RESEARCH ARTICLE

Optimized Firewall with Traffic Awareness

Mimi Cherian
Computer Department, PIIT, Mumbai University, India.

Madhumita Chatterjee
Computer Department, PIIT, Mumbai University, India.

Abstract –Firewall is one of the well known network-based security devices that have been widely used since the initial days of computer network security. Firewall is designed to allow or reject network traffic depending on firewall rules that displays the types of packets should be accepted or rejected in protected network. Currently growth complexity in network is high and it's very common to find firewall policies consisting of many rules.

Packet filtering is the one of the major contemporary firewall design techniques. An important design goal is to arrive at the decision at the packet only [1]. Firewall access rule list consists of rules which are sequentially checked. This implies that firewall filtering overhead and costing will be higher when the order of the matching rules is higher. Hence it is vital, to minimize the filtering overhead. It's critical to have necessary ordering of rules in the firewall rule set. Firewall validates all inbound and outbound packets by analyzing data packet and then by comparing packets with many firewall rules, that defines whether to accept or discard the traffic. It is very important to improve the firewall policies to improve performance of network.

Index Terms – Firewall, Security, Filtering, Traffic.

1. INTRODUCTION

Firewall validates all inbound and outbound packets by analyzing data packet and then by comparing packets with many firewall rules, that defines whether to accept or discard the traffic. With increasing reliance on the Internet, security remains a top concern due to the increase in number of potential sources of attack networks require protection from unintentional incidents and malicious acts [1].

It is very important to improve the firewall policies to improve performance of network. The overall performance of a firewall is crucial in enforcing and administrating security, especially when the network is under attack. The continuous growth of the Internet, coupled with the increasing sophistication of the attacks, is placing stringent demands on firewall performance [2]. So firewall optimization has a great demand to get good performance. Exiting research efforts developed techniques for either intra-firewall or inter-firewall optimization within a single administrative domain [3]. Network performance is mostly dependent on efficient performance of firewall as decision has to be made whether to accept or reject [4]. Current solutions and technical methods suffer various issues and drawbacks. The critical drawback is least awareness of packet traffic leading to multiple sequential comparison and overhead in computation. In our proposed optimized firewall system we try to improve performance of firewall by accepting or denying packet at the earliest.

The approach considers dynamic packet traffic scenarios and creates a dynamic firewall access rule set which leads to converting rules into binary format using BDD data structure. The Optimized Firewall will dynamically understand traffic behavior pattern and adaptively modifies the access firewall rules to avoid critical degradation in performance due to the dynamic traffic pattern and binary conversion.

Packet filter firewall is the common core component of the any network monitoring tool, which processes every packet header and passes those packets according to filter rules present in the access list [5]. Packet filter firewall is the common core component of the any network monitoring tool, which processes every packet header and passes those packets according to filter rules present in the access list [6]. Multi-dimensional firewall optimization is proven to be NP hard. This has led the research community to focus on Developing various "optimization" heuristics to make firewalls more efficient and dependable [7].

2. RELATED WORK

Gopal Pault et. al, [1] presents an approach in which packet filtering is done based on a binary decision diagram as it improves the storage and look up time for access rule Comparison.BDD approach also improves the packet comparisons compared to list based packet filter. Subrata Acharya et. al, [2] propose an approach to consider traffic based factors for optimizing firewall. They provide a technique to reorder the access rules based on pattern of traffic of packets and then do the packet filter thus reducing comparison computation overhead. P. R. Kadam et. al, [3] provides few techniques used for removal of rule redundancy and leading this work to the searching of malicious user in the system. Anssi Kolehmainen, [4] provides few algorithms used for firewall optimization that emphasize minimizing memory usage and others minimizing execution time. Hongxin Hu et. al. [5] proposed a novel anomaly management framework that facilitates systematic detection and resolution of firewall policy anomalies. A rule-based segmentation mechanism and a grid-based representation technique were introduced to achieve the goal of effective and efficient anomaly analysis. Ravi Shankar P et. al.[7] proposed an optimized classification approach for internet traffic by analyzing the behavior of the nodes for allowing or disconnection of the incoming node by computing the posterior probabilities of the factors with respect to the node.

Zouheir Trabelsi et. al, [8] proposes the technique to improve firewall packet filtering time through optimizing the order of security policy filtering fields for early packet rejection. Hazem Hamed et. al, [9] proposed mechanism is based on the optimization of the filtering fields order according to traffic statistics. First method is a novel algorithm for maximizing early rejection of unwanted flows with minimal impact on other flows. Second method is a new packet filtering dynamic optimization technique that uses statistical search trees to utilize traffic characteristics and minimize the average packet matching time. A. El-Atawy et. al [10] proposed different schemes that focus on statistical filtering to improve average packet filtering time.

3. OPTIMIZED FIREWALL

Our approach tries to optimize the existing firewall by creating a dynamic optimized rule set that permits to deny and accept packets at the earliest with minimal computational overhead. We created dynamic optimized access rules set depending on the pattern of packet traffic; later this optimized rule set goes through Binary Decision Diagram for binary conversion. The optimized rule set created will enhance packet rejection at the earliest and minimize overhead in computation. The results clearly indicate that the proposed traffic-aware optimization strategies have great potential to significantly improve the performance of firewalls and reduce their operational cost.

A. Architecture of proposed optimized firewall

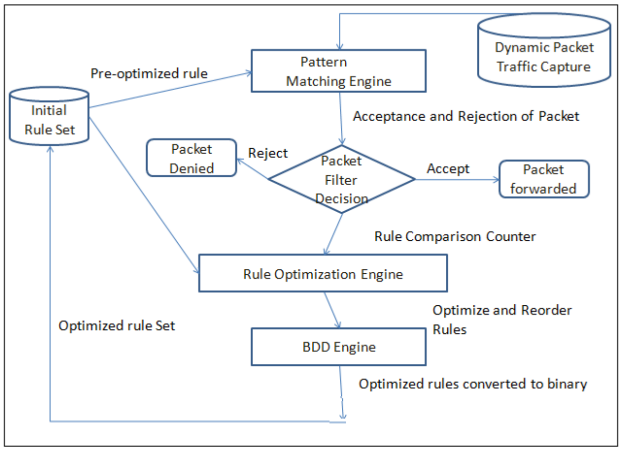


Fig 1. Architecture of Optimized Firewall

In Optimized Firewall Architecture we have an initial rule set which consists of many rules that decides whether incoming packet from public network needs to be rejected or accepted belonging to particular source and destination. These rules undergo optimization and BDD conversion forming an optimized rule set.

The above architecture consists of following modules:

- Pattern matching Engine
- Rule Optimization Engine
- BDD Engine

1) Module 1: Pattern matching Engine:

The initial rule set which consists of pre-optimized rules is taken as input to Pattern matching Engine. We have also taken another input the incoming packets from public network captured by packet sniffer. Both the initial rule set and incoming packets are taken as input for pattern matching.

The rules in initial rule set consists of the fields source address, destination address, service type, protocol number, port number, inbound or outbound and action to be taken. The incoming packets captured by sniffer is pattern matched with the existing rules in initial Rule set, later packets are rejected and accepted accordingly. In Fig 2 we have pre-optimized initial rule set.

Rule	Src Ip	Dst Ip	Src Port	Dst Port	Protocol	In/Out	Action
R1	S1	D1	SP1	DP1	P1	IN	DENY
R2	S2	D2	SP2	DP2	P2	OUT	ACCEPT
R3	S3	D3	SP3	DP3	P1	IN	ACCEPT
R4	S1	D1	SP4	DP4	P1	OUT	DENY
R5	S2	D2	SP5	DP5	P2	IN	ACCEPT

Fig 2 TABLE I: Pre-optimized Initial Rule set

2) Module 2: Rule Optimization Engine:

The pattern matching engine takes the decision of packets to be rejected and accepted, along with it a counter is maintained that has number of times rules were compared. The counter along with initial rule set is input for Rule Optimization Engine. In Fig 3 we have modules within Rule Optimization Engine.

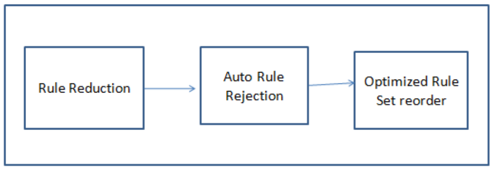
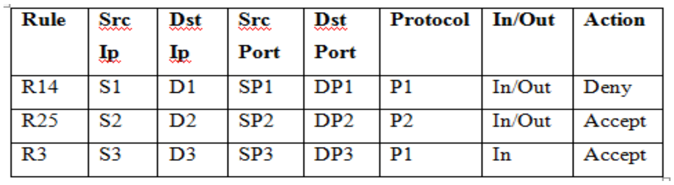


Fig 3. Rule Optimization Engine

Rule Reduction:

In this phase the rules that have the same source, destination, protocol and action but difference in inbound and outbound fields then they will be combined as single rule. The rule reduction will minimize the pattern matching overhead and improves the performance of firewall. In Fig 4 we have reduced.

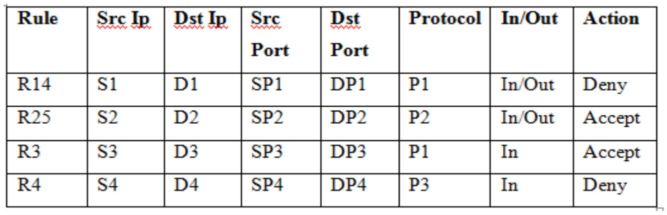


Rule	Src Ip	Dst Ip	Src Port	Dst Port	Protocol	In/Out	Action
R14	S1	D1	SP1	DP1	P1	In/Out	Deny
R25	S2	D2	SP2	DP2	P2	In/Out	Accept
R3	S3	D3	SP3	DP3	P1	In	Accept

Fig 4. TABLE II: Rule Reduction

Auto Rule Rejection:

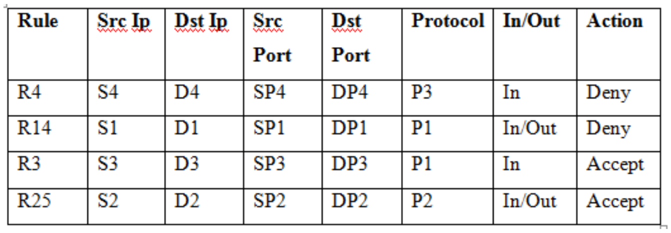
When malicious packets are sent frequently and increases the traffic of incoming packets it degrades the performance of firewall. Resolving this issue is done by our Rule optimization engine, the engine automatically rejects these packets after certain counter limit exceeds, thus improving the performance of firewall. In Fig 5 we have table with auto rejection rule R4 added as it exceeds its counter limit.



Rule	Src Ip	Dst Ip	Src Port	Dst Port	Protocol	In/Out	Action
R14	S1	D1	SP1	DP1	P1	In/Out	Deny
R25	S2	D2	SP2	DP2	P2	In/Out	Accept
R3	S3	D3	SP3	DP3	P1	In	Accept
R4	S4	D4	SP4	DP4	P3	In	Deny

Fig 5. TABLE III: Auto Rule Rejection

Optimized Rule Set Total Reorder:



Rule	Src Ip	Dst Ip	Src Port	Dst Port	Protocol	In/Out	Action
R4	S4	D4	SP4	DP4	P3	In	Deny
R14	S1	D1	SP1	DP1	P1	In/Out	Deny
R3	S3	D3	SP3	DP3	P1	In	Accept
R25	S2	D2	SP2	DP2	P2	In/Out	Accept

Fig 6. TABLE IV: Optimized Rule Set Reorder:

The Packets that are rejected or accepted by pattern matching engine will have a counter maintained. The counter is used to keep count of number of times each rule is compared by incoming packets of public network. Maintaining counter helps to reorder the rules which are highly accessed as top priority in the optimized rule set. When rules are reordered based on counter it enhances early packet rejection thus reducing comparison overhead and increases efficiency of firewall. Fig 6 shows the total reordered optimized rule set based on counter values in which it keeps auto reject rule at high priority for rejecting rule efficiently rule compared to Fig 2.

3) Module 3: BDD Engine:

The optimized rules received from rule Optimization Engine is given as input to BDD module. The module will automatically take the optimized rule list as input and provides its corresponding. blif file (convert into binary) as output. The algorithm developed for the blif file creation is as follows:

Input: Optimized Rule List

Output: containing binary form of the rule list.

1) Take each rule from the rule list.

2) Extract the protocol number, source address and port, and destination address and port.

3) Convert each part in to its binary format in required number of bits.

4) Store the each part of the converted binary form in to the file

This blif file is input to CUDD package to receive optimized BDD output files BDD algorithm decides packet rejection and acceptance. Thus we get the optimized access rule set which enhances early acceptance and rejection of packets with less overhead in comparison and computation.

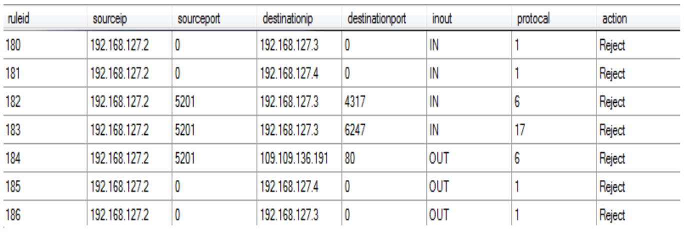
The above optimized rule set in binary replaces the initial pre-optimized rule set. The cycle of optimizing rule set continues to create dynamic optimized firewall rule set as, our Firewall Optimizer keeps continuous watch on incoming packets packet enters from public network.

B. Implementation

The Firewall Optimizer is coded in Vb.Net and its working is tested on 3-4 systems with different OS configuration (Windows 7, Ubuntu). We have used Iperf tool to generate TCP and UDP packet traffic to test the project.

C. Test Scenarios

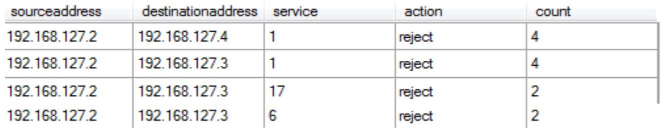
We have created some rules for Pre-Optimized Access rule list of firewall. In the Fig 7 rules for rejecting ICMP, TCP, UDP and HTTP packets are mentioned.



ruleid	sourceip	sourceport	destinationip	destinationport	inout	protocal	action
180	192.168.127.2	0	192.168.127.3	0	IN	1	Reject
181	192.168.127.2	0	192.168.127.4	0	IN	1	Reject
182	192.168.127.2	5201	192.168.127.3	4317	IN	6	Reject
183	192.168.127.2	5201	192.168.127.3	6247	IN	17	Reject
184	192.168.127.2	5201	109.109.136.191	80	OUT	6	Reject
185	192.168.127.2	0	192.168.127.4	0	OUT	1	Reject
186	192.168.127.2	0	192.168.127.3	0	OUT	1	Reject

Fig 7.Pre-Optimized Rule Set

The pre-optimized rule set is given to Optimization Engine. The rules with same source and destination address with same action will be combined as single rule as long as they are outbound and inbound. The Optimization Rule Engine provides the Optimized Rule Set in Fig 8. The optimized rules set have decreased the count of rules from 8 rules to 4 rules.



sourceaddress	destinationaddress	service	action	count
192.168.127.2	192.168.127.4	1	reject	4
192.168.127.2	192.168.127.3	1	reject	4
192.168.127.2	192.168.127.3	17	reject	2
192.168.127.2	192.168.127.3	6	reject	2

Fig 8.Optimized Rule Set

The Optimized rule is then given for BDD conversion which converts these optimized rules into binary format.

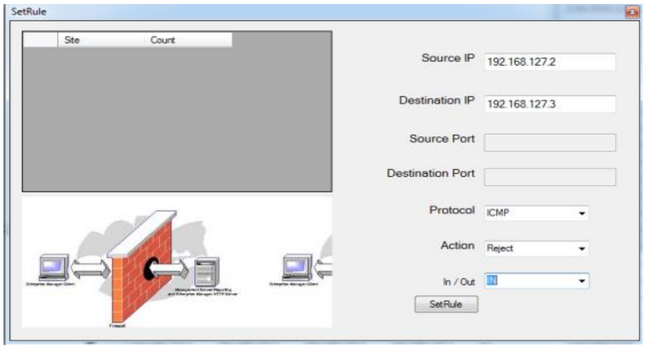
ICMP Test Scenario:

We are initially able to accept ICMP packets as by default our firewall optimizer accepts the packet, later we can add rule to reject ICMP packet by providing destination and source address which will hence forth stop any ICMP packet incoming or outgoing through given source and destination

1) Incoming ICMP packets Accepted by Firewall Optimizer

2) Incoming ICMP packets Rejected by Firewall Optimizer after rule is been added

In Fig 9 the inbound reject rule for ICMP is added and in Fig 10 the screenshot of ICMP packets rejected is shown.



Source IP 192.168.127.2
Destination IP 192.168.127.3
Source Port
Destination Port
Protocol ICMP
Action Reject
In / Out IN

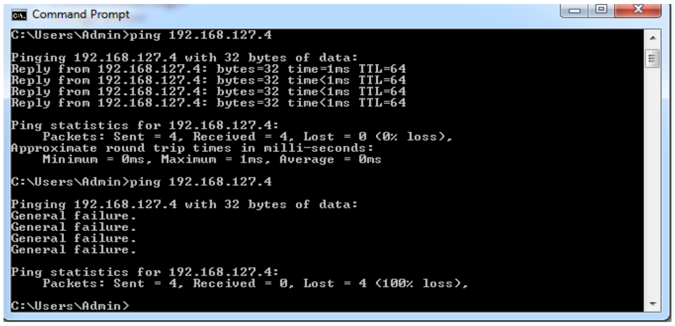
Fig 9. ADD ICMP RULE

TCP Test Scenario

We are able to accept TCP packets as by default our firewall optimizer accepts the packet, later we can add rule to reject TCP packet by providing destination and source address which will hence forth stop any TCP packet incoming or outgoing through given source and destination. In fig 11 both scenarios are shown

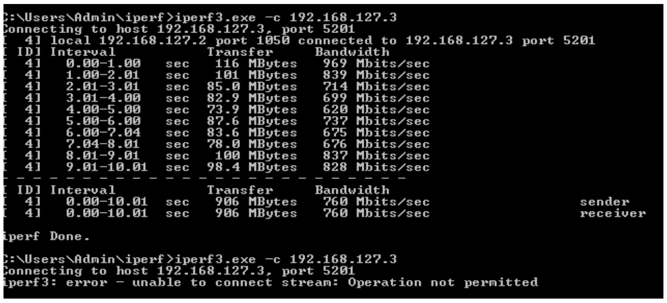
3) Incoming TCP packets Accepted by Firewall Optimizer

4) Incoming TCP packets Rejected by Firewall Optimizer after rule is been added:-



```
C:\Users\Admin>ping 192.168.127.4

Pinging 192.168.127.4 with 32 bytes of data:
Reply from 192.168.127.4: bytes=32 time=1ms TTL=64
Reply from 192.168.127.4: bytes=32 time<1ms TTL=64
Reply from 192.168.127.4: bytes=32 time<1ms TTL=64
Reply from 192.168.127.4: bytes=32 time<1ms TTL=64

Ping statistics for 192.168.127.4:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms

C:\Users\Admin>ping 192.168.127.4

Pinging 192.168.127.4 with 32 bytes of data:
General failure.
General failure.
General failure.
General failure.

Ping statistics for 192.168.127.4:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

C:\Users\Admin>
```

Fig 10.ICMP ACCEPT REJECT



```
C:\Users\Admin\iperf>iperf3.exe -c 192.168.127.3
Connecting to host 192.168.127.3, port 5201
[  4] local 192.168.127.2 port 1050 connected to 192.168.127.3 port 5201
[ ID] Interval           Transfer     Bandwidth
[  4]   0.00-1.00   sec   116 MBytes   969 Mbits/sec
[  4]   1.00-2.01   sec   101 MBytes   839 Mbits/sec
[  4]   2.01-3.01   sec  85.0 MBytes   714 Mbits/sec
[  4]   3.01-4.00   sec  82.9 MBytes   699 Mbits/sec
[  4]   4.00-5.00   sec  73.9 MBytes   620 Mbits/sec
[  4]   5.00-6.00   sec  87.6 MBytes   737 Mbits/sec
[  4]   6.00-7.04   sec  83.6 MBytes   675 Mbits/sec
[  4]   7.04-8.01   sec  78.0 MBytes   676 Mbits/sec
[  4]   8.01-9.01   sec   100 MBytes   837 Mbits/sec
[  4]   9.01-10.01  sec  98.4 MBytes   828 Mbits/sec
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth
[  4]   0.00-10.01  sec   906 MBytes   760 Mbits/sec                  sender
[  4]   0.00-10.01  sec   906 MBytes   760 Mbits/sec                  receiver

iperf Done.

C:\Users\Admin\iperf>iperf3.exe -c 192.168.127.3
Connecting to host 192.168.127.3, port 5201
iperf3: error - unable to connect stream: Operation not permitted
```

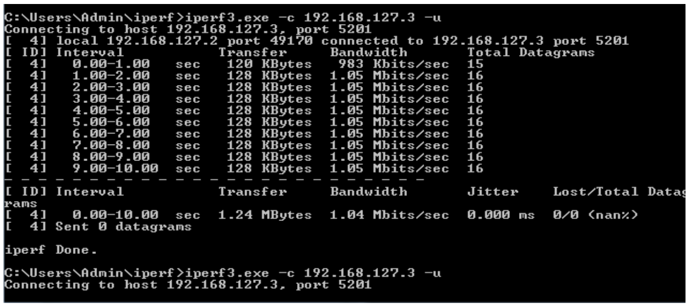
Fig 11.TCP ACCEPT REJECT

UDP Test Scenario:

We are able to accept UDP packets as by default our firewall optimizer accepts the packet, later we can add rule to reject UDP packet by providing destination and source address which will hence forth stop any UDP packet incoming or outgoing through given source and destination. In fig 12 both scenarios are shown

5) Incoming UDP packets Accepted by Firewall Optimizer:-

6) Incoming UDP packets Rejected by Firewall Optimizer after rule is been added:-



```
C:\Users\Admin\iperf>iperf3.exe -c 192.168.127.3 -u
Connecting to host 192.168.127.3, port 5201
[  4] local 192.168.127.2 port 49170 connected to 192.168.127.3 port 5201
[ ID] Interval           Transfer     Bandwidth       Total Datagrams
[  4]   0.00-1.00   sec   120 KBytes   983 Kbits/sec  15
[  4]   1.00-2.00   sec   128 KBytes  1.05 Mbits/sec  16
[  4]   2.00-3.00   sec   128 KBytes  1.05 Mbits/sec  16
[  4]   3.00-4.00   sec   128 KBytes  1.05 Mbits/sec  16
[  4]   4.00-5.00   sec   128 KBytes  1.05 Mbits/sec  16
[  4]   5.00-6.00   sec   128 KBytes  1.05 Mbits/sec  16
[  4]   6.00-7.00   sec   128 KBytes  1.05 Mbits/sec  16
[  4]   7.00-8.00   sec   128 KBytes  1.05 Mbits/sec  16
[  4]   8.00-9.00   sec   128 KBytes  1.05 Mbits/sec  16
[  4]   9.00-10.00  sec   128 KBytes  1.05 Mbits/sec  16
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Jitter    Lost/Total Datag
rams
[  4]   0.00-10.00  sec  1.24 MBytes  1.04 Mbits/sec  0.000 ms  0/0 (nan%)
[  4] Sent 0 datagrams

iperf Done.

C:\Users\Admin\iperf>iperf3.exe -c 192.168.127.3 -u
Connecting to host 192.168.127.3, port 5201
```

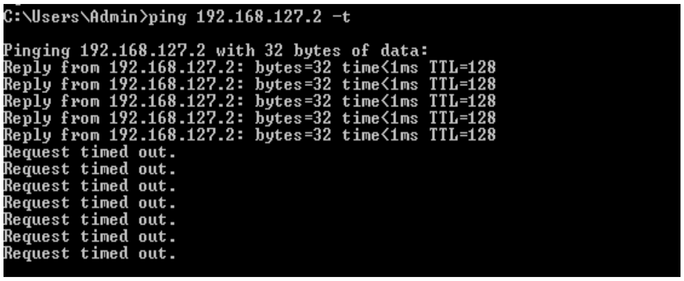
Fig 12.UDP ACCEPT REJECT

Auto Rule Rejection:

Our Firewall optimizer has sniffer which continuously monitors our network, thus if it finds packet repeatedly sent, after certain limit our optimizer reject that packet by adding automatic rule into rule list. In fig 13 auto reject is shown.

7) Incoming ICMP packets Accepted by Firewall Optimizer (inbound) :-

8) Incoming ICMP packets Rejected by Firewall Optimizer (inbound) after counter limit is crossed:



```
C:\Users\Admin>ping 192.168.127.2 -t

Pinging 192.168.127.2 with 32 bytes of data:
Reply from 192.168.127.2: bytes=32 time<1ms TTL=128
Reply from 192.168.127.2: bytes=32 time<1ms TTL=128
Reply from 192.168.127.2: bytes=32 time<1ms TTL=128
Reply from 192.168.127.2: bytes=32 time<1ms TTL=128
Reply from 192.168.127.2: bytes=32 time<1ms TTL=128
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
```

Fig 13.AUTO REJECT

HTTP Test Scenario

In this test scenario we can block HTTP packets from being sent into our private network by specifying the source and destination address of the packet. In fig 14 and Fig 15 HTTP accept and reject screenshots are shown.

9) Incoming HTTP packets Accepted by Firewall Optimizer:-

10) Incoming HTTP packets Rejected by Firewall Optimizer after rule is been added:-

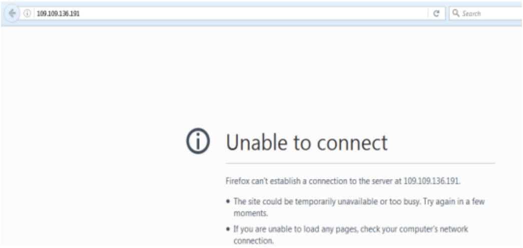
109.109.136.191

Search
LabLogic
Life Sciences | Nuclear Medicine and PET | Radiation Safety | Consumables | Services | About Us

About Us

LabLogic are manufacturer and supplier of instruments and software to the Life Science, Nuclear Medicine / PET and Radiation Safety sectors.

Fig 14.HTTP ACCEPT



109.109.136.191

Unable to connect

Firefox can't establish a connection to the server at 109.109.136.191.

- The site could be temporarily unavailable or too busy. Try again in a few moments.
- If you are unable to load any pages, check your computer's network connection.

Fig 15.HTTP REJECT

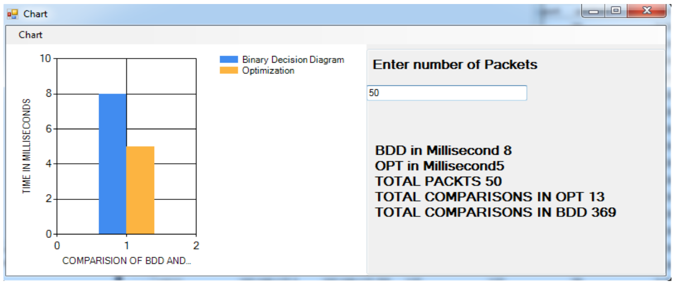
D. Result and Analysis:

We have considered the tabular comparison of performance between BDD packets and Optimized packets in Fig 16.

Sr.no	Number-Pkts	BDD-Comp	OPT-Comp	BDD-T	OPT-T
1	50	369	13	8	5
2	100	391	21	8	6
3	250	398	21	9	7
4	500	398	21	10	7
5	700	1175	42	10	8

Fig 16. TABLE V: Performance Comparison

We have considered the graphical comparison of performance between BDD packets and Optimized packets in Fig 17. The graph clearly states that Optimized rule set takes less time comparison time for packet filtering compared to only BDD conversion based rule set.



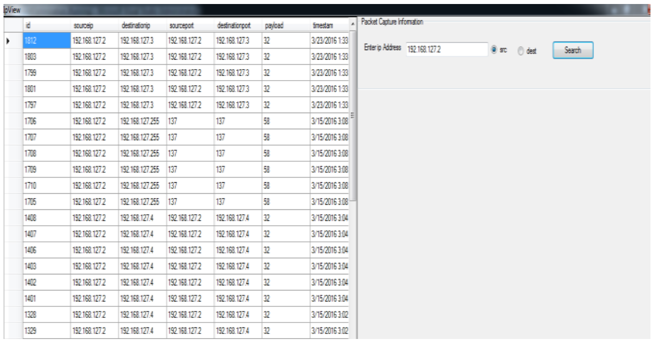
Enter number of Packets

50

BDD in Millisecond 8
OPT in Millisecond5
TOTAL PACKTS 50
TOTAL COMPARISONS IN OPT 13
TOTAL COMPARISONS IN BDD 369

Fig 17 . Performance Comparison Graph

The optimized firewall keeps continuous watch on network and can capture packets to maintain history to optimize the rule set further. Sniffer is the tool with Firewall Optimizer that does this watch. Once packets are captured we can filter them according to given source and destination address which enhances the monitoring of network. In below figure we have a snapshot of packets captured based on a particular source address in Fig 18.



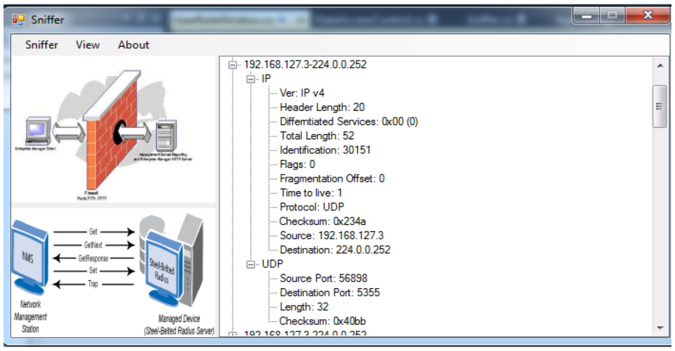
id	sourceip	destinationip	sourceport	destinationport	payload	timestam
1812	192.168.127.2	192.168.127.3	192.168.127.2	192.168.127.3	32	3/23/2016 1:33
1803	192.168.127.2	192.168.127.3	192.168.127.2	192.168.127.3	32	3/23/2016 1:33
1799	192.168.127.2	192.168.127.3	192.168.127.2	192.168.127.3	32	3/23/2016 1:33
1801	192.168.127.2	192.168.127.3	192.168.127.2	192.168.127.3	32	3/23/2016 1:33
1797	192.168.127.2	192.168.127.3	192.168.127.2	192.168.127.3	32	3/23/2016 1:33
1706	192.168.127.2	192.168.127.255	137	137	58	3/15/2016 3:08
1707	192.168.127.2	192.168.127.255	137	137	58	3/15/2016 3:08
1708	192.168.127.2	192.168.127.255	137	137	58	3/15/2016 3:08
1709	192.168.127.2	192.168.127.255	137	137	58	3/15/2016 3:08
1710	192.168.127.2	192.168.127.255	137	137	58	3/15/2016 3:08
1705	192.168.127.2	192.168.127.255	137	137	58	3/15/2016 3:08
1408	192.168.127.2	192.168.127.4	192.168.127.2	192.168.127.4	32	3/15/2016 3:04
1407	192.168.127.2	192.168.127.4	192.168.127.2	192.168.127.4	32	3/15/2016 3:04
1406	192.168.127.2	192.168.127.4	192.168.127.2	192.168.127.4	32	3/15/2016 3:04
1403	192.168.127.2	192.168.127.4	192.168.127.2	192.168.127.4	32	3/15/2016 3:04
1402	192.168.127.2	192.168.127.4	192.168.127.2	192.168.127.4	32	3/15/2016 3:04
1401	192.168.127.2	192.168.127.4	192.168.127.2	192.168.127.4	32	3/15/2016 3:04
1328	192.168.127.2	192.168.127.4	192.168.127.2	192.168.127.4	32	3/15/2016 3:02
1329	192.168.127.2	192.168.127.4	192.168.127.2	192.168.127.4	32	3/15/2016 3:02

Packet Capture Information

Enter ip Address 192.168.127.2 ● src ○ dest

Fig 18. PACKET CAPTURE

Sniffer when keeps watch on network it dynamically shows the captured packets in the active monitor. In fig 19 we have the active monitor which has the packets displayed based on specific source or destination address. Packet sniffer starts receiving the packets asynchronously or iteratively from connected socket so that we can capture all incoming packets till stops the sniffer.



```
192.168.127.3-224.0.0.252
  IP
    Ver: IP v4
    Header Length: 20
    Differntiated Services: 0x00 (0)
    Total Length: 52
    Identification: 30151
    Flags: 0
    Fragmentation Offset: 0
    Time to live: 1
    Protocol: UDP
    Checksum: 0x234a
    Source: 192.168.127.3
    Destination: 224.0.0.252
  UDP
    Source Port: 56898
    Destination Port: 5355
    Length: 32
    Checksum: 0x40bb
192.168.127.3-224.0.0.252
```

Fig 19. SNIFFER

The above few test scenarios were implemented by our Firewall optimizer. We can consider the same scenarios for Inbound and Outbound packets as well.

We have used iperf as external tool to generate the desired TCP and UDP packets for testing our Firewall Optimizer. We can set client and server through iperf which helps in creating inbound and outbound packets for monitoring efficiency of our Firewall Optimizer.

4. CONCLUSION

Various different techniques are used to optimize the firewall like optimization of rules [11] , intra firewall optimization [12], [13], inter firewall optimization [14] [15] and [16] etc. Hit count of rule access based re-ordering uses traffic characteristics to build a long-term rule hit profile. The approach used to build this profile exploits traffic variability [17]. Here we integrate binary conversion and traffic awareness on access control list to give an optimized rule set.

The results clearly indicate that the proposed traffic-aware optimization strategies have great potential to significantly improve the performance of firewalls and reduce their operational cost [18].

Our Optimized Firewall optimizes the firewall rules based on dynamic packet traffic and also does BDD conversion. It helps in reducing computation overhead for packet and rule comparison, thus making the Optimized Firewall more efficient and better in performance. It also keeps continuous monitor on our network using sniffer which helps to reject malicious packets.

REFERENCES

[1] Gopal Pault, Amaresh Pothnal, C. R. Mandalt, Bhargab B. Bhattacharya ,Design and Implementation of Packet Filter Firewall using Binary Decision Diagram, IEEE Students Technology Symposium , 2011.

[2] Subrata Acharya, Jia Wang, Zihui Ge, Taieb F.Znati and Albert Greenberg,Traffic-Aware Firewall Optimization Strategies, 2010.

[3] P.R.Kadam, V.K. Bhusari, Review on Redundancy removal of rules for Optimizing Firewalls, International Journal of Research in Engineering and Technology, Sep-2014.

[4] Anssi Kolehmainen,Optimizing Firewall Performance, 2008.

[5] Hongxin Hu, Gail-Joon Ahn, and Ketan Kulkarni Detecting and Resolving Firewall Policy Anomalies

[6] C. Shen, T. Chung, Y. Chang and Y. Chen, "PFC: A New High Performance Packet Filter Architecture", Journal of Internet Technology, Vo1.8, No.1, Page (s): 67-74, 2007.

[7] Ravi Shankar P , Santosh Naidu P , "A Dynamic Approach of Malicious Node Detection for Internet Traffic Analysis" In Proceedings of IJCNA, 2014.

[8] Zouheir Trabelsi and Safaa Zeidan, Multilevel Early Packet Filtering Technique based on Traffic Statistics and Splay Trees for Firewall Performance Improvement, Communication and Information Systems Security Symposium 2012.

[9] Hazem Hamed, Adel El-Atawy, and Ehab Al-Shaer,On Dynamic Optimization of Packet Matching in High- Speed Firewall, IEEE Journal, Oct-2006.

[10] A. El-Atawy, T. Samak, E. Al-Shaer and H.Li. Using online traffic statistical matching for optimizing packet filtering performance. IEEE INFOCOM'07, pages 866-874, 2007.

[11] V. Srinivasan, S. Suri, and G. Varghese, "Packet classification using tuple space search," in In Proceedings of SIGCOMM. ACM Press, 1999.

[12] J. Cheng, H. Yang, S. H.Wong, and S. Lu. Design and implementation of cross-domain cooperative firewall. In Proceedings of the IEEE ICNP, pages 284 – 293, 2007.

[13] Bremler-Barr A and Hendler D. Space-efficient tcam-based classification using gray coding. In Proceedings of the IEEE INFOCOM, 2007.

[14] C. R. Meiners A. X. Liu and Y. Zhou. All-match based complete redundancy removal for packet classifiers in tcams. In Proceedings of the IEEE INFOCOM, pages 574 – 582, 2008.

[15] E. Torng A. X. Liu and C. Meiners. Firewall compressor: An algorithm for minimizing firewall policies. In Proceedings of the IEEE INFOCOM, 2008.

[16] E. Al-Shaer and H. Hamed. Discovery of policy anomalies in distributed firewalls. In Proceedings of the IEEE INFOCOM, pages 2605 – 2616, 2004.

[17] S. Acharya, J. Wang, Z. Ge, T. Znati, and A. Greenberg, "A Traffic-Aware Framework and Optimization Strategies for Large Scale Enterprise Networks," Technical Report, pp. 1–20, September 2005

[18] M. Roughan, A. Greenberg, C. Kalmanek, M. Rumsewicz, J. Yates, and Y. Zhang, "Experience in measuring backbone traffic variability: Models, metrics, measurements and meaning," in IMW '02: Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurement. NewYork, NY, USA: ACM Press, 2002, pp. 91–92.

Authors

Mimi Cherian received B.E from K.C College of Engineering Mumbai University and pursueing ME from PIIT College, Mumbai University. Published paper on Game theory based network selection in 4G in ICCCCIT-2015 conference also published paper on Firewall Optimization with Traffic Awareness using BDD in IJCIS 2016.

Madhumita Chatterjee completed Mtech and received PhD in Computer Science from Mumbai University in IIT Mumbai. Area of Interest: Networking and Security. Professor and Head Comp Engineering Department, teaching since 24 years.