Rhodes University

Department of Computer Science

Mungole Mukupa

Firewall Rule set Optimization

Supervisor: Mr Barry Irwin

Literature Review

Abstract

A Firewall is an intermediate system placed between two networks of different trust levels. This is mostly between the secure corporate network and the less secure public Internet. It is intended to keep the corporate network safe from intruders, hackers and other malicious entry into the corporate network. This is done by implementing the corporate security policy for controlling traffic and managing connections between internal and external network hosts. The continuous growth of the Internet combined with the increase in the complexity of the attacks, is placing heavy demands on the firewall performance. Recent works in this area have investigated ways of improving firewall performance based on different criteria. This research looks at ways of optimizing the rule set to reduce on the packet matching time by combining traffic characteristics with rule matching patterns in a dynamic way to increase throughput by reducing packet inspection time. This is because only a percentage of the total rules set in the rule base are matched by a given packet. This is inefficient and drastically reduces throughput.

Contents

1 Introduction

Internet connectivity is growing with most enterprises shifting to the use of web based services for service provision [14]. This is seen to be sharpening their com- competitive edge as it gives them and their customers, rapid access to information. Firewalls provide a mechanism for protecting these enterprises from the less secure Internet over which their customers or collaborating partners transfer packets destined for the corporate network [6,12,18]. Firewalls have grown to take up more tasks than merely filtering traffic to managing bandwidth, routing control, packet forwarding e.t.c.[8, 22]. This research focuses on the Network Layer operation of a firewall and explores ways of improving its performance by optimizing the rule set. A corporate network policy is translated into rules that are defined as the rule base or rule set. These rules are configured on the firewall to provide the security and packet availability while ensuring that data,system integrity and confidentiality are maintained [2, 3,12]. Research in the subject of improving firewall performance continues to be done by many network researchers some of whose work we will be discussing in this review. This research looks at combining many aspects of firewall functionality in designing a tool that can be integrated in a firewall to optimize the rule set

and give the much sought after increase in throughput. Below is a functional illustration of a firewall:

2 History of Firewalls

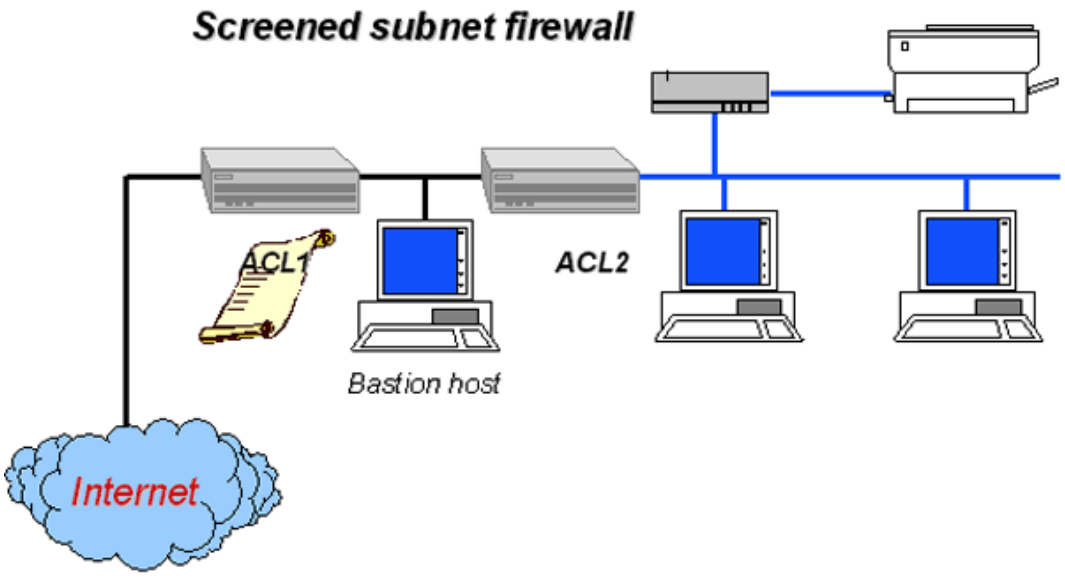
A firewall is a form of network perimeter defense designed to help an organization comply with the rules defined in its IT security policy. The name Firewall is derived from the ancient design of building a wall between two or more buildings or rooms to stop the spread of re if it broke out [25]. The first generation of firewalls were made by Cisco Systems Inc's IOS software division around 1985. They were called Packet Filter firewalls. Late 80s into the 90s saw the second generation of firewalls called Circuit Level Firewall pioneered by Dave Presotto and Howard Trickey of AT&T Bell laboratories [25 , 26]. They also worked on the third generation called Application Layer Firewalls and were publicized in 1991 by Marcus Ranum of AT&T Bell laboratories. Ranum's work became the first commercial product called SEAL by Digital Equipment Corporation. Check Point Software released the first commercial firewall product based on the fourth generation architecture in 1994 [7, 19]. The fifth generation firewall architecture, the Kernel Proxy architecture was worked on by Scott Wiegel, chief scientist at Global Internet Software Group Inc in 1996. Cisco Systems 2 Inc released the first commercial product, Cisco Centri Firewall, based on this architecture in 1997 [24].

3 Firewall Categories

Firewalls are categorised according to different approaches based on their functionality and how they relates to a given network. Below are the classes and examples for each category.

- Processing mode (Packet filtering, Application gateways, Circuit gateways, MAC layer firewalls and Hybrids)
- Development era (Based on generations as discussed below)
- Intended deployment structure (Stand alone, self contained, Small Office/Home Office , SOHO, Commercial grade)
- Architectural implementation (Packet filtering routers, Screened host firewalls, Dual-homed rewalls, and Screened subnet rewalls)

Figure 1: Firewall Architecture: Screened Subnet (Taken from [12])



4 Evolution of Firewalls

Firewalls have evolved from being basic packet filtering devices to the now obtaining complex implementations incorporating functions like management of network bandwidth, enforcing requirements to use a web proxy server, protecting resources in the organisation, preventing malicious attackers from the Internet among others [19 , 20]. The growth of networks, the desire by companies to share their intranets with customers and business partners has brought challenges and a greater need for network security [16]. The innovations in firewall technologies resulted largely from Department of Defense research and funded projects [25]. These firewall generations are summed up below as types:

5 Firewall Types

This research is focusing on the processing mode type of firewalls as they relate to Network Layer packet filtering. We howver give a summary of firewall classifications here following.

5.1 Packet Filter Firewall

These are based on first generation firewall technology. They analyze network traffic at the transport layer [19]. They examine each IP network packet to see if it matches one of the rules defined for allowing or denying data flows. The decision is based on the information they get from the packet's transport layer headers and the direction the packet is going into [24]. They are therefore configured to check:

- transport layer type (TCP, ICMP, UDP)
- source port
- destination IP address
- Source IP address
- Network interface the packet arrives on
- destination port

Packet filters do the above by applying a rule set residing in the TCP/IP kernel that defines what action goes with which rule. They come under three sub categories:

5.1.1 Static Filtering

These firewalls require that filtering rules governing how the firewall decides which packets are allowed and which are denied are developed and installed.Hence the name static [4,15]

5.1.2 Dynamic Filtering

Dynamic filtering firewalls allow the firewall to react to emergent event and update or create rules to deal with traffic events dynamically.

5.1.3 Stateful Filtering

This type of firewalls keep track of each network connection between internal and external systems using a state table [27]. Filtering decisions are made based on the kept information about a given connection.

5.2 Circuit Level Firewalls

These are based on second generation firewall technology [25]. They work based on the fact that a packet is either a data packet or a connection request belonging to a connection or circuit between two peer transport layers [24]. These firewalls work by:

- checking that each connection setup follows a handshake system for the transport layer protocol being used.

- storing a session identifier for the connection
- connection state: handshake, established, or closing
- only forwarding packets after the handshake is complete
- maintaining a table of valid connections and removing it once the connection is terminated
- closing the virtual circuit after transmission

Like packet filters, circuit level firewalls work by applying a rule set that is maintained in the TCP/IP kernel [7 , 26]. They allow all network packets bound for that connection through the firewall as stated in the firewall server's routing table but do not do further security checks on the packets [24].They also perform Network Address Translation and perform checks to see if a packet has been spoofed and whether the data being transported complies with the protocol definition.

5.3 Application Layer Firewall

Also called third generation firewall. These firewalls evaluate packets for valid data at the application layer before allowing a connection.

- examines data in network packets at the application layer
- maintains connection state and sequencing information
- can validate passwords and service requests
- most of them include proxy services for specific services such as HTTP or FTP which provide more checks and generate audit records about the traffic they transfer.

Application layer firewalls generally do not focus on the network packet information. This makes the risk higher if the network layer is not performing well [23]. They also employ proxies that require additional passwords or some validation of some sort.This brings more delays.

5.4 Dynamic Firewalls [Stateful Firewall]

A fourth generation firewall type allowing modification of the rule base. A virtual connection is established and the packet is allowed to travel the firewall server [19]. These provide support for UDP packets by associating them with a virtual connection [20]. The connection information is kept for a short period and the connection is terminated if no response packet is received within that short time. They are good for not allowing unwanted UDP packets into a network because the response packet must contain a destination address that matches the original source address [25].

5.5 Hybrid Firewall

Because of the need to do more than packet inspection, firewalls are being implemented as hybrid systems. These are mostly implemented by adding packet filtering to an application gateway. Cisco PIX firewalls are an example of such hybrid firewalls [25].

6 Firewall Architecture Review

The above architectures have their own advantages and disadvantages when evaluated based on performance and security.Based on performance, Packet Filers provide the highest performance, then Circuit Level, Dynamic Packet Filter re walls and Application Layer firewalls [19,25]. This is a reverse in terms of security. Application Layer Firewalls make packets go through more protocol layers inspection and detail [16]. Application Layer Firewall therefore are seen to be more secure than Dynamic Packet filtering firewalls that are viewed more safer than Circuit Level firewalls and Packet filtering firewalls [26]. Application Layer firewalls are more process intensive and hence slower but considered to be the most secure. A firewall is just one part of the network security system of a private network. These firewalls do not protect the network from viruses, insider attacks and previously unknown attacks [23]. This is because most firewall technology works on a 'catch up' and 'protect from known threats' technology [24]. Therefore keeping the network secure demands continued updates of the firewall rule set and other security policies [20]. The fact that a network is dynamic and not all rules in the firewall rule base are matched at all times is the motivation for this project which is seeking to and ways of optimizing the rule set to improve throughput at layer 3, Network Layer. Below is a sample design of the firewall and demonstrated functionality:

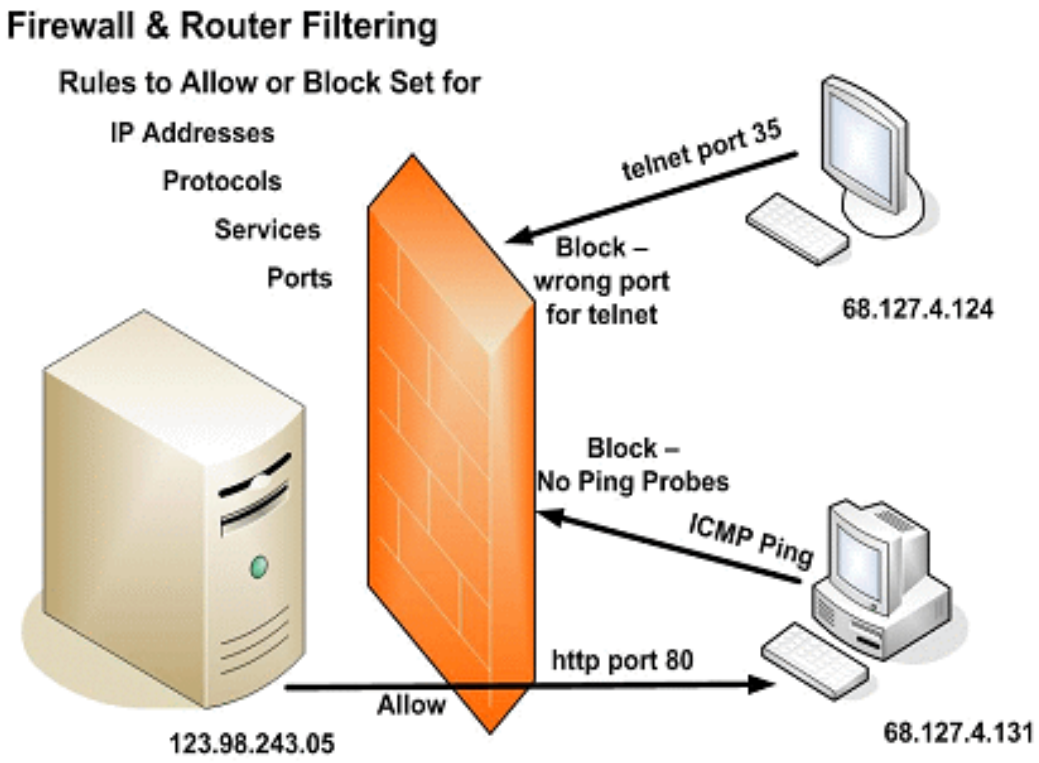


Figure 2: Example Firewall Deployment (Adapted from [16])

7 Project Target Firewall

Network performance highly depends on the efficiency of the firewall because for each network packet entering or leaving a network a decision is made whether to accept or deny it based on the rules defined in the rule set [7]. This project sees Network level filtering, layer 3, as being a critical point to do the filtering optimization at, as the other levels like Application Layer firewalls depend on the packet filtering at the lower level for them to perform effectively.
We are therefore targeting to devise and implement a firewall rule set optimization method in a tool for packet filtering in *ipfw* firewall system for FreeBSD implementation.

7.1 IPFW

The IPFIREWALL (IPFW)is a FreeBSD sponsored firewall application.It is authored and maintained by FreeBSD [8]. It uses stateless rules and a 7 legacy

rule coding technique to provide simple stateful logic [7, 23]. Ipfw supports both IPv4 and IPv6. It has seven components that make it powerful as listed below:

- kernel firewall filter rule processor and its integrated packet accounting facility
- the logging facility
- the divert rule which triggers the NAT facility
- advanced special purpose facilities
- the dummynet traffic shaper facilities
- the fwd rule forward facility and bridge facility
- the bridge and ipstealth facility

8 Firewall Rule set

A rule set is a group of rules programmed to allow or deny packets.The decision to allow or deny is based on the values contained in the packet. The firewall rule set processes both the packets arriving from the public Internet, as well as the packets originating from the internal network [7, 23]. Every service based on TC/IPA i.e.: telnet, www, mail, etc. is predefined by its protocol i.e SSH, HTTP, SMTP e.t.c and privileged (listening) port. Packets destined for a specific service, originate from the source address using an unprivileged (high order) port and target the specific service port on the destination address. All the above parameters (i.e. ports and addresses) are used as traditional selection criteria to create rules which will pass or block services.[5 , 8 , 25]. When a packet enters the firewall it is compared against the first rule in the rule set and progresses one rule at a time moving from top to bottom of the set in ascending rule number sequence order.When the packet matches the selection parameters of a rule, the rule's action field value is executed and the search of the rule set terminates for that packet [7, 8, 23]. This is referred to as the first match wins search method. If the packet does not match any of the rules, it is caught by the default rule which denies all packets and discards them without any reply back to the originating destination [4 - 24].
Firewall configuration is a heavy and error prone task [23]. This is because rule sets grow to thousands of rules and network traffic trends keep changing [20]. Below is a sample firewall rule set for IPFW:

```
#!/bin/sh
########################## defaultruleset#########
# flush out the list before we begin
ipfw -q -f flush
#rules command prefix
cmd="ipfw -q add"
skip="skipto 800"
pif="bridge0"
```

```
#####################################################
# Allow LAN traffic
#####################################################
$cmd 005 allow all from any to any via em0
#####################################################
# No restrictions on Loopback interface
#####################################################
$cmd 010 allow all from any to any via lo0
#####################################################
# check if packet is inbound and NAT address if it is
#####################################################
$cmd 014 divert natdip from any to any in via $pif
#####################################################
# Allow packets part of established flows if added to dynamic rules
#####################################################
$cmd 015 check-state
#####################################################
# Outbound traffic section
#####################################################
$cmd 020 $skip tcp from any to 192.168.46.134 53 out via $pif setup keep-state
#Allow out non-secure www function
$cmd 040 $skip tcp from any to any 80 out via $pif setup keep-state
#Allow out secure www function https over TLS SSL
$cmd 050 $skip tcp from any to any 443 out via $pifsetup keep-state
#Allow out send & get email function
$cmd 060 $skip tcp from any to any 25 out via $pif setup keep-state
$cmd 061 $skip tcp from any to any 110 out via $pif setup keep-state
#Allow out FreeBSD (make install & CVSUP) functions
$cmd 070 $skip tcp from me to any out via $pif setup keep-state uid root
#Allow out ping
$cmd 080 $skip icmp from any to any out via $pif keep-state
#Allow out time
$cmd 090 $skip tcp from any to any 37 out via $pif setup keep-state
#Allow out nntp news (i.e news groups)
$cmd 100 $skip tcp from any to any 119 out via $pif setup keep-state
#Allow out secure FTP, Telnet and SCP using SECURE SHELL
$cmd 110 $skip tcp from any to any 22 out via $pif setup keep-state
#Allow out whois
$cmd 120 $skip tcp from any to any 43 out via $pif setup keep-state
#Allowntp time server
$cmd 130 $skip udp from any to any 123 out via $pif keep-state
#####################################################################
# Inbound traffic Section: from internet into LAN
#####################################################################
#Deny all inbound traffic from non-routable reserved address space
$cmd 300 deny all from 192.168.0.0/16 to any in via $pif #RFC 1918 prvt IP
$cmd 301 deny all from 172.16.0.0/12 to any in via $pif #RFC 1918 prvt IP
$cmd 302 deny all from 10.0.0.0/8 to any in via $pif #RFC 1918 prvt IP
$cmd 303 deny all from 127.0.0.0/8 to any in via $pif #Loopback
$cmd 304 deny all from 0.0.0.0/8 to any in via $pif #Loopback
$cmd 305 deny all from 169.254.0.0/16 to any in via $pif #DHCP auto-config
$cmd 306 deny all from 192.0.2.0/24 to any in via $pif #reserved for docs
$cmd 307 deny all from 204.152.64.0/23 to any in via $pif #sun cluster
$cmd 308 deny all from 224.0.0.0/3 to any in via $pif #Class D & E multicast
#Allow ident
$cmd 315 allow tcp from any to any 113 in via $pif
$cmd 316 allow udp from any to any 113 in via $pif
```

```
#Deny all Netbiosserives 137=name 138-datagram host2nameserve request=81
$cmd 320 deny tcp from any to any 137 in via $pif
$cmd 321 deny tcp from any to any 138 in via $pif
$cmd 322 deny tcp from any to any 139 in via $pif
$cmd 323 deny tcp from any to any 81 in via $pif
#Deny any late arriving packets
$cmd 330 deny all from any to any frag in via $pif
#deny ACK packets that did not match the dynamic rule table
$cmd 332 deny tcp from any to any established in via $pif
#Allow configuration traffic for DHCP
$cmd 360 allow udp from 192.168.46.0/24 to any 68 in via $pif keep-state
#Allow in standard www function for APACHE server
$cmd 370 allow tcp from any to me 80 in via $pif setup limit src-addr 2
#Allow in secure FTP, Telnet and SCP from Internet
$cmd 380 allow tcp from any to me 22 in via $pif setup limit src-addr 2
#Allow in non-secure Telent session from internet(unencrypted traffic)
$cmd 390 allow tcp from any to me 23 in via $pif setup limit src-addr 2
#Skip to location for outbound stateful rules
$cmd 800 divert natdip from any to any out via $pif
$cmd 801 allow ip from any to any
#Default deny rule $cmd 999 deny log all from any to any
############ End of rule set ##############################
```

As can be seen from the sample code above, troubleshooting or editing a longer rule set of up to 1500 rules in depth is not a simple task [16]. Even more, each packet is checked all these rules to see if it matches for it to be allowed or denied into a network.We therefore propose optimizing the rule set using many approaches to limit the size and aid administrators in optimization using the tool that will be designed in this project.

9 Firewall Optimization Techniques

Increased firewall complexity breeds more vulnerability and also reduces availability of network services and applications an enterprise uses [2,8]. The increase in network size, bandwidth, and processing power of networked hosts continues to increase the demand for optimizing firewall operations to improve performance [14]. The following techniques are some of the considerations being explored in optimizing firewall performance.

9.1 Rule set clean up

This approach analyzes the rule set to identify inconsistencies and redundancy in the rule set and removes them. Rules that make no unique contribution to the firewall behavior are removed. These are either Redundant or Shadowed rules [21]. Redundant rules never match packets because there are more preceding rules matched first. Unused rules that have the log option but have no logs showing that they have matched packets. They are therefore candidates for removal when optimizing though their removal may affect firewall behavior later [10]. Remove rules for unused Network groups E.g. if the organization does not have a mail server, SMTP rules can be taken out of the configuration [21].This

should also be done for unused Network objects. Remove unused service objects and disable them on machines E.g. ICMP Type. A blend of this depends on the specific organizational network needs and should be tailored well to achieve better packet filtering that improves throughput [17].

9.2 Rule set re-ordering

This technique relies on the network statistics logged on the firewall or written to a file in a network database [14]. It lists most used rules in decreasing order of usage by hit count and percentage hit count. These rules can then be moved towards the beginning of the rule set to improve performance. Optimize the rule set by ordering rules based on the rule usage data and rule order dependencies that does not alter the firewall behavior [10]. This moves the most used rules toward the beginning of the rule set until they are very close to the source of an order dependency. The complexity with this method is the dependencies problem[9].

9.3 Rule Grouping

It is evident that a major part of the network traffic matches a small subset of the firewall rules[3-21] This therefore calls for selecting these rules and calling them by groups depending on their usage [6]. This scheme divides the filtering policy into two layers of rules, (a) most active rules-those performing the most packet matching and (b) inactive rules-perform much less matching. Rules are checked and if two or more rules are found to have the same matching action, they are merged [21]. This reduces the rule set size and consequently the search time for the filtering algorithm because less rules are inspected when a deny or allow decision is to be made by the firewall.

9.4 Rule Frequency re-ordering

The number of times a rule is triggered is recorded and used to determine matching patterns and arrangement of the whole rule set. Ehab Al-Shaer et.al. [12] propose an adaptive way of dynamically optimizing firewall rule sets using actively calculated statistics. This looks at other filtering categories other than the traditional IP header. Subrata *et.al.*[2] and Ehab Al-Shaer *et.al* [12] presented a technique that uses Internet traffic characteristics to optimize re- wall filtering policies. They have called this method statistical matching. This plus other methods is what this research has adopted in coming up with a tool that will optimize a given rule set to achieve higher throughput [22].

9.5 Rule Editing

Firewalls have thousands of rules and hundreds of IP addresses to take care of. The typical approach is to scan through all these rules in a linear method until a match is found for the packet being inspected [8 ,18, 22]. This is the

inefficiency this research is seeking to address by optimizing the rule set. To optimize therefore, first remove the configuration errors or anomalies in the implemented rule set [13]. This can be done by aggregating rules, rule checking against matched packets, rearranging them, and other algorithm specific ways e.g. Pre-calculating values though it uses up memory [5]. This helps remove these four classed mistakes;

1. Shadowing
2. Redundancy
3. correlation
4. irrelevance

Most techniques used to classify packets use the behavior of filtering rules without taking into account, traffic behavior in their optimization algorithms [12].

9.6 Go To Function

Modern Firewalls come with a feature allowing skipping from one rule or rule set to another rule in a rule set [28]. The go to function is used to switch the search and match ow from the default one(next rule in the list) to the one specified in the go to command. In IP Tables it is called jump [8 , 23]. This causes the search algorithm to skip all rules following until it reaches the specified one called a target. IPFW has this functionality and is called skipto i.e. skip the following rules until the one specified after the skipto [7]. This works in a firewall more like a break or jump in a normal program. This function or command makes optimization possible in that, not all rules in a list are evaluated for matching with a given packet.

10 Rule-set Optimizing Tools

There is work done in the research and design of tools that can be used to optimize firewall rule-sets. These tools have taken different design approaches and incorporate different ways of looking at traffic in relation to rule-sets and security policies.

10.1 Athena FirePAC [1]

This is a proprietary tool owned by Athena Security Inc. It operates on off-line network traffic to perform firewall rule set checks on Cisco, Check Point and Netscreen firewalls. Its functionality is centered on removing vulnerabilities, non-compliance and errors in the firewall configuration. It is configured to perform comparison functionality to see how packet flows have been affected by specific rule and object changes [21]. Its auditing feature finds misconfiguration,

[1] http://www.athenasecurity.net/athenafirepac.html

redundant rules,overshadowed rules, and unused objects that can be removed to optimize the rule-set. It does not perform rule removal automatically but recommends which rules can be removed and rule re-ordering based on audited traffic flows. Therefore, it is more of a monitoring tool and a good one for use by administrators to proceed with rule-set optimization.

10.2 OPTWALL [2]

This is a hierarchical traffic aware framework for firewall rule-set optimization proposed by Mehmud *et al* [2] of the University of Pittsburgh. OPTWALL divides a rule-set into multiple rule sets to reduce the packet matching time to a rule. The proposed tool offers adaption schemes that should dynamically change priority of a rule based on the traffic. This adaptation is based on a heuristic solution that takes initial filter determinations of hit count per rule, hit count of related rules and a random measure of how often those rules match traffic. Those values are then used to calculate the cost of a rule. They have presented a 35% improvement in operational cost of a heavily loaded firewall [2]. This proposed solution has not been produced yet but does show that optimizing a rule set offers a speed up in packet inspection time.

10.3 Policy Advisor [3]

This tool was suggested by Ehab Al-Shaer in his research paper [**?**]. It takes into consideration, rule set design to recommend optimization techniques. Their focus was on correctness and usability of firewall rule-sets using this tool than the computation complexity and optimization of the algorithm. It inspects a rule-set for shadowing anomaly, correlation anomaly, redundancy anomaly, and generalization anomaly. Their anomaly detection algorithm works by applying relationships to rules in a set and compares the fields specified for matching actions to recommend an optimization action. The administrator can then perform those actions from a graphical interface. This too uses offline analysis to process recommendations on the rule set.

11 Observations

Other methods used explore the subject in a similar way as outlined above though their naming conversions may be different. Adel El-Atawy et al [3] propose a technique that uses dynamic statistics collected on firewall policy rules and uses these results to construct a set of rules for denying or allowing traffic. Most research has presented and acknowledged that this is an NP-Complete problem [12-21]. They used an algorithm that processes the firewall policy offline and comes up with optimal solutions. The solution with the least

[2] http://www.chautari.org/forums/index.php?showtopic=12303
[3] http://www.mnlab.cs.depaul.edu/projects/SPA/

processing cost is dynamically selected based on network traffic statistics. This can be likened to [2].
Heuristic based algorithms have been employed in finding the shortest time of matching a packet by doing the least search [9]. These are a combination of most of the techniques discussed above. Disjoint rule set creator for rule 13 set based optimization [2], Direct Acyclical Graphs [9], Filtering Trees[3] among others, all use one or more of the above techniques as a single implementation or in a hybrid form. Most of these methods are based on the standard performance evaluation benchmarks set by the Internet Engineering Task Force in its [1].

12 Research Focus

The motivation for this research is the fact that network traffic is never static [8,13,22]. Therefore, keeping a rule set of fixed length in this dynamic environment creates so much redundancy and inconsistencies. Sometimes it is even difficult to know which requests for important network services are allowed to pass through the firewall because of large rule sets [18,28]. This therefore,impacts on throughput and drastically reduces firewall performance, [11]. This is why the work being done in this research to optimize a rule set for performance gain fits in.

13 Conclusion

Firewall rule set optimization means reducing the time spent checking through a rule set to find a rule that matches a packet against the many rules configured in a given rule set.This takes many approaches like rule grouping, merging, skipping, and rule editing among others.Work to find better ways of optimizing a rule set while maintaining the semantics to have the needed security between the trusted and untrusted network continues. It is evident that not all traffic traversing a network matches all the rules in the rule set. This therefore increases the processing time unnecessarily and drastically impacts on firewall performance with regard to throughput. Rule sets grow to large depths in numbers therefore optimizing them is necessary to reduce matching time. Making a firewall react dynamically to ever changing traffic with varying characteristics is another solution to this problem This has however been described as an NP-Hard task to accomplish [2 - 11]. This is the basis of the work being done in this research.

References

[1] 3511, I. R. Firewall performance benchmarking.

[2] Acharyaý, S., Wangü, J., Geü, Z., Znatiý, T. F., and Greenbergü, A. Traffic-aware firewall optimization strategies.

[3] Adel El-Atawy, Hazem Hamed, E. A.-S. Adaptive statistical optimization techniques for firewall packet filtering.

[4] Adolfo Rodriguez, John Gatrell, J. K. R. P. *TCP/IP Tutorial and Technical Overview*. IBM Redbooks, 2001.

[5] And, M. L. Firewall security: Policies, testing and performance evaluation.

[6] Ehlert, S., Zhang, G., and Magedanz, T. Increasing sip firewall performance by ruleset size limitation. In *PIMRC* (2008), pp. 1–6.

[7] FreeBSD. *Firewalls*. FreeBSD Document Project, 2010, ch. 30, p. 777.

[8] FreeBSD, F. *FreeBSD Handbook 2010*. FreeBSD Document Project, 2010.

[9] Fulp, E. W. Optimization of network firewall policies using directed acyclic graphs.

[10] Gianluca Maiolini, Alessio Nicotra, P. T. A. B. Automated framework for policy optimization in firewalls and security gateways. *Information Assurance and Security* (2009).

[11] Grote, A., Funke, R., and Heiss, H.-U. Performance evaluation of firewalls in gigabit-networks. In *Proc. 1999 Symposium on Performance Evaluation of Computer and Telecommunication Systems. Chicago, Society for Computer Simulation* (July 1999).

[12] Hamed, H., and Al-shaer, E. Dynamic rule-ordering optimization for high-speed firewall filtering. In *Proceedings of the 2006 ACM Symposium on Information, computer and communications security* (2006), ACM Press, pp. 332–342.

[13] Hazelhurst, S. A proposal for dynamic access lists for tcp/ip packet filering.

[14] Hunt, R., and Verwoerd, T. Reactive firewalls - a new technique. *Computer Communications, Elsevier, U.K. Vol 26,No 12,* (2003).

[15] Karen Scarfone, P. H. Nist: Guidelines on firewalls and firewall policy.

[16] Katic, T. Pale, P. Optimization of firewall rules. Information Technology Interfaces.

[17] Kolehmainen, A. Optimizing firewall performance. *Seminar on Internetworking* (2007).

[18] Marmorstein, R. A tool for automated iptables firewall analysis. In *Freenix Track, USENIX Annual Technical Conference* (2005), pp. 71–82.

[19] Márton Illés, T. B. The evolution of the firewall. June 2006.

[20] Paco Hope, Yanek Korff, B. P. *Mastering FreeBSD and OpenBSD Security*. O'Reilly, 2005.

[21] Security, A. Firewall cleanup and optimization.

[22] Shimonski R.J, Shinder D.L, D. S. T. *Best Damn Firewall Book Period*. Syngress, 2003.

[23] Steve Suehring, R. Z. *Linux Firewalls, Third Edition*. Sams Publishing, 2005.

[24] Sunshadowz. Types of firewals. http://www.sunshadowz.com/articles/firewalls, May 2010.

[25] Systems, C. *Evolution of the firewall industry*. Cisco Press, 2002.

[26] Training, I. Firewalls: Overview.

[27] Whitman, M. *Principles of Information Security 2nd Edition*. Course Technology, 2004.

[28] Zhao, L., Shimae, A., and Nagamochi, H. Linear-tree rule structure for firewall optimization. In *CIIT '07: The Sixth IASTED International Conference on Communications, Internet, and Information Technology* (Anaheim, CA, USA, 2007), ACTA Press, pp. 67–72.