Optimization of Firewall Filtering Rules by a Thorough Rewriting

Yi Zhang [1] Yong Zhang [2] and Weinong Wang [3]

[1, 2, 3] Department of Computer Science and Engineering
Shanghai Jiaotong University, Shanghai, 200030, P. R. China
{zhangyi, zhangyong2001, wnwang}@sjtu.edu.cn

Abstract The management of firewalls in today's enterprise network environment is a complex and error-prone task. Effective techniques and tools for administrating the firewall configurations should be available to network administrators. In this paper, we present such a technique by using the geometry technology to model the firewall configurations. Each filtering rule is mapped onto a hyperspace object in a 3-dimensional hyperspace. The semantics of a firewall's configuration could then be visually comprehensible. Our approach enables a thorough rewriting of a firewall's legacy rules so as to largely optimize them. Moreover, the concept of rule anomaly/conflict that is usually defined between two rules is extended to rule anomaly/conflict among two or more rules and can be identified easily in our model.

1 Introduction

Firewalls are the widely deployed mechanism to achieve network security. They can figure out possible network attacks and unauthorized traffic based on a set of rules that are defined according to the organization's security policies and stored in their rule-bases.

Once a firewall has been put into place, administrators have to configure it with appropriate filtering rules to enforce an enterprise's relevant security policies. Configuring firewall correctly is crucial to network security. However, this is a difficult task due to the complexity and dynamic nature of today's enterprise network. More often rules are loaded onto a firewall, they are seldom reviewed and cleaned up. The existences of conflicting and obsolete rules therefore are always inevitable, which would jeopardize the security and decrease the performance of the firewall system overtime. The main reason for the occurrence of this problem is the lack of effective firewall management techniques and tools that are able to help us to analyze and purify the legacy firewall rules efficiently.

Configuration and management of firewalls have received considerable attention in the literature. A large number of techniques have been provided dealing with the detection and resolution of rule conflicts [1-4], verification of filtering rules against organization's security policies [5-7], etc. We argue that most of the above techniques could make only

piecemeal optimization of the legacy firewall rule-base, e.g., by adding the resolusive rules to resolve conflicts, which would increase the rule number and make later rule adding and verification complicated. No existing technique is able to effectively and efficiently perform a thorough rewriting of the legacy firewall rules, and give it a thorough clean up to improve its performance and security.

This paper presents such a technique that enables a thorough rewriting of a firewall's legacy rules. This would considerably reduce the rule number, resolve the rule conflicts and improve the performance of the firewall system. We provide an appropriate modeling technique by representing each filtering rule as a hyperspace object in a 3-dimensional hyperspace. The semantics of a firewall's configuration is then visually comprehensible. Techniques for rewriting filtering rules will also be given in this paper. We will also show how to identify rule conflicts in our model and extend the concept of rule conflict usually defined between two rules to rule anomaly among two or more rules.

The remainder of this paper is organized as follows: Section 2 provides the background information. Section 3 introduces the modeling of filtering rule into the hyperspace. Section 4 discusses the identification of rule anomalies. Section 5 discusses how to rewrite the whole legacy firewall rule-base completely. Section 6 introduces the related work. Finally, section 7 concludes the paper and outlines future research.

2 Background

A firewall is typically located at the frontier of the protected network to control accesses to or from it. A firewall implements a network access policy by forcing connections to pass through the firewall, where they can be examined and evaluated [8]. Network access policies are configured into the rule-base maintained in a firewall in the form of a list of ordered filtering rules. A filtering rule defines the action performed on a specific packet flow. It basically consists of four fields: (source, destination, service, action), where source and destination stand for sender and receiver's IP addresses, respectively. Service consists of a protocol and a port or a range of ports. Action is either "permit" or "deny", standing for whether the corresponding packets are allowed to pass through or not.

There are two important properties of most of today's firewalls. One is the adoption of the closed policy, which denies traffic by default and allows a traffic only if there exists a "permit" rule for it. The other is the order-sensitivity of the rule-base, i.e., the first matched rule applies. The firewall checks from the beginning of the rule-base in sequence until it finds a rule that can apply to a new session.

Within closed policy context, the introduction of "deny" rules is considered as a way to conveniently support exceptions. This would undoubtedly bring rule conflicts. Some of these rule conflicts are desirable because of the purpose of the introduction of "deny" rules. Others are undesirable since they would result in unexpected contradictions to the enterprise's security policies.

When a firewall rule-base contains a large number of rules possibly written by different administrators at different times, the possibility of the occurrence of undesirable rule conflicts is very high. Besides conflicting rules, existence of redundant rules and obsolete rules are always inevitable as well. All of them are usually referred to as rule anomalies in the literature. Al-Shaer and Hamed enumerate five types of rule anomalies as follows [3]:

1) **Shadowing Anomaly** A rule is shadowed when a previous rule matches all the packets that match this rule, such that the shadowed rule will never be activated.

2) **Correlation Anomaly** Two rules are correlated if they have different filtering actions, and the first rule matches some packets that match the second rule and the second rule matches some packets that match the first rule.

3) **Generalization Anomaly** A rule is a generalization of a preceding rule if they have different actions, and if the first rule can match all the packets that match the second rule.

4) **Redundancy Anomaly** A rule is redundant if there is another rule that produces the same matching and action such that if the redundant rule is removed, the security policy will not be affected.

5) **Irrelevance Anomaly** A filtering rule in a firewall is irrelevant if this rule does match any traffic that may flow through this firewall. This exists when both the source address and the destination address fields of the rule do not match any domain reachable through this firewall.

The order-sensitivity property of the rule-base would cause performance problem to firewall systems. This is because order-sensitivity requires the firewall match the packet against filtering rules in sequence from the very beginning of the rule-base until a match is found. The organization and the number of the rules will undoubtedly have significant effect on this matching process. To improve the efficiency of this process, i.e., reducing the search time and space requirements of this process [8], rules must be properly ordered and organized, and the number of rules must be reduced as largely as possible.

3 Modeling of Firewall Filtering Rule

To illustrate our methodology, we consider a hypothetical enterprise network with three subnets, namely, D1, D2, and D3, as shown in figure 1. The “Internet” branch in figure 1 represents all networks external to the enterprise’s network. There is a special external network within the “Internet”, called C, which represents the network of a cooperative enterprise. A firewall is placed at the boundary between the enterprise network and the “Internet”. The configuration of the firewall is shown in table 1.

Each filtering rule is mapped onto an object in a 3-dimentional hyperspace. The three axes of this hyperspace represent the source IP address, destination IP address, and the service port number, respectively. Each such rule object is associated with a color representing the action part of the rule, which is either “black”/ “permit” or “gray”/“deny”.

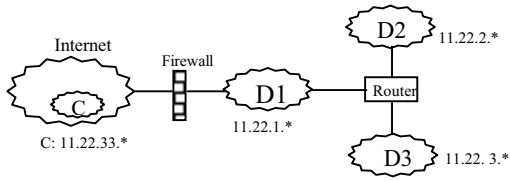


Fig. 1. The example network

Table 1. Filtering Rules of the Example Firewall

Rule No.	Source	Destination	Protocol	Port	Action
1	11.22.1.*	*.*.*.*	tcp	80	permit
2	11.22.2.*	*.*.*.*	tcp	80	permit
3	11.22.3.0/16	*.*.*.*	tcp	80	deny
4	11.22.3.*	*.*.*.*	tcp	80	permit
5	*.*.*.*	11.22.33.16/8	tcp	80	permit
6	11.22.1.0-11.22.2.255	11.22.33.0/16	tcp	80	deny
7	11.22.1.*	11.22.33.*	tcp	21	permit
8	11.22.2.*	11.22.33.*	tcp	21	permit
9	11.22.2.*	*.*.*.*	tcp	21	deny
10	11.22.3.*	*.*.*.*	tcp	21	permit
11	11.22.1.0-11.22.3.255	11.22.33.20	tcp	21	permit
12	11.22.33.*	11.22.44.*	tcp	21	permit
13	*.*.*.*	*.*.*.*	tcp	*	deny

As an example, a filtering rule allowing the video service provided by the network 11.22.33.* to hosts in the subnet 11.22.1.* is as follows:

Source	*Destination*	*Protocol*	*Port*	*Action*
*11.22.1.**	*11.22.33.**	*udp*	*4000-6000*	*permit*

The rule is mapped onto a cube in the hyperspace as shown in figure 2.

Although the protocol information could also be included in the model by extending the dimensionality of the model to four, we argue that this would unnecessarily increase the complexity of the model. Note that the number of the protocols involved is very limited and rules with different protocols can be modeled and analyzed separately because they are considered to be irrelevant to each other from the viewpoint of the purpose we perform related analyses for. This dimensionality reduction makes the resultant model simpler and more intuitive. In most occasions, the model's dimensionality could be further reduced to two with only source and destination axes left. This is because most filtering

rules concern only a few well-known services, e.g., http, ftp, telnet, smpt, etc. These rules regarding different services could also be considered as being independent to each other and could be analyzed separately.

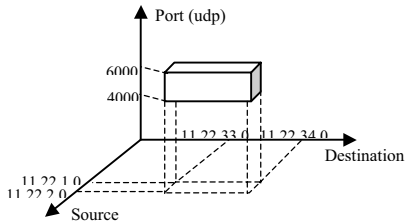


Fig. 2. The mapping of the filtering rule in the 3-dimensional hyperspace

4. Visualizing the Semantics of the Firewall Configuration and Identifying Rule Anomalies

By mapping filtering rules into the hyperspace, we are able to obtain a straightforward and intuitive display of the semantics of a firewall's configuration. Meanwhile, various rule anomalies can be identified easily during the rule mapping process.

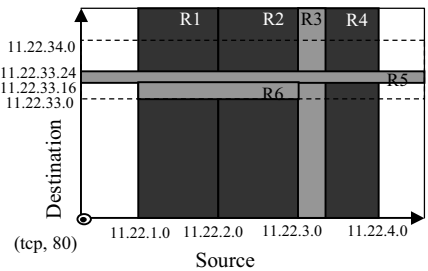


Fig. 3. The mapping of *http* related rules

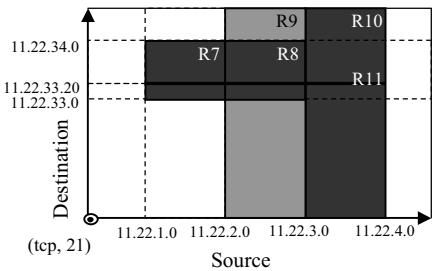


Fig. 4. The mapping of *ftp* related rules

Since the rule order is crucial in determining the semantic of the firewall configuration, rule mapping should be in strict accordance with the order of the rules, i.e., mapping rules one by one from the very beginning of the rule-base. A "permit" rule is mapped to a black object in the hyperspace while a "deny" rule to a gray one. If a later mapped object overlaps the previous ones with different color, the overlapping part would remain its original color. Figure 3 and 4 show the mapping of the filtering rules given in table 1. To reduce the dimensionality complexity, they are displayed separately in two 2-dimensionl hyperspace, which can be regarded as two sections of the 3-dimensional

hyperspace. Figure 3 shows those rules related to http service, (tcp, 80), while figure 4 related to the ftp service, (tcp, 21).

To discover and identify rule anomalies, we investigate several situations during the rule mapping:

(1). **Shadowing Anomaly** If the whole space of the present mapped rule object has been filled with one or more already existing rule objects with the color/action contradict to that of this one, it denotes the occurrence of a shadowing anomaly. Shadowing can be regarded as a serious error in the rule-base because this might cause a legitimate traffic being blocked or an illegitimate traffic being permitted. As an example, Rule 6 is shadowed by Rule 1 and 2 in figure 3. If we wanted to block the access from D1 and D2 to the http services provided by the subset 11.22.33.0/16, Rule 6 should be moved before Rule 1.

(2). **Redundancy Anomaly** If the whole space of the rule object has been filled with one or more already existing rule objects with the same color, it denotes the occurrence of a redundancy anomaly. Redundancy can also be regarded as an error because it unnecessarily increases the size of the rule-base. For example, Rule 11 is redundant to Rule 7, 8, and 10 in figure 4. Therefore, Rule 11 can be safely removed.

(3). **Generalization Anomaly** If the rule object envelops completely one or more already existing rule objects with the different color/action, it is then a generalization of those preceding rules. Generalization is often used to exclude a specific part of the traffic from a general filtering action [3]. It is therefore not necessarily an error in the rule-base. For example, Rule 4 is a generalization rule of Rule 3 in figure 3.

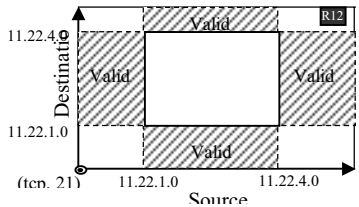


Fig. 5. The valid space of the example firewall

(4). **Correlation Anomaly** If two rule objects with different color/action overlap each other, it denotes the occurrence of a correlation anomaly, which is not necessarily an error. However, the traffic corresponding the overlap part would receive a different treatment if we reverse their order. The administrator should pay much attention to this type of anomaly. For example, Rule 3 and 5 have a correlation anomaly between them. The color/action of the overlap part is determined by the one that precedes the other.

(5). **Irrelevance Anomaly** A firewall is deployed to control access to or from the protected network or networks. If both the source and destination addresses of a rule do not match any domain reachable through this firewall, it can be regarded as an irrelevant rule. Correspondingly, in the hyperspace, each firewall has a valid space. If a rule object falls outside the valid space, it is then an irrelevant rule object. Figure 5 shows the valid

space of the example firewall. We can see Rule 12 falls outside the valid area and can be regarded as an irrelevant rule.

5 The Thorough Rewriting of Filtering Rules

After all filtering rules have been mapped into the hyperspace, including the last default "deny" rule that would fill all the remaining space with the gray color, the semantics of the firewall configuration is now graphically and intuitively displayed to the administrator. Note that the invalid space is covered by gray color entirely irrespective of whether or not it has been covered by irrelevant rules. The blacked spaces in the hyperspace represent the allowed traffics and the administrator could easily validate the firewall configuration against the enterprise's security policy. Error could be easily discovered and corrected by reversing the color of the incorrect space. Then it is time for the thorough rewriting of the filtering rules back into the rule-base. This is an interesting and challenging task involving the combination of the space based on two principles given below:

(1). The number of produced rules should be as little as possible.

(2). The average searching length (the sum of checked rules) for the firewall to match an arriving packet should be as shorter as possible.

The first principle requires the blacked spaces should be represented by as less as possible black and gray cubes in the 3-dimensional hyperspace or rectangles in the 2-dimensional hyperspace. The gray objects are usually used to represent exceptions within the black objects. The second principle requires the produced rules should be arranged in an appropriate order to shorten the searching length. According to the second principle, some auxiliary "deny" rules might be added into the rule-base to shorten the searching length. These auxiliary "deny" rules are not necessary because the default "deny" rule would finally apply to the corresponding traffic in which situation the firewall has to search through the whole rule list. The addition of auxiliary "deny" rules would inevitably increase the size of the rule-base and contradict to the principle one. However, this is acceptable if we could find a trade-off between them.

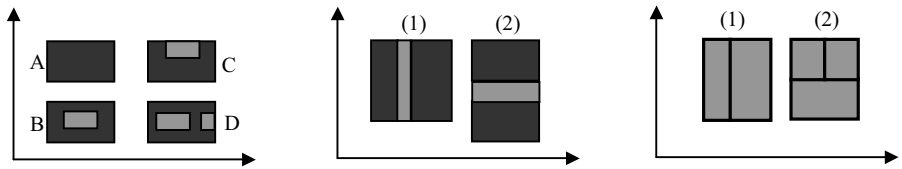
5.1. Representing the Blacked Space in the Hyperspace

For simplicity reason, we only discuss the representation of the blacked space in a 2-dimensional hyperspace, which can be regarded as the section of the 3-dimensional hyperspace. This is acceptable because most related rules regarding a certain service have identical properties along the port axis. If it is not so, these related rules can be split into two or more groups possibly overlapping with each other to meet this requirement.

Now the question becomes how to use as less as possible black and gray rectangles to represent the blacked area in a 2-dimensional hyperspace. It must be pointed out that the optimal resolution is hard to achieve, we could only get a fair optimal solution.

The representing process follows the steps given bellow:

(1). Each separate blacked space in the valid area is represented by one black rectangle and zero or more gray rectangles. The black rectangle covers the whole area of a separate blacked space along its border with the possible inclusion of extra gray spaces. This included gray space can be represented by one or more gray rectangles and are taken as the associated exception rectangles of this black rectangle. The introduction of the notion of the associated exception rectangle is necessary. When mapping the rectangles back to filtering rules, the rules mapped from a black rectangle, which we call the associating rectangle from now on, must be preceded by those rules mapped from its associated exception rectangles.



(a) Representing blacked space **(b)** Combing black rectangles **(c)** Combing gray rectangles

Fig. 6. Representation and combination of rule rectangles

We identify several typical situations, as shown in figure 6(a). The blacked space A is itself a rectangle and can be represented by a black rectangle directly. The blacked space B and C are represented by one black rectangle and one associated exception gray rectangle, respectively. The blacked space D is represented by one black rectangle and two associated exception gray rectangles. More complicated situations are possible, in which, more than two associated exception associated rectangles are needed. However, they can be dealt with in the same way as the simpler situations.

(2). In this step, the black rectangles could be combined if they fall into the two situations shown in figure 6(b). The combination would produce one big rectangle and one additional associated exception gray rectangle. This combination process could be repeated until no possible combination could be performed.

(3). In this step, gray rectangles produced in step 1 and 2 might be combined. Two or more gray rectangles can be joined together to form a larger rectangle if possible. No black rectangle should be produced during the combination. Figure 6(c) illustrates some of the typical situations. The joined gray rectangle becomes the associated exception rectangle of all the involved associating black rectangles.

Figure 7 shows the representing process for rules shown in figure 3. Figure 7(a) displays the effect of the firewall configuration concerning the http service. There are 3 blacked spaces that can be represented by 3 black rectangles, 1, 2, and 3. Black rectangle 1 and 2 can be combined as one big black rectangle 4 and one gray rectangle 5, as shown in figure 7(b). In figure 7(c), black rectangle 4 and 3 can be further combined to form a black rectangle 6 and a gray rectangle 7. Correspondingly, we can get three filtering rules:

*Rule 1: (11.22.1.0-11.22.2.255, *.*.*.*, (tcp, 80), deny);*

*Rule 2: (11.22.3.0/16, *.*.*.*, (tcp, 80), deny);*
*Rule 3: (11.22.1.0-11.22.3.255, *.*.*.*, (tcp, 80), permit).*
Comparing to six rules in figure 3, the number of rules has been largely reduced.

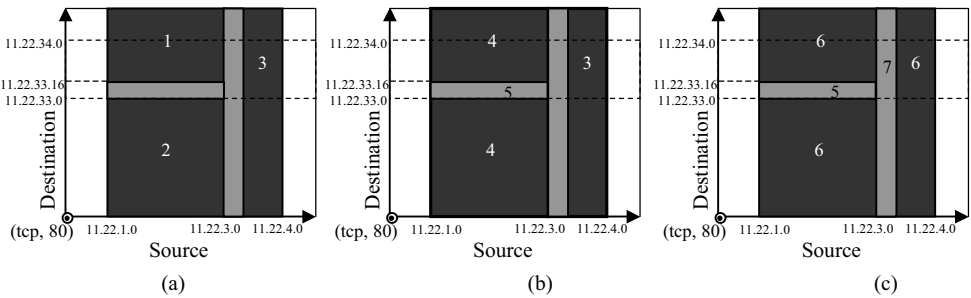


Fig. 7. The representing process of the *http* related filtering rules

5.2 Organizing the Rule Objects in Proper Order

The ordering of rules in the order-sensitive firewall rule-base impacts directly not only the semantics of its configuration, but also the performance of the firewall system. In the previous subsection, each separate black rectangle is associated with zero or more exception gray rectangles that should be placed before it when mapping these rectangles back into rules. However, the order of none associated rectangles is left undetermined, since it has no effect on the semantics of the firewall configuration. In this section, we discuss how to organize the rules in proper order and make further optimization to improve the performance of the firewall system.

The principle of the rule organization is to achieve a shortest average rule searching length. The decision is made based on the statistics of the traffic arriving at the firewall. The rules applying to the heavier traffic should be placed before those applying to less heavy traffic. The statistics should be collected over a period long enough to reflect the real distribution of the traffics.

With the traffic statistics, the rule objects could be ordered in the following steps:

(1). Each type of traffic is mapped in the hyperspace as a dot with a weight corresponding to the statistic of this type of traffic.

(2). The sum of the weights of all the dots contained in each rectangle is calculated. Make sure each dot should be counted only once. If the dot falls within the scope of a gray rectangle that is an associated exception rectangle of a black rectangle, it should not be counted in the associating black rectangle even though it also contains it.

(3). The rectangles (rule objects) are organized in a decreasing order according to the dot weights they contain.

(4). The order should be adjusted. Remember that the associated exception rectangle should be placed before its associating rectangle. If an exception rectangle were placed

behind its associating rectangle because it has a smaller dot weights sum, it should be moved to the position immediately before its associating rectangle.

5.3 Further optimizations

Finally, we could make further optimization by adding auxiliary "deny" rules to improve the performance. They are not necessary because the last default "deny" rule can finally apply to the relevant traffics. However, the firewall has to scan the whole rule list till the last rule. Although the addition of the auxiliary rules will increase the size of the rule-base, it could significantly reduce the average searching length.

So far the quantified definition of the average searching length is given as below.

Definition 1. (Average Searching Length ASL) The average searching length of a firewall calculated on the specific traffic statistics is defined as:

$$ASL=1/N * \sum i * W_i \qquad i=1..n \qquad (1)$$

Where N is the total amount of the statistical traffic, n is the number of filtering rules including the last default "deny" rule, i is the order number of the rule, and W_i is the dot weights associated with the i^{th} rule (object).

The process of identifying and inserting the auxiliary "deny" rules follows the steps given below.

(1). Find the dot with the largest weight among those that do not belong to any existing rectangles.

(2). Draw a rectangle containing this dot and extend it as large as possible without overlapping any existing rectangle to include as many dots as possible. We will usually have two alternatives that extend as large as possible along the two axes respectively. The one that has the larger dot weights will be selected.

(3). Insert the new rectangle at the proper position into the existing rectangle list as described above. The inserting process should start from the end of the list because the list is not in strict decreasing sequence after the order adjustment.

(4). Calculate the new ASL'. If ASL' < ASL, the addition of the new rule is successful and repeat the process from (1) to (4). Otherwise, cancel the insertion of the new rule and stop.

The traffic statistics represented by weighted dots in figure 8 is just for illustration purpose. These traffics relate to the ftp service in our example. Figure 8, corresponding to figure 4, has two black rectangles, I and II, after rectangle combinations. The dot weights contained in rectangle I is 20+30=50, and in rectangle II, it is 50+100+120+130=400. Therefore, rectangle II should be placed preceding rectangle I. The average searching length is calculated as:

ASL=(400*1+50*2+450*3)/900=2.056

Now, if the auxiliary gray rectangle III is added as shown in figure 8, the weights it contains is 300. It should be placed before rectangle I and after II. The new ASL is:

ASL'=(400*1+300*2+50*3+150*4)/900=1.944

ASL' is smaller than ASL and the addition of rectangle III is acceptable. However, if the auxiliary gray rectangle IV is further added, the ASL would become:

ASL"=(400*1+300*2+80*3+50*4+70*5)/90=1.989

ASL" is larger than ASL'. Thus, the addition of rectangle IV is unacceptable.

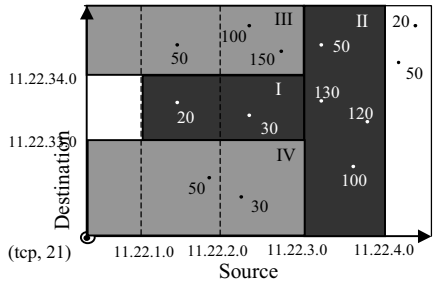


Fig. 8. The statistics of the *ftp* related traffics

6 Related Work

Bartal et al. present a firewall management toolkit called Firmato in [1]. Roles are introduced for expressing security policies by attributing each role a set of capabilities. Each such capability represents an allowed service. A host or a host group can assume several roles according to the security policies. Firmato provides an entity-relationship framework for representing both the security policy and the network topology. However, modeling network topology by the entity-relationship is an indirect and limited way. Therefore, it is difficult in their method to identify appropriate firewalls for the enforcement of generated rules, so they have to replicate the generated rules onto every firewall in the network.

Mayer et al. present a firewall analysis tool, Fang, which allows the administrator to discover and test the global firewall policy [6]. It could be regarded as an extension of Firmato and uses the same basic methodology. It therefore shares the same strengths and weakness with Firmato.

Al-shaer and Hamed provide a subtle method for the optimizing of a firewall rule-base [3]. They represent the filtering rules by a policy tree, which is a single-rooted tree. Every path from the root to a leaf presents a filtering rule and vice versa. In this way, relations and anomalies among filtering rules can be easily discovered. However, their method can only analyze the relationship between two rules and cannot give a thorough optimization of a legacy firewall rule-base.

There are many other works that has been reported in this area. Guttman provides a Lisp-like definition language for defining filtering policy [7]. However, it ignores the effect of rule ordering. Eronen and Zitting use an expert system to verify the functionality

of filtering rules by performing queries [2]. All these tools can help us to verify the correctness of the firewall rules. However, they base their effectiveness on the administrator's experience and expertise to write proper queries to identify various problems with the firewall rules.

7 Conclusions and Future Work

This paper presents a technique to thoroughly rewrite a firewall's legacy rule-base. Each filtering rule is mapped onto an object in a hyperspace. The semantic of firewall configuration is visually comprehensible. Geometry technology can therefore be utilized to process these rule objects and then map them back to filtering rules into rule-base after appropriate optimization. Rule anomalies can be easily discovered and resolved during the mapping of filtering rules. In addition, we extend the concept of rule anomaly usually defined between two rules in the literature, to rule anomaly among two or more rules.

The methodology given in this paper suits for optimizing the rule-base for a single firewall. However, a typical enterprise network today might employ multiple firewalls connecting multiple zones of a network to each other. Cooperation of all these firewalls is important to ensure the security of the whole network. We will extend our method to handle distributed firewalls, which will emphasize on the cooperation of all the firewalls in an enterprise network.

Reference:

1. Bartal Y., Mayer A., Nissim K., and Wool A.: Firmato: A Novel Firewall Management Toolkit. In Pro. Of 20th IEEE Sym. On Security and Privacy, OakLand, CA, pp.17-31, 1999.
2. Eronen P. and Zitting J.: An Expert System for Analyzing Firewall Rules. Proc. 6th Nordic Wksp. Secure IT-Systems (NordSec 2001), Nov. 2001.
3. Al-Shaer E. S. and Hamed H. H.: Modeling and Management of Firewall Policies. In IEEE Transactions on Network and Service Management, Vol. 1(1), April 2004.
4. Eppstein D. and Muthukrishnan S.: Internet Packet Filter Managementand Rectangle Geometry. Proceedings of 12th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), January 2001.
5. Hari B., Suri S. and Parulkar G.: Detecting and Resolving Packet Filter Conflicts. Proceedings of IEEE INFOCOM'00, March 2000.
6. Mayer A., Wool A., and Ziskind E.: Fang: A Firewall Analysis Engine. In Proceedings of the 2000 IEEE Symposium on Security and Privacy, 2000.
7. Guttman J. D.: Filtering Postures: Local Enforcement for Global Policies. In Proc. IEEE Symp. on Security and Privacy, Oakland, CA, 1997.
8. Panko R.: Corporate Computer and Network Security, Prentice Hall, 2003.