Optimizing Packet Filter Firewall using Duple Decision Scheme

Anshu Aneja* & Vivek Thapar**

*Assistant Professor, Department of Information Technology, Guru Nanak Dev Engineering College, Ludhiana, Punjab, INDIA.
E-Mail: anshuaneja@yahoo.co.in
**Assistant Professor, Department of Computer Science and Engineering, Guru Nanak Dev Engineering College, Ludhiana, Punjab, INDIA.
E-Mail: vivek_thapar_engg@yahoo.com

***Abstract*—**Network firewalls can use a database of rules to decide which packets will be allowed to move in and out and from one network onto another. However with the increase in size of rule list, it‟s very hard to manage and validate the rules, which can also increase the cost of rule lookup and that may add significantly to latency. This paper presents the study, design and implementation of a packet filter firewall using binary decision diagram which provides faster processing of packets while maintaining the integrity of the original security policy. Duple Decision Scheme or Ordered binary decision diagrams (OBDD) is an efficient and effective method of representing and manipulating Boolean expressions and can be used for the representation of the rule set. This paper explores how OBDD can be used to device methods that can help in validation and analysis of rules to improve performance, and facilitate hardware support.

***Keywords*—**Access List, BDD, Firewall, Multidimensional Filtering, Packet Filtering, Protocol Analyzer, Rule based Selection

***Abbreviations*—**Colorado University Decision Diagram (CUDD), Constraint Logic Programming (CLP), Deep Packet Inspection (DPI), Directed Acyclic Graph (DAG), Ordered Binary Decision Diagram (OBDD), Protocol Source Destination (PSD), Reduced Order Binary Decision Diagram (ROBDD)

I. INTRODUCTION

THE ever-increasing complexity in network infrastructures is making critical demand for network monitoring tools. Network monitoring tools allows individual user processes to have great flexibility in selecting which packets they will receive. A common core component of network monitoring tools is a packet filter [Mogul et al., 1987] which is a programmable selection criterion for selecting packets from a packet stream.

However, some means of controlling access to computers are already present with special programs called „firewalls‟, an old fire-fighting term for controlling damage when a fire has erupted [Mogul et al., 1987].

Firewall is becoming ubiquitous and indispensable to the operation of the network. The continuous growth of the Internet, coupled with the increasing sophistication of attacks, however, is placing further demands and complexity on firewalls design and management. Over the past few years a considerable number of studies have been made on packet filter and packet classification, but they lack in performance and flexibility [Lakshman & Stidialis, 1998]. This research work is carried out to specify the practical demonstration of packet capturing and filtering for high speed computing which is the futuristic need of the organization. Implementation of such packet filter using Duple Decision Scheme or BDD gives more advantages in terms of memory usage and look up time. In the case of the list-based packet filter firewall where rules are checked one by one for each incoming packet, the time taken to decide on a packet is proportional to the number of rules. In this work we present a BDD-based approach which gives much better result in terms of number of comparisons or accesses the rule list.

II. PACKET FILTER FIREWALL

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 [Ingham & Forrest, 2002]. It works in the network layer of the protocol stack and is the simplest type firewall compare to other types of firewall exist [Acharya et al., 2006; Shen et al., 2007]. In this type of firewall there is no consideration of applications, therefore focus is on the individual packets. Each incoming and outgoing packet contains the IP header, which has the information about source and destination address. IP packet internally contains the transport layer protocol information like source, destination port and protocol types. Packet filter firewall is known as less secured one when compared with its counterparts because of its operation in low level devices.

However the literature shows that even though such firewall is less secure, they are more efficient and easily accessible.

III. LITERATURE SURVEY

Due to the enormous impact of firewalls on network security, there has been a significant amount of research work on how to optimize firewalls. Much of this work, however, has been in the area of firewall rule and policy modeling and optimization. Few attempts have been made to achieve multi-dimensional firewall optimization. In Al-Shaer & H. Hamed (2004), present a set of techniques and algorithms that provides automatic discovery of firewall policy anomalies to reveal rule conflicts and potential problems in legacy firewalls, and anomaly-free policy editing for rule insertion, removal, and modification. In Adel El-Atawy et al., (2005), a novel algorithm has been proposed for maximizing early rejection of unwanted flows without impacting other flows significantly. Second, authors present a new packet filtering optimization technique that uses adaptive statistical search trees to utilize important traffic characteristics and minimize the average packet matching time. In Shen et al., (2007), the author proposed new high-performance packet filter architecture named Packet Filter Cache for network monitoring tools. PFC adds a filter rule cache before an existing packet filter. The filter rule cache stores the hash value of a filter rule as a hash table entry that can be searched in O (1) memory access. By taking advantage of the hash lookup speed, PFC can significantly boost filtering performance. In Christiansen & Fleury (2004), the author described the use of Multi-Terminal Interval Decision Diagrams (MTIDDs) as the central structure in the packet filtering mechanism of a firewall. The work includes a mapping of traditional filter specification to predicate logic, and an empirical evaluation of the scheme. A disadvantage in using IDDs and MTIDDs is that the build algorithm has polynomial complexity which can significantly impact the compile time for large filters. In Errin W. Fulp & Stephen J. Tarsa (2005), the author introduced a new firewall security policy representation called the policy tree or trie that maintains policy integrity while requiring significantly less processing time. While the policy trie has shown great promise, more research is needed to evaluate its ability to manage a dynamic policy. In Acharya et al., (2006), a tool to model firewall policies and detect conflicts is described. In this work, the authors focus mainly on single prefix rules. Similarly, in Eronen & Zitting (2001), a Constraint Logic Programming (CLP) framework to analyze rule sets is discussed. These research works offer a good insight in how to model and analyze rule sets. Neither of these works, however, considers optimizing a multi-dimensional rule set. The approach proposed by Srinivasan et al., (1999) optimizes the firewall rule set using Directed Acyclic Graphs (DAGs) to describe rule dependencies. However, it does not provide a methodology to build the DAG. Furthermore, for complex graphs this scheme is ineffective. The approach proposed in this paper removes all the dependencies and hence it becomes possible to achieve optimum rule ordering. In Srinivasan et al., (1999), a framework to analyze and optimize rule sets is described. However, the authors do not provide specific details on how optimization can be achieved within the proposed framework. Furthermore, this work does not consider the traffic characteristics in its optimization approach. The proposed accelerating firewall toolset differs from the above approaches, in its unique approach to consider firewall traffic characteristics in optimizing firewall rule sets.

IV. CURRENT APPROACH

The rule sets implemented in firewalls and routers are important for both security and performance. Unfortunately as the size of the filter rule sets grow it becomes more difficult to understand the rules and the rule sets may become a bottle-neck. Preliminary research has shown that OBDD representations of the rule sets may be a viable method for dealing with these two problems.

A Binary Decision Diagram (BDD) or Duple Decision scheme or branching program is a data structure that is used to represent a Boolean function [Bryant, 1986; Bryant, 1992]. Unlike other compressed representations, operations are performed directly on the compressed representation. A Boolean function can be represented as a rooted, directed acyclic graph, which consists of several decision nodes and terminal nodes. There are two types of terminal nodes called 0-terminal and 1-terminal.Each decision node N is labeled by Boolean variable V_N and has two child nodes low child and high child. The edge from node V_N to a low (or high) child represents an assignment of V_N to 0 (resp. 1).To evaluate expression given an interpretation of the variables, you start at the root and move downwards.

For example the Boolean expression ($x1$or $x2$) and x3 can be represented by the decision diagram in figure 1.

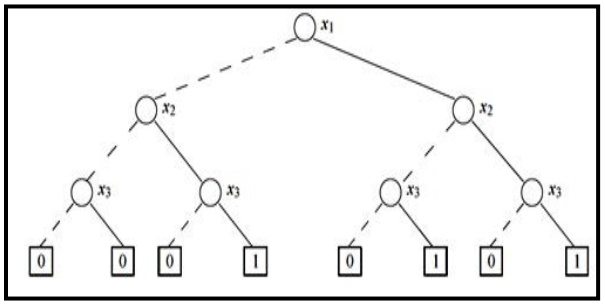

Figure 1 – A Simple Decision Diagram for ($x1$ or $x2$) and $x3$

Table 1 – Total Boolean Variables required for the representation for Access List

Dimension Field	Boolean Variables	Total Number
Protocol Type	p7,p6,……..p1,p0	8
Source IP Address	sa31,sa30,……..sa1,sa0	32
Destination IP Address	da31,da30,……da1,da0	32
Source Port	sp15,sp14,……..sp1,sp0	16
Destination Port	dp15.dp14,……dp1,dp0	16
Total		104

The basic theory of BDD-based packet filter firewall is given in the report [Hazelhurst et al., 1998]. Each packet contains header information like protocol, source address, source port, destination address and destination port. All these are represented using numbers and can be converted in to the equivalent binary format. Suppose, the protocol is TCP and its number is 6, which can be converted to binary as 00000110. To store these 8 bits 8 variables are needed and they can be named as p7, p6, p5, p4, p3, p2, p1, p0. Traversal on the BDD will be done from the top to bottom.

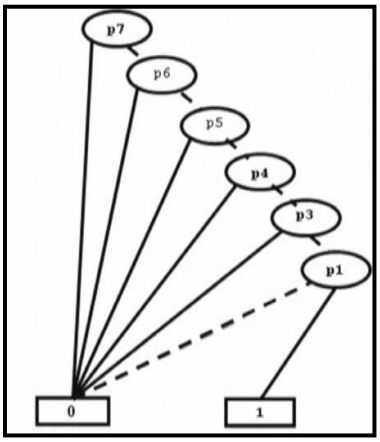

Figure 2 – ROBDD Representation

In traditional list based packet filtering, rule sets are in proprietary formats and rules are expressed in the form of if condition then action. To take decision rules are searched one by one to see whether the condition matches the incoming packets: if it does, the packet is accepted or rejected depending upon the action. If the condition does not match the rule, the search continues with the following rules. Thus the whole list is being traversed in serial fashion which takes too much lookup time to decide the fate of packet. The main disadvantage of list-based algorithm is that rules are placed without prior information like traffic characteristics. This may lead to the increase in decision making time. To achieve a significant amount of gain in decision making the most occurring rule in accepting or rejecting a packet can be placed at the beginning of the rule set, this is called list-based approach with promotion. List-based approach was proven to be the best with rules promotion but it lacks in efficiency.

In our system we will use the above mentioned BDD based approach to represent the rule set in a Boolean expression. A recursive function will be defined that will convert the rules into Boolean expressions. Once the BDD representation of the rule set has been built, it can be used for lookup. On the arrival of packet the relevant packet information can be extracted. This header information provides a mapping from variables to truth values. Using this mapping we start at the root of BDD and works downwards until we get to a leaf. If the variable at the current node is false move to the left, if it is true move to the right [Hazelhurst et al., 1998].For deciding the action of the packet the look up algorithm perform the computation without any repetition. BDD represents multidimensional filtering and holds the all dimension values in binary format. The cost of constructing BDD is very reasonable and feasible and lookup can be done very quickly Doing the same with linear representation of the rules is much more difficult and costly.

The proposed system will have certain potentials that can intercept and log traffic passing over a digital network or part of a network. As data streams flow across the network, the sniffer captures each packet and eventually decodes and analyzes its content can perform Deep Packet Inspection (DPI) to review network packet data according to the appropriate RFC or other specifications.

Snap Shots of Proposed System

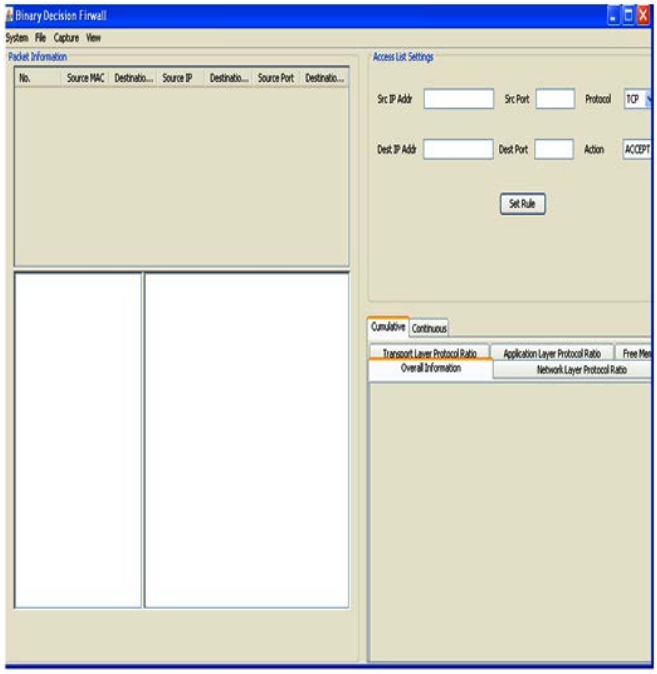

Figure 3 – Home Screen of System

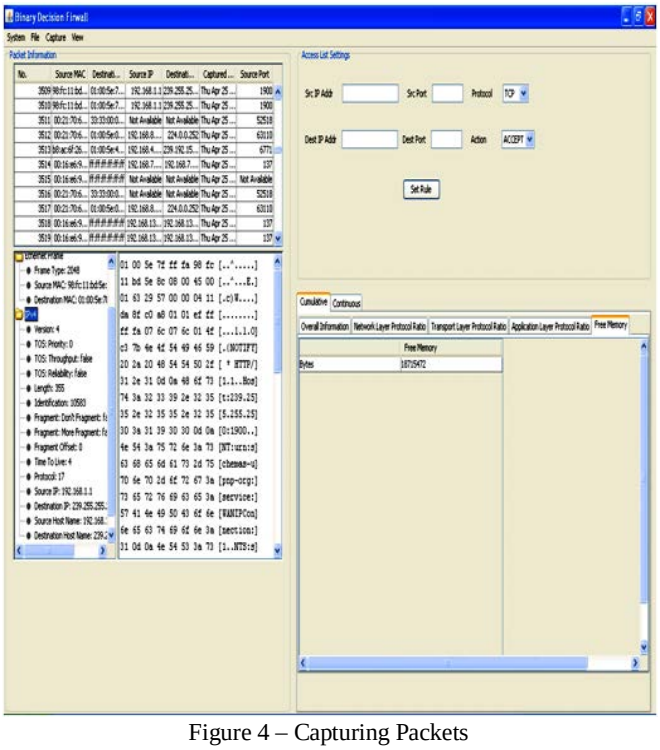

Figure 4 – Capturing Packets

Figure 5 – Multidimensional Filtering

src_ip	dest_ip	src_port	dest_port	protocol_no	date_time
192.168.1.21	122.178.225.33	2962	80	6	Mon Mar 19 17:32:16
192.168.1.21	122.178.225.33	2962	80	6	Mon Mar 19 17:32:16
192.168.1.21	122.178.225.33	2962	80	6	Mon Mar 19 17:32:16
192.168.1.21	122.178.225.33	2962	80	6	Mon Mar 19 17:32:16
192.168.1.21	122.178.225.33	2959	80	6	Mon Mar 19 17:32:17
192.168.1.21	122.178.225.33	2957	80	6	Mon Mar 19 17:32:17
192.168.1.21	122.178.225.33	2957	80	6	Mon Mar 19 17:32:17
192.168.1.21	122.178.225.33	2959	80	6	Mon Mar 19 17:32:17
192.168.1.21	122.178.225.33	2959	80	6	Mon Mar 19 17:32:17
192.168.1.21	122.178.225.33	2963	80	6	Mon Mar 19 17:32:17
192.168.1.21	122.178.225.33	2963	80	6	Mon Mar 19 17:32:17
192.168.1.21	122.178.225.33	2957	80	6	Mon Mar 19 17:32:17
192.168.1.21	122.178.225.33	2957	80	6	Mon Mar 19 17:32:17
192.168.1.21	122.178.225.33	2959	80	6	Mon Mar 19 17:32:17
192.168.1.21	122.178.225.33	2963	80	6	Mon Mar 19 17:32:17
192.168.1.21	122.178.225.33	2963	80	6	Mon Mar 19 17:32:17
192.168.1.21	122.178.225.33	2958	80	6	Mon Mar 19 17:32:18

Figure 6 – All the Rejected Packets

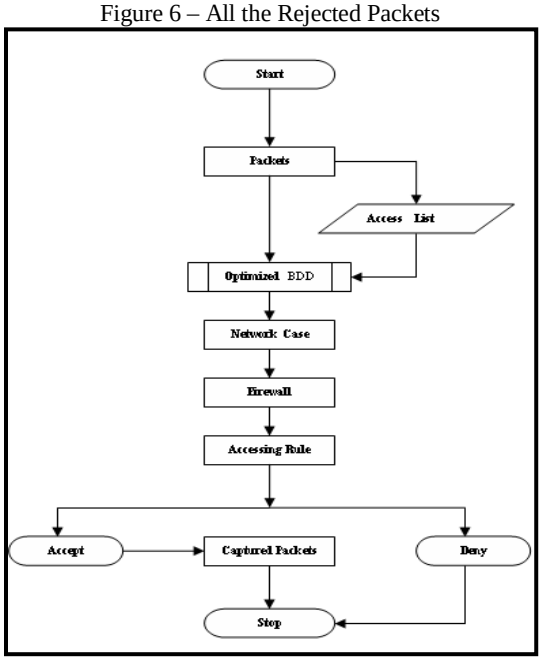

Figure 7 – Flow Chart of Current Proposed System

V. DESIGN AND DEVELOPMENT DETAILS

In this section we are going to discuss about the implementation details and what are the modules implemented for the successful completion of the packet filter firewall project.

5.1. Development of a Module to Generate .blif File

We have implemented a module which automatically takes the access list as input and gives the corresponding .blif file as output. We can give this .blif file as input to the CUDD package [Somenzi, 1998] to obtain the optimized form of BDD as output files.

5.1.1. Algorithm: .blif File Generation

// Input: Rule List
// Output: blif file containing the binary form of the rule list.
While (Rule is exist in 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 blif file.

5.2. Development of a Packet Filter

In this module packet filter firewall takes the dot file generated from CUDD [Acharya et al., 2006] as the input file and decides the action of the incoming and outgoing packets.

5.2.1. Algorithm: BDD Lookup Algorithm

// Input: CUDD order, dot file.
// Output: ACCEPT or REJECT the packet. Check the head of the BDD given by the CUDD If (solid) then
final_op = 1
else
final_op = 0
lookupytr=first node of the BDD
while(lookupytr == 1 OR lookupytr == 0) if(header
lookup ylr = low (lookup ytr)
if(lookupytr == /) then
ACCEPT the packet
else
REJECT the packet

This algorithm searches the BDD generated by the CUDD according to the rule set for which the BDD is given. Thereafter it takes the each bit of the header and compares with the BDD. Once all the bits in the header are checked by traversing the BDD from root to leaf node the decision of packet's acceptability is considered. Normally the packet filter information is stored in a 2-dimensional vector, which contains the information all BDD nodes and their high and low edge values.

VI. RESULTS AND ANALYSIS

We have carried out many experiments on the access lists generated by our own and also from the real world access list.

We have used Protocol Source-Destination (PSD) order for the experiments and found significant outcome in the performance of packet filter firewall. Initially access list length is set to four to differentiate between a fixed order and the best variable order generated by CUDD.

For example, table 2 and table 3 illustrate two sets of access lists and the corresponding comparative result is shown in figure 8.

Table 2 – Sample Access List containing Rules

Rule#	Proto type	src_addr	src_mask	src_port	dest_addr	dest_mask	dest_port	Action
1	TCP	0.0.0.0	255.255.255.255	1023	146.141.0.0	0.0.255.255	25	Permit
2	IP	146.141.0.0	0.0.255.255	80	146.141.0.0	0.0.255.255	120	Permit
3	TCP	0.0.0.0	255.255.255.255	1023	146.141.0.0	0.0.255.255	80	Permit
4	IP	0.0.0.0	255.255.255.255	25	146.141.0.0	0.0.255.255	80	Permit
3	TCP	0.0.0.0	255.255.255.255	1023	146.141.0.0	0.0.255.255	1023	Permit

Table 3 – Number of Nodes Varies using User Order and CUDD Order

Access List No.	List Length	No. of Nodes using User Order	No. of Nodes using CUDD Order
1	1	92	92
1	2	188	124
1	3	200	128
1	4	206	134

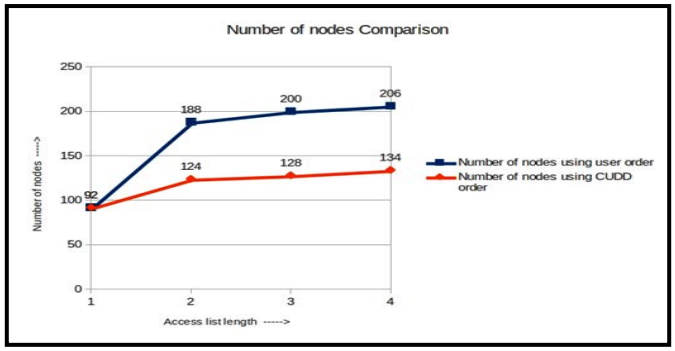

Figure 8 – Number of Nodes in BDD using User Order and CUDD Generated Order

From figure 8, it is realized that the number of nodes using the CUDD generated order is much lower than the user given fixed order. Therefore, storage needed for the corresponding packer filter firewall can be reduced significantly.

6.1. Results for Real World Access List

For real world access list we have considered the access control list from a real LAN, which has around 267 rules. We have taken initial 40 rules as first access list. We have used this first access list for the both the firewalls (BDD based and list-based with promotion) as input. After that, in each stage, we have added 40 more rules to get another set of results. For each access list we have checked the number of comparisons required for the reject or acceptance of the given set of input packets, which are generated randomly. In BDD-based packet filter the number of comparisons is the number of nodes visited by the firewall to decide on the acceptance or rejection of the packets [Paul et al., 2011]. For acceptance of the packet it will take more comparisons than the reject as it goes to the bottom level of BDD for confirmation. In list-based packet filter number of comparisons is the number of rules it has visited and in each rule how many packets it has checked. We have generated two different types of protocol; one is "most-accept packets" and another is "most-reject packets". In the case of "most-reject packets" protocol BDD-based packet filter will traverse lesser nodes in the BDD to decide the fate of the packet. But in the case of list-based packet filter the whole set of rules in the current access list will be traversed. Second type of input data we used is the "most-accept packets" protocol. In this case BDD-based packet filer the algorithm traverses the BDD from the top to bottom to decide the packet's acceptance. But in case of list-based approach the incoming packet may match with any rule in the access list. Hence, the total number of comparisons required will be depending on the position of the matching rule in the access list. Interestingly, in the second case also our design gives better result compared to list-based packet filter firewall.

Table 4 and 5 show the average number of comparisons taken by both the firewalls to decide it acceptance or rejection on the given one million random packets. In the case of the "most-reject packets" the performance of the BOD-based packet filter is higher as the number of comparisons is significantly less. For one million incoming packets with access list sets from 40 to 267 the average reduction in number of comparisons for BDD-based packet filter firewall is 75% when compared with list-based packet filter firewall and "most-reject packets" protocol is chosen. For "most-accept packets" protocol the average reduction is nearly 34%, but for 80-267 access list sets this reduction is even better, nearly 50%. Hence, for both the protocols the performance of BDD-based packet filter firewall is significantly better than the list-based packet filter firewall system. Table 4 and 5 Comparative results between BDD based and List based approaches

Figure 9 and figure 10 illustrate these comparative results in graphical format.

Table 4 – Comparison for Most Reject Packets

Most Reject Packets	Number of Comparisons with 1 Million Packets		
Access List Length	BDD Based	List Based	% Reduction
40	32054234	55668183	63.46
80	32026791	120134708	73.34
120	32537347	171710572	81.07
160	42643811	198408729	78.50
200	48068402	215368648	77.68
267	49690340	220103588	77.42
		Average Reduction = 75.24	

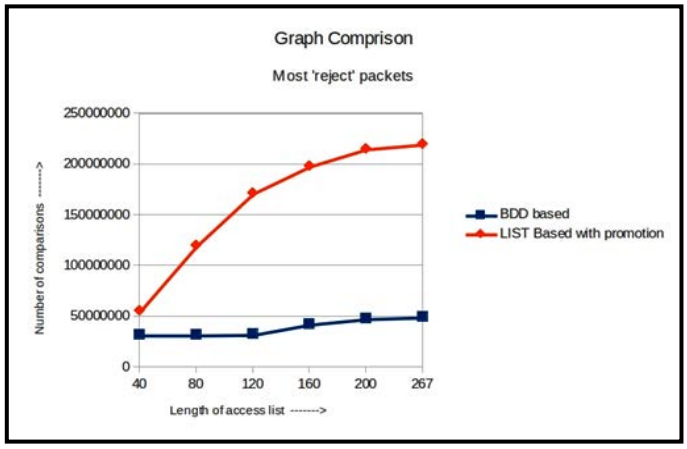

Figure 9 – Graph Comparison for "Most-Reject Packets"

Table 5 – Comparison for Most Accept Packets

Most Accept Packets	Number of Comparisons with 1 Million Packets		
Access List Length	BDD Based	List Based	% Reduction
40	60386771	40360524	-49.61
80	58431012	68786826	15.05
120	57783157	100364081	42.47
160	58144886	129886903	55.23
200	55868916	164023455	65.93
267	55555756	208888173	73.40
		Average Reduction = 33.74	

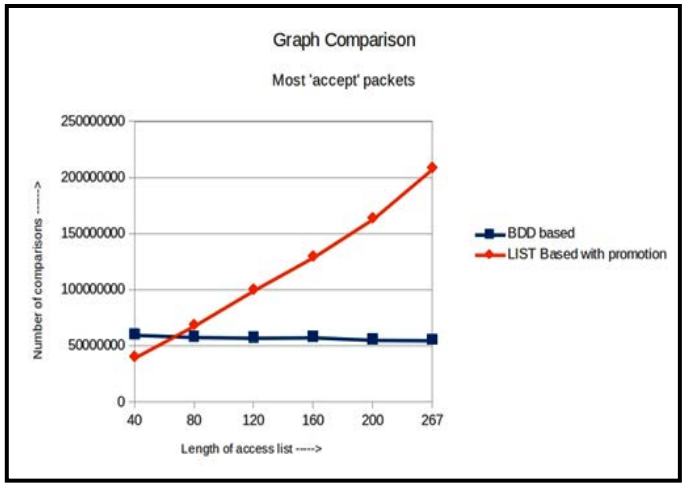

Figure 10 – Graph Comparison for "Most-Accept Packets"

VII. Conclusion

Aim of this work is to explore alternative representations for rule sets. At this stage the focus is exploring the „back end" – internal representations of the rule set that can be used to improve lookup performance and provide network managers a tool to understand and analyze the rule set. List-based packet filter considerably lacks its efficiency when compared to BDD-based approach. We remove the redundant computation using BDD approach and implement the multidimensional filtering. Our duple decision diagram or BDD-based design for packet filter firewall takes less space for storage and less look-up time for accept or reject the incoming packets.

Beside that our system can be used to analyze network problems and detect network intrusion attempts. It can also provide help in debugging client/server communications and debugging and analyzing network protocol implementations.

References

[1] R.E. Bryant (1986) "Graph-based Algorithms for Boolean Function Manipulation", *Transaction on Computers*, Vol. C-35, No. 8, Pp. 677–691.

[2] J. Mogul, R. Rashid & M. Accetta (1987), "The Packet Filter: An Efficient Mechanism for User-Level Network Code", *Proceedings of the Eleventh ACM Symposium on Operating Systems Principles*, Pp. 39–51.

[3] R.E. Bryant (1992), "Symbolic Boolean Manipulation with Ordered Binary Decision Diagrams", *ACM Computing Surveys*, Vol. 24, No.3, Pp. 293–318.

[4] S. Hazelhurst, A. Fatti & A. Henwood (1998), "Binary Decision Diagram Representations of Firewall and Router Access Lists", jtp:/(ftp.cs. wits.ac.zalpublresearchlreportsITR-Wi.)

[5] T.V. Lakshman & D. Stidialis (1998), "High Speed Policy-based Packet Forwarding using Efficient Multi-Dimensional Range Matching", *Proceedings of SIGCOMM*, ACM Press, Pp.203–214.

[6] F. Somenzi (1998), "CUDD: CU Decision Diagram Package"; http://bessie.colorado.edu;'-fabioICUDD, University of Colorado at Boulder.

[7] V. Srinivasan, S. Suri & G. Varghese (1999), "Packet Classification using Tuple Space Search", *Proceedings of SIGCOMM*, ACM Press, Pp. 135–146.

[8] P. Eronen & J. Zitting (2001), "An Expert System for Analyzing Firewall Rules", *Proceedings of the 6th Nordic Workshop on Secure IT Systems (NordSec 2001)*, Copenhagen, Denmark, Pp. 100–107.

[9] K. Ingham & S. Forrest (2002), "A History and Survey of Network Firewalls", *Technical Report 2002-37*, University of New Mexico Computer Science Department.

[10] M. Christiansen & E. Fleury (2004), "An MTIDD based Firewall using Decision Diagrams for Packet Filtering", *Telecommunication Systems*, Vol. 27, No. 2–4, Pp .297–319.

[11] E. Al-Shaer & H. Hamed (2004), "Modeling and Management of Firewall Policies", *IEEE Transactions Network and Service Management*, Vol. 1, No. 1, Pp.2–10.

[12] Adel El-Atawy, Hazem Hamed & Ehab Al-Shaer (2005), "Adaptive Statistical Optimization Techniques for Firewall Packet Filtering", *IEEE Infocom*, Pp.1–15.

[13] Errin W. Fulp & Stephen J. Tarsa (2005), "Trie-based Policy Representations for Network Firewalls", *Proceedings of the IEEE International Symposium on Computer Communications*, Pp. 434–441.

[14] S. Acharya, Wang, Z. Ge, T. Znati & A. Greenberg (2006), "Simulation Study of Firewalls to Aid Improved Performance", *Proceedings of ANSS*, Pp. 18–26.

[15] S. Acharya, J. Wang, Z. Ge, T. Znati & A. Greenberg (2006), "Traffic Aware Firewall Optimization Strategies", *Proceedings of ICC*, Pp. 2225–2230.

[16] C. Shen, T. Chung, Y. Chang & Y. Chen (2007), "PFC: A New High Performance Packet Filter Architecture", *Journal of Internet Technology*, Vo1.8, No.1, Pp. 67–74.

[17] G. Paul, A. Pothnal, C.R. Mandal & Bhargab B .Bhattacharya (2011), "Design and Implementation of Packet Filter Firewall using Binary decision diagram", *Proceedings of Student Technology Symposium*, IIT, Kharagpur, Pp. 17–22.

Anshu Aneja did his Graduation from Punjab Technical University, Kapurthala in 2003 and perusing his M.Tech in Information Technology at Guru Nanak Dev Engineering College, Ludhaiana, India from Punjab Technical University, Kapurthala. His research interests are in the fields of Network Security, Wan Technologies, Internetworking and Routing Protocols. He has presented many papers in different seminars and conferences His research papers have been published in various national and international journals.

Vivek Thapar did his graduation from Punjab Technical University, Kapurthala and Post Graduation from Punjabi University with 72%. He is involved in research since last four years. His research paper has been published in many national and internationals journals. He has presented many papers in different seminars and conferences. Currently he is involved in developing novel software for different statistical methods and presently working as Asst. Prof in Computer Science and Engineering at Guru Nanak Dev Engineering College, Ludhiana, India. His area of Specialization is Network Security and Web Technologies. He is currently doing PhD from Punjab Technical University.