from another machine. It starts to explore all the available vulnerabilities. Figure 4 shows recognition of Mysql root login attempt. So, the second snort works properly, too.



```
:1:1] PSNG_TCP_PORTSCAN [**] [Classification: Attempted Information Leak] [Priority: 2
120:13] SNMP trap tcp [**] [Classification: Attempted Information Leak] [Priority: 2]
118:13] SNMP request tcp [**] [Classification: Attempted Information Leak] [Priority:
118:13] SNMP request tcp [**] [Classification: Attempted Information Leak] [Priority:
120:13] SNMP trap tcp [**] [Classification: Attempted Information Leak] [Priority: 2]
118:13] SNMP request tcp [**] [Classification: Attempted Information Leak] [Priority:
120:13] SNMP trap tcp [**] [Classification: Attempted Information Leak] [Priority: 2]
:1:1] PSNG_TCP_PORTSCAN [**] [Classification: Attempted Information Leak] [Priority: 2
:1:1] PSNG_TCP_PORTSCAN [**] [Classification: Attempted Information Leak] [Priority: 2
)03:1] NII Cross-site scripting attempt [**] [Classification: Web Application Attack]
)03:1] NII Cross-site scripting attempt [**] [Classification: Web Application Attack]
)03:1] NII Cross-site scripting attempt [**] [Classification: Web Application Attack]
)2:8] ICMP Destination Unreachable Port Unreachable [**] [Classification: Misc activit
)2:8] ICMP Destination Unreachable Port Unreachable [**] [Classification: Misc activit
)2:8] ICMP Destination Unreachable Port Unreachable [**] [Classification: Misc activit
)2:8] ICMP Destination Unreachable Port Unreachable [**] [Classification: Misc activit
)2:8] ICMP Destination Unreachable Port Unreachable [**] [Classification: Misc activit
)49:6] SQL ping attempt [**] [Classification: Misc activity] [Priority: 3] {UDP} 192.1
)49:6] SQL ping attempt [**] [Classification: Misc activity] [Priority: 3] {UDP} 192.1
)2:8] ICMP Destination Unreachable Port Unreachable [**] [Classification: Misc activit
)2:8] ICMP Destination Unreachable Port Unreachable [**] [Classification: Misc activit
)2:8] ICMP Destination Unreachable Port Unreachable [**] [Classification: Misc activit
56:6] POLICY PCAnywhere server response [**] [Classification: Misc activity] [Priority
)2:8] ICMP Destination Unreachable Port Unreachable [**] [Classification: Misc activit
)2:8] ICMP Destination Unreachable Port Unreachable [**] [Classification: Misc activit
```

Figure3. Snortl Log File



```
10/10-14:28:12.238925  [**]  [1:402:8] ICMP Destination Unreachable Port
10/10-14:28:12.277664  [**]  [1:402:8] ICMP Destination Unreachable Port
10/10-14:28:12.282441  [**]  [1:402:8] ICMP Destination Unreachable Port
10/10-14:28:12.287178  [**]  [1:402:8] ICMP Destination Unreachable Port
10/10-14:28:12.314632  [**]  [1:402:8] ICMP Destination Unreachable Port
10/10-14:28:12.328655  [**]  [1:402:8] ICMP Destination Unreachable Port
10/10-14:28:12.367392  [**]  [1:1420:13] MYSQL root login attempt[**]  [C
10/10-14:28:12.368492  [**]  [1:1419:12] MYSQL root login attempt[**]  [C
10/10-14:28:12.368598  [**]  [1:402:8] ICMP Destination Unreachable Port
10/10-14:28:12.390410  [**]  [1:402:8] ICMP Destination Unreachable Port
10/10-14:28:12.390518  [**]  [1:402:8] ICMP Destination Unreachable Port
10/10-14:28:12.406483  [**]  [1:1418:13] SNMP request tcp [**] [Classifi
10/10-14:28:12.410511  [**]  [1:402:8] ICMP Destination Unreachable Port
10/10-14:28:12.410657  [**]  [1:402:8] ICMP Destination Unreachable Port
10/10-14:28:12.410748  [**]  [1:402:8] ICMP Destination Unreachable Port
10/10-14:28:12.410969  [**]  [1:402:8] ICMP Destination Unreachable Port
10/10-14:28:12.421769  [**]  [1:402:8] ICMP Destination Unreachable Port
10/10-14:28:12.470386  [**]  [1:402:8] ICMP Destination Unreachable Port
10/10-14:28:12.470488  [**]  [1:402:8] ICMP Destination Unreachable Port
```

Figure4. Snort2 Log File

B. System Performance Test

To validate the proposed model, two scenarios have been discussed. First, using one Snort that processed the whole portion of the traffic with the entire rules .Second, two Snorts allocated an amount of traffic with a portion of rules processed the incoming traffic in parallel.

As described before, the incoming traffic will be loaded between two sensors with a subset of rules, each sensor is responsible to process a portion of traffic and send the generated alerts to the central centre, hence the overall processing time of the incoming traffic will decrease, and consequently the performance of the signature based network intrusion detection system will improve.

As we are going to compare the processing time of the packets for two snorts running in parallel with one snort, so the input should be same. TCPDUMP file that is a dataset for testing the IDS performance will be used as the input of the sensors. Wireshark as a network packet analyzer can be used for capturing and displaying the payload of the packet. TCPDUMP file is divided to two files .In order to divide the file, filtering must be used.

As mentioned before, One file includes all the packets grouped by these protocols: HTTP, POP3, IMAP4, and SMTP, and the other contains the entire packet grouped by these protocols: FTP, SSH, DNS Server, Telnet. Based on the port numbers used by the protocols, all the rules will be assigned to two snorts. The first file has 34406 packets, and the second file has 25455 packets.

After configuring and running both Snorts, the processing time for both files obtained. Run time for packet processing for the first snort with 34406 packets was

0.937000 seconds, and for the second one with 25455 packets was 0.531000 seconds. Now, for comparing the performance of the proposed architecture with centralized architecture, the above steps must be done for the one snort again. Two files must be merged in the Wireshark to have all the packets in one file contain all the HTTP, POP3, IMAP4, SMTP, SSH, DNS Server, and FTP protocols with an entire set of rules. Finally, processing time of all packets with all rules obtained .It took 1.628000 seconds. With a simple mathematics calculation 42% is obtained, and it shows the improved percentage of the performance.

VI. CONCLUSION

In this paper a parallel technique for improving the performance of signature based network intrusion detection systems was proposed. To improve the performance of these systems several approaches have been proposed before, such as using efficient string matching algorithm since an underperforming passive system drops many packets and misses many attacks in the high speed network, using hardware acceleration, and finally using parallelism. Hence the whole system can achieve a higher throughput. Experimental results show the proposed architecture reduces the processing time of the traffic; consequently improves the performance of signature based network intrusion detection system. Also, recognition test prove that both Systems can detect attacks correctly.

As future work, we will try to design a new string matching algorithm for our proposed architecture to improve the efficiency of signature-based intrusion detection task.

REFERENCES

[1] R.Gurley Bace," Intrusion Detection,"7 Th, Ed ,Macmillan Technical Publishing: Dwyer, D.2000.

[2] J.Allen,A.Christie,W.Fithen,J.Mchugh,and J.Pickle," State of the Practice of Intrusion Detection Technologies," Technical Report. CMU/SEI-99-TR-028 ESC-99-028, Networked Systems Survivability Program, January 2000.

[3] M.Garuba,C.Liu,and D.Fraites, "Intrusion Techniques: Comparative Study of Network Intrusion Detection Systems," Fifth International Conference on Information Technolog, New Generations.2008. USA: IEEE, 1-7.

[4] E. Culler and G. Singh," Parallel Computer Architecture: A Hardware/Software Approach," Morgan Kaufman, 1999.

[5] C.Anagnostakis and E.Markatos," An Active Traffic Splitter Architecture for Intrusion Detection," 11th IEEE/ACM International symposium on Digital Object Identifier,2003, Hellas: IEEE.2003, 238-241.

[6] Z.Zhuang,Y. Luo,M. Li,and C,Weng," A Resource Scheduling Strategy for Intrusion Detection on Multi-core Platform," IFIP International Conference on Network and Parallel Computing, China ,2008.

[7] X.Zhao and J.Sun, "A Parallel Scheme for IDS," Second International Conference on Machine Learning and Cybernetics, Wan, 2-5 November .2003.

[8] H.Scallay,K. Alshalfan, and O.B.Fredj," A scalable distributed IDS Architecture for High speed Networks," IJCSNS International Journal of Computer Science and Network Security, VOL.9, No.8, August.2009. Riyadh, - Saudi Arabia.

[9] C.Kopek,E.Fulp,and P.Wheeler,"Distributed Data Parallel Techniques For Content-Matching Intrusion Detection Systems," Military Communication Conference. 2007. Orland, Fl, USA: IEEE.2008. 1-7

[10] P.Wheeler and E. Fulp," A taxonomy of parallel techniques for intrusion detection," *ACMSE 2007.*

[11] P.Wheeler," Techniques for Improving the Performance of Signature Based Network Intrusion Detection Systems," Master of Science,Thesis,Wake Forest University, 2003.

[12] R.Farley and E.Fulp ,"The effect of processing delay on function parallel network firewalls," In IASTED international conference on parallel ,and distributed computing and networks, 2006.

[13] R.Farley and E.Fulp," A function parallel architecture for high-speed firewalls," In IEEE international conference on communications,2006.

[14] Port forward, [Online].Available: www.portforward.com/

[15] The industry-standard windows packet capture library,"Winpcap ," 2010.[Online]. Available: www.winpcap.org